In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 381-387, Sendai, Japan, Sept 2004.

Automatic Synthesis of Communication-Based Coordinated Multi-Robot Systems

Chris Jones and Maja J Matarić
Computer Science Department, University of Southern California
941 West 37th Place, Los Angeles, CA 90089-0781 USA
Email: {cvjones|maja@robotics.usc.edu}

***Abstract*— To enable the successful deployment of task-achieving multi-robot systems (MRS), coordination mechanisms must be utilized in order to effectively mediate the interactions between the robots and the task environment. Over the past decade, there have been a number of elegant experimentally demonstrated MRS coordination mechanisms. Most of these mechanisms have been task-specific in nature, typically providing only empirical insights into coordination design and little in the way of systematic techniques to assist in the design of coordinated MRS for new task domains. To fully realize the potentials of MRS, formally-grounded systematic techniques amenable to analysis are needed in order to facilitate the design of coordinated MRS. We address this problem by presenting a formal framework for describing and reasoning about coordination in a MRS. Using this principled foundation, we are developing a suite of general methods for *automatically synthesizing* the controllers of robots constituting a MRS such that the given task is performed in a coordinated fashion. This paper presents a method for the automatic synthesis of a specific type of controller, one that is stateless but capable of inter-robot communication. We also present a graph coloring-based approach for minimizing the number of necessary unique communication messages. The synthesis of such communicative controllers provides a means for assessing the uses and limitations of communication in MRS coordination. We present experimental validation of our formal approach of controller synthesis in a multi-robot construction domain through physically-realistic simulations and in real-robot demonstrations.**

I. Introduction

The study of multi-robot systems (MRS) has received increased attention in recent years. This is not surprising as continually improving technology has made it realistic to consider the deployment of MRS consisting of increasingly larger numbers of robots. With the growing interest in MRS comes the expectation that, at least in some important respects, multiple robots will be superior to a single robot in achieving a given task. Potential advantages of MRS over a single robot are frequently expounded in the literature. For example, total system cost, it is frequently claimed, may be reduced by utilizing multiple simple and cheap robots as opposed to a single complex and expensive robot. Furthermore, the inherent complexity of some task environments may *require* the use of a heterogeneous group of robots as the necessary capabilities are too substantial to be met by a single robot. Finally, multiple robots may provide increased robustness by taking advantage of inherent parallelism and redundancy.

However, the utilization of MRS poses potential disadvantages and additional challenges that must be addressed if MRS are to present a viable and effective alternative to single robot systems. Of paramount importance is the complexity introduced by the management of multiple, interacting robots. In order for a task-achieving MRS to be effective, the robots' actions must be carried out in a coordinated fashion and directed towards the achievement of the given task. A MRS lacking effective coordination is less likely to present a solution that is more desirable or effective than a single robot solution. Correctly executing a task in a multi-robot system presents fundamentally different issues from doing so in a single robot system. In a MRS, it cannot be assumed that a particular robot is always aware of the task progress resulting from the actions of other robots. Formally, from the perspective of an individual robot in a MRS, the task environment is highly non-stationary.

From a few robots performing a manipulation task, to tens of robots exploring a large space, to thousands of ecosystem monitoring nano-robots, as the number of robots in the system increases, so does the necessity and importance of coordination. Coordination is defined as "the act of regulating and combining so as to produce harmonious results" [1]. In the context of MRS, coordination is the process of appropriately regulating the robots' actions such that a given task or goal is successfully achieved. Our work is focused on *distributed* MRS, in which each robot operates independently under local sensing and control, with coordinated group behavior arising out of local interactions between the robots and the task environment. The design of such coordinated distributed MRS can be quite challenging because unexpected collective behaviors may emerge due to unanticipated ramifications of the robots' local interactions. Nonetheless, many elegant hand-crafted coordination mechanisms have been demonstrated, both in simulation and on physical robots. The nature of the employed mechanisms have taken many forms, seemingly limited only by the ingenuity of the designer.

Unfortunately, MRS coordination design still remains more of an art than a science. The coordination mechanisms employed are usually task-specific. Designers typically provide little formal analysis as to expected system performance and rarely provide informal explanations as to why the employed mechanism is more appropriate than possible alternatives. The design of coordination mechanisms needs to be systematic and formally grounded in order to move it into the realm of science and fully realize the advantages of MRS over single robot systems. Thus, *a central challenge facing the MRS community is the design of principled methods for the synthesis and analysis of coordination mechanisms*.

To address this issue, we have developed a formalism which provides a principled framework for precisely defining and reasoning about the intertwined entities involved in any task-achieving MRS – the world, task definition, and the capabilities of the robots themselves, including action selection, sensing, maintenance of internal state, and inter-robot communication. Using this principled foundation, we are developing a suite of general methods by which to *automatically synthesize* the controllers of robots constituting a MRS such that a given task is performed in a coordinated fashion. Each of these methods is directed toward the synthesis of a specific type of controller. We taxonomize controllers based on the following three characteristics: deterministic or probabilistic action selection (DA/PA), using internal state or stateless (IS/NIS), and capable or incapable of inter-robot communication (Comm/NComm). Our synthesis methods for controllers across this taxonomy provide more than just pragmatic tools for building coordinated MRS. Given their formal grounding, they also provide a means to systematically determine the fundamental limitations of each type of controller, to understand the inherent relationships between different controller types, to contribute methods to systematically reduce one controller type to another, and to provide insight into the general requirements necessary for achieving different forms of coordination. We aim to facilitate formal answers to fundamental questions, such as: '*In what conditions is it necessary for the robots to be able to communicate?*', '*In what conditions is communication alone insufficient?*', and '*When are the use of internal state and communication interchangeable?*'.

In our previous work, we presented a method for the synthesis of DAct-IS-NoComm controllers and defined situations in which internal state is useful to achieve the necessary coordination [10] and a macroscopic MRS modeling approach directed to the analysis of homogeneous MRS composed of robots executing DAct-IS-NoComm controllers [11]. In this paper, we present a new method for the synthesis of a different type of controller, a DAct-NoIS-Comm. In addition, we show when and why DAct-NoIS-Comm controllers are useful in achieving the desired coordination. Furthermore, we provide a graph coloring-based approach for minimizing the number of unique communication messages used. We present experimental validation of our formal approach to DAct-NoIS-Comm controller synthesis in physically-realistic simulations and in real-robot demonstrations in a multi-robot construction task domain.

II. Related Work

Related work on the synthesis and analysis of MRS coordination mechanisms includes, but is not limited to, the work of Donald [5] that presents the derivation of information invariants aimed at defining the informational requirements of a given task and ways in which those requirements can be satisfied in a robot controller. Parker [19] extends the idea of information invariants by defining equivalence classes among task definitions and robot capabilities to assist in the choice of an appropriate controller class. Dudek et al. [6] present a taxonomy which classifies multi-robot systems based on communication and computational capabilities. Martinoli et al. [14] presents a general methodology by which the collective behavior of a group of mobile robots can be accurately studied using a simple probabilistic model. Balch [4] presents hierarchic social entropy, an information theoretic measure of robot team diversity in an effort to understand the role of heterogeneity in MRS coordination. Gerkey and Matarić [8] present a principled framework and an analysis methodology for the formal study of multi-robot task allocation. Lerman and Galstyan [13] present a mathematical model of the dynamics of collective behavior in a multi-robot adaptive task allocation domain. Alternative approaches to the synthesis of MRS controllers can be found in evolutionary methods [7] and learning methods [15, 18]. There also exist a number of MRS design environments, control architectures, and programming languages which assist in the design of task-achieving coordinated MRS [16, 3, 2].

III. Definitions and Notation

We now provide necessary definitions for the formalism. The *world* is the environment in which the MRS is expected to perform a defined task. We assume the world is Markovian, the state is an element of the finite set S of all possible states, and is populated by a finite set of homogeneous robots R. An action performed in the world by a single robot is drawn from the finite set A of all possible actions. An *observation* x made by a robot, drawn from the finite set of all observations X, consists of accessible information external to the robot and formally represents a subset of the world state. The world is defined by a probabilistic state transition function $P(s, x, a, s')$ that states the probability the world state at time $t+1$ is s' given the world state at time t is s and a robot making observation x executes action a. We note that the world state transition function involves an observation because the tasks we consider are spatial in nature and the physical location where an action is performed is just as important as the action itself. In this representation, an observation x is equated with the spatial location where the action a is performed. Therefore, an action a, executed upon the observation of x_i, will transition the world differently than the same action a, performed upon the observation of x_j. The probabilistic *observation function* $O(s, x)$ gives the probability the observation x will be made by a robot in world state s. We assume that an observation x may only be made at one physical location in the world in a state s. We define a *task*, assumed to be Markovian, as a set of n ordered world states $T_s = \{s_0, s_1, ..., s_n\}$ which must be progressed through in sequence. We assume the initial state of the world is s_0 and the task terminates when the world state s_n is achieved. We define *correct task execution* to be the case where, for all task states $s_i \in T_s$, $i < n$ the only actions executed by any robot are those that transition the world state to s_{i+1}. Therefore, we define an observation and action pair, x and a, to be correct for task state s_i if $P(s_i, x, a, s_{i+1}) > 0$. We assume that an observation x and action a cannot be correct for more than one task state. The robots we consider do not maintain any internal state or representation; however, they are capable of inter-robot communication. The set of all possible communication messages a robot may send and receive is denoted by the set C. The actual communication content or mechanism of each message is not important, we only require that each message is uniquely distinguishable and instantaneously received by other robots. For example, in our implementation each message is just a unique integer broadcast over the wireless network connecting the robots. The communication message a robot is currently sending is denoted as c_s. We assume a robot may receive any number of messages simultaneously. The set of messages a robot is currently receiving is denoted as c_r. Two functions define a robot's behavior in the world, known collectively as the robot's *controller*. The *action function* $Act(x, c_r, a)$ specifies the probability a robot will execute action a given it is currently observing x and receiving communication messages c_r. The *communication function* $Comm(x, c)$ specifies the probability a robot will send communication message c given that it is currently observing x. Although the controller is modeled with probabilistic functions to maintain consistency with our other work, in this paper these functions are treated as deterministic, i.e., Act and $Comm$ will always return either 0 or 1.

IV. DAct-NoIS-Comm Controllers

In this section we present a systematic procedure for synthesizing a DAct-NoIS-Comm controller, a stateless controller with deterministic action selection and inter-robot communication capabilities. This entails defining the robots action and communication functions. We also discuss the uses and limitations of such controllers in the facilitation of coordination in MRS.

A. Synthesis

There are four high-level steps in the synthesis process: **1)** synthesize a baseline DAct-NoIS-NoComm controller, **2)** identify situations in which communication can be used to better facilitate coherent coordination, **3)** assign specific communication messages to each of these situations, and **4)** incorporate these communication assignments into a DAct-NoIS-Comm controller. The full synthesis process is given by the procedure *Build_DAct-NoIS-Comm_Controller*, shown at the bottom of Figure 1.

Step 1: We synthesize a DAct-NoIS-NoComm controller, which is simply a stateless, non-communicative controller that we will augment with communication to synthesize the DAct-NoIS-Comm controller. The process of synthesizing a DAct-NoIS-NoComm controller is given by the procedure *Build_DAct-NoIS-NoComm_Controller*, shown at the top of Figure 1. For each $s_i \in T_s$, the synthesis procedure adds a rule to the action function of the form $Act(x, \{\}, a) = 1$, such that x and a are correct for task state s_i.

procedure Build_DAct-NoIS-NoComm_Controller()
  for all $a \in A, x \in X$ do
    $Act(x, \{\}, a) = 0$
  endfor
  for all $s_i \in T_s$ do
    for all $a \in A, x \in X$ s.t. $(O(s_i, x) > 0 \wedge P(s_i, x, a, s_{i+1}) > 0)$ do
      $Act(x, \{\}, a) = 1$
    endfor
  endfor
end procedure Build_DAct-NoIS-NoComm_Controller

procedure Build_DAct-NoIS-Comm_Controller()
  Build_DAct-NoIS-NoComm_Controller()

  for all $s_i \in T_s$ do
    $X_c(s_i) = \{x_0, x_1, ..., x_n\}$ s.t. $\forall x \in X_c(s_i) \nexists s_k (O(s_i, x) > 0 \wedge k \neq i \wedge O(s_k, x) > 0)$
    $X_a(s_i) = \{x_0, x_1, ..., x_m\}$ s.t. $\forall x \in X_a(s_i) \exists a (O(s_i, x) > 0 \wedge Act(x, \{\}, a) > 0 \wedge P(s_i, x, a, s_{i+1}) > 0)$
  endfor

  Graph_Color($\bigcup_{\forall s_i \in T_s} \{X_c(s_i) - X_a(s_i)\}$)

  for all $s_i \in T_s$ do
    for all $x \in \{X_c(s_i) - X_a(s_i)\}$ do
      $Comm(x, Assigned\_Comm(x)) = 1$
    endfor
    $c = \{\}$
    for all $x \in \{X_c(s_i) - X_a(s_i)\}$ do
      $c = c \bigcup Assigned\_Comm(x)$
    endfor
    for all $x \in X_a(s_i), a \in A$ s.t. $(Act(x, \{\}, a) = 1)$ do
      $Act(x, \{\}, a) = 0$
      $Act(x, c, a) = 1$
    endfor
  endfor
end procedure Build_DAct-NoIS-Comm_Controller

Fig. 1. Procedure for synthesizing a DAct-NoIS-Comm controller.

However, such a DAct-NoIS-NoComm controller leaves room for error if x and a are correct for some task state s_i but there exists another task state s_j where x and a are *not* correct and $O(s_j, x) > 0$. In such situations, an MRS composed of robots with DAct-NoIS-NoComm controllers cannot enforce the action sequence necessary for correct task execution. This is a common problem with purely reactive controllers in sequential task domains. In the DAct-NoIS-Comm synthesis steps that follow, we incorporate the use of communication to improve coordination in these situations. Due to sensing and action uncertainty, the addition of communication cannot guarantee correct task execution, but it can be used to increase the probability of correct task execution.

Step 2: We define a set of observations that will serve as the basis of the DAct-NoIS-Comm controller's communication function. For each task state $s_i \in T_s$, we define a set of observations $X_c(s_i)$ (Figure 1, lines 13-16), the union of which can *only* occur in state s_i. We also define $X_a(s_i)$, a set of observations such that, for each $x \in X_a(s_i)$, there exists an action a, where x and a are correct for s_i. We note that if $\exists s_i, s_j \in T_s$ such that $\{\forall x(O(s_i, x) > 0)\} \subseteq \{\forall x(O(s_j, x) > 0)\}$, then the state s_i is fundamentally unobservable. In such a situation, one cannot guarantee that a MRS composed of robots executing a DAct-NoIS-Comm controller will correctly execute the task, even in the absence of sensing and action uncertainty.

Step 3: We assign specific communication messages to all observations in $\{X_c(s_i) - X_a(s_i)\}$ for each $s_i \in T_s$, as defined in Step 2. The simplest solution to this problem is to assign a specific, unique communication message to each observation in this set. However, in many MRS communication bandwidth can be quite limited (e.g., in a MRS composed of autonomous underwater vehicles), and so it is advantageous to minimize the number of bits transfered in each communication message; the more unique communication messages used, more bits in each communication message will be required to uniquely identify the message. Furthermore, in some MRS communication is not achieved through digitized radio transmissions but through other means such as the release of chemicals, sound, or light. In such cases, the number of unique communication messages a robot is capable of sending and receiving can be very constrained. Therefore, to minimize the actual number of *unique* communication messages needed, we use a *graph coloring approach*. This step is *not* used to minimize the number of instances in which communication is used, which is decided by the process in Step 2. Although graph coloring is NP-complete, there are a number of well studied heuristics that provide understood bounds on resulting solution quality [17]. Furthermore, the graph coloring approach is desirable in many domains to reduce the number of unique messages required, but it is not absolutely required as the direct assignment of unique messages to each necessary observation is suitable.

The problem of assigning unique communication messages to a given set of observations $O = \bigcup_{\forall s_i \in T_s} \{X_c(s_i) - X_a(s_i)\}$ can be reduced to a graph coloring problem as follows. First, a graph G, consisting of a set of vertices V and a set of edges E, both initially empty, is created. Next, a vertex is added to V for each observation in O. Then, we add edges to E between each pair of vertices in V for which the associated observations interfere with each other. The test for interference between two observations x_i and x_j is given by the function $I(x_i, x_j)$ shown in Equation 1. Now a standard graph coloring algorithm may be applied to G in which the color assigned to a vertex in V corresponds to a specific communication message assigned to the observation represented by that vertex. The function $Assigned\_Comm(x)$ as used in Figure 1 returns the communication message assigned to the observation x as a result of the graph coloring in that step.

$$I(x_i, x_j) = \begin{cases} 1, & \text{if } \exists s \in T_s(O(s, x_i) > 0 \wedge O(s, x_j) > 0), \\ 1, & \text{if } \exists s_u, s_v \in S \exists x \in X_a(s_v)(O(s_u, x) > 0 \wedge \\ & \quad x_j \in \{X_c(s_v) - X_a(s_v)\} \wedge O(s_u, x_i) > 0), \\ 1, & \text{if } \exists s_u, s_v \in S \exists x \in X_a(s_v)(O(s_u, x) > 0 \wedge \\ & \quad x_i \in \{X_c(s_v) - X_a(s_v)\} \wedge O(s_u, x_j) > 0), \\ 0, & \text{otherwise.} \end{cases} \tag{1}$$

Step 4: We synthesize the DAct-NoIS-Comm controller by augmenting the DAct-NoIS-NoComm controller synthesized in Step 1. This is accomplished by adding the communication function and appropriately modifying the action function such that an action is not executed unless all necessary communications are being simultaneously received. Through the graph coloring approach presented in Step 3, a specific communication message was assigned to each observation in the set $\{X_c(s_i) - X_a(s_i)\}$ for each $s_i \in T_s$. The controller communication function is constructed (Figure 1, lines 19-21) by adding a communication rule for all $x \in \{X_c(s_i) - X_a(s_i)\}$ for all $s_i \in T_s$ of the form $Comm(x, Assigned\_Comm(x)) = 1$, where $Assigned\_Comm(x)$ is the specific communication message assigned to the observation x in Step 3. Such a communication rule will cause the robot to send the communication $Assigned\_Comm(x)$ every time the observation x is made. The action function is modified (Figure 1, lines 22-30) so that for each rule of the action function, $Act(x, \{\}, a) = 1$, where x and a are correct for a state s_i is modified to become $Act(x, c_r, a) = 1$, where c_r is the set of specific communication messages mapped in Step 3 to the observations in $\{X_c(s_i) - X_a(s_i)\}$. All probabilities not explicitly declared for the controller are 0.

B. Discussion

Due to imperfect robot action and sensing capabilities, there is no guarantee that the synthesized DAct-NoIS-Comm controller will correctly execute the given task. However, the use of a communicative controller leads to significantly improved performance over a similar non-communicative controller, as we demonstrate in Section V. Importantly, the synthesized DAct-NoIS-Comm controller is, however, guaranteed to correctly execute the task in the absence of sensing and action uncertainties.

A DAct-NoIS-Comm controller synthesized by the procedure in Figure 1 is only one, and certainly not the only, way in which communication can be used to facilitate coordination. In fact, from a pragmatic standpoint, a MRS composed of robots executing such DAct-NoIS-Comm controllers has many disadvantages that other forms of communicative MRS may not exhibit. For example, the efficiency of the MRS in terms of time to task completion will usually be quite poor, as many events have to happen simultaneously before actions can be performed. A part of this problem stems from the fact that DAct-NoIS-Comm controllers are stateless. Allowing the robots to retain some form of non-transient internal state or representation would likely improve the system performance in a number of

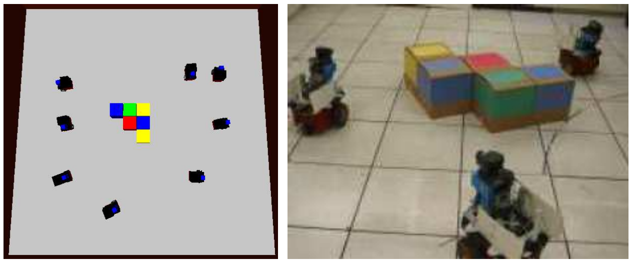

Fig. 3. (left) Snapshot of an 8-robot experiment in simulation. (right) Snapshot of a 3 robot real-world experiment.

respects. However, from the perspective of identifying and understanding the fundamental requirements of coordination, MRS composed of stateless, communicative robots are quite interesting. By isolating the use of communication, analysis of such MRS provides a means to better understand when and why communication is able to facilitate coordination and when it is insufficient. Knowledge of the limitations of communication helps identify when and why the integration of other controller features, such as internal state, becomes necessary.

V. Case Study: Coordination in Multi-Robot Construction

The formalism and synthesis method described in Sections III and IV-A, respectively, are task in-specific. In this section we apply the formalism and synthesis method to a specific multi-robot construction task domain. Using both physically-realistic simulation and physical robots, we experimentally demonstrate and validate our approach to the synthesis of coordinated MRS through the use of DAct-NoIS-Comm controllers. This task requires the sequential placement of a series of cubic colored bricks into a planar structure. For all examples used in this section, a brick's color is denoted by the letters `R`, `G`, `B`, and `Y` which stand for `Red`, `Green`, `Blue`, and `Yellow`, respectively. The construction task starts with a seed structure, which is a small number of initially placed bricks forming the core of the structure.

Our simulation experiments were performed using Player and the Gazebo simulation environment. Player [9] is a server that connects robots, sensors, and control programs over a network. Gazebo [12] simulates a set of Player devices in a 3-D physically-realistic world with full dynamics. Together, the two represent a high-fidelity simulation tool for individual robots and teams that has been validated on a collection of real-robot robot experiments using Player control programs transferred directly to physical Pioneer 2DX mobile robots. In all simulation experiments 8 robots were used, and in all real-robot experiments 3 robots were used. The robots were either realistic models of or actual ActivMedia Pioneer 2DX mobile robots. Each robot, approximately 30 cm in diameter, is equipped with a differential drive, a forward-facing 180 degree scanning laser rangefinder, and a forward-looking color camera with a 100-degree field-of-view and a color

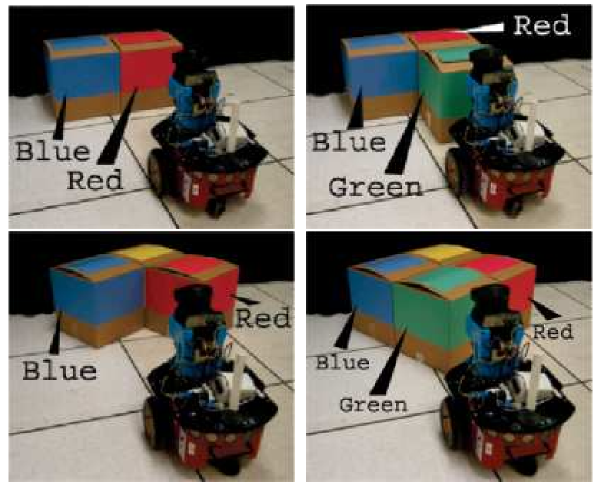

Fig. 4. Example observations and actions in the construction domain. (top left) Robot in position to make observation `<FLUSH R B>`. (top right) Immediately after robot performs action `<G RIGHT FLUSH R B>`. (bottom left) Robot in position to make observation `<CORNER R B>`. (bottom right) Immediately after robot performs action `<G CORNER R B>`.

blob detection system. The bricks are taller than the robot's sensors, so the robots can only sense the local bricks on the periphery of the structure (i.e., robots do not have a birds-eye view of the entire structure). Figure 3 shows snapshots of our simulation and real-world experimental setup.

A. Formal Definitions for Construction Task

In order to cast the construction task in the formal framework presented in Section III, we now define the world, task definitions, observations, and actions for the construction task domain. The *world* state is defined as a specific spatial configuration of bricks, including the color of each brick. A construction *task* is defined as a *sequence* of brick configurations (i.e., world states), providing a specific construction sequence. Observations in the construction domain are made up of the spatial configuration and color of bricks in the field-of-view of the robot's laser rangefinder and color camera and within an appropriate range and bearing. Two categories of observations can be made. The first is two adjacent, aligned bricks. A situation in which such an observation is made is shown in Figure 4 and is denoted as `<FLUSH R B>`. The second is two adjacent bricks forming a corner. A situation in which such an observation is made is shown in Figure 4 and is denoted as `<CORNER R B>`. The observations `<FLUSH R B>` and `<FLUSH B R>` constitute two different observations in which the spatial relationship between the Red and Blue bricks are switched. A similar point holds for the observations `<CORNER R B>` and `<CORNER B R>`. Due to uncertainty in sensing, the probability a given `FLUSH` observation will be mistaken as a `CORNER` observation is 1.1% and the probability a given `CORNER` observation will be mistaken as a `FLUSH` observation is 11.5%.

Actions are the placement of individual bricks to the growing structure. We do not consider construction tasks

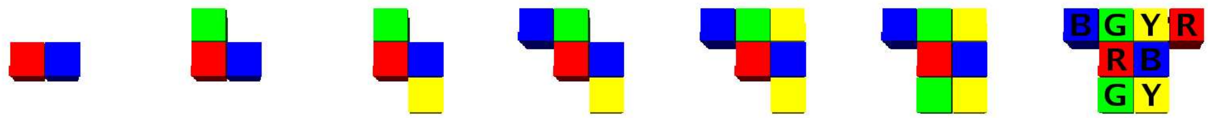

Fig. 2. The sequence of world states defining a construction task as seen from overhead (a view not available to the robots in the MRS), from s_0 to s_6, left to right. The last state provides the color of each brick.

Action Function
Act(<FLUSH R B>, {}, <G RIGHT FLUSH R B>) = 1
Act(<FLUSH B R>, {c_0}, <Y RIGHT FLUSH B R>) = 1
Act(<FLUSH R G>, {c_1}, <B LEFT FLUSH R G>) = 1
Act(<CORNER G B>, {c_2}, <Y CORNER G B>) = 1
Act(<CORNER Y R>, {c_3}, <G CORNER Y R>) = 1
Act(<FLUSH Y B>, {c_0}, <R RIGHT FLUSH Y B>) = 1

Communication Function
Comm(<FLUSH R G>, c_0) = 1
Comm(<FLUSH B Y>, c_1) = 1
Comm(<FLUSH B G>, c_2) = 1
Comm(<FLUSH G Y>, c_3) = 1

TABLE I

SYNTHESIZED ACTION AND COMMUNICATION FUNCTIONS FOR THE CONSTRUCTION TASK SHOWN IN FIGURE 2. $c_0, c_1, c_2, c_3 \in C$.

```
procedure Execute_DAct-NoIS-Comm_Controller()
  repeat forever
    x ← current observation
    c_r ← communications being received
    if ∃c(Comm(x, c) > 0) then
      send communication c with prob. Comm(x, c)
      execute a random walk
    else if obstacle nearby then
      execute obstacle avoidance
    else if ∃a(Act(x, c_r, a) > 0) then
      execute action a with prob. Act(x, c_r, a)
      execute a random walk
    else
      execute a random walk
    endif
  endrepeat
end procedure Execute_DAct-NoIS-Comm_Controller
```

Fig. 5. High-level controller integrating the synthesized action and communication functions for the construction task domain.

in which robots may remove bricks from the structure nor those in which sub-structures consisting of multiple bricks may be connected together. Other actions performed by the robots, such as moving through the environment, do not affect the world state and are therefore not explicitly considered. Three categories of actions can be executed. The first is the placement of a brick on the right side (from the perspective of the acting robot) of a pair of adjacent, aligned bricks. The immediate result of such an action is demonstrated in Figure 4 and is denoted as <G RIGHT FLUSH R B>. The second is identical to the first except that the brick is placed on the left side of a pair of adjacent, aligned bricks. This action is denoted as <G LEFT FLUSH R B>. The third is the placement of a brick in the corner formed by two other bricks. The immediate result of such an action is demonstrated in Figure 4 and is denoted as <G CORNER B R>. Due to uncertainty in action, the probability an attempted CORNER action will succeed is 78% and the probability an attempted FLUSH action will succeed is 98.5%.

B. *Synthesized Controllers*

We applied our systematic method for synthesizing DAct-NoIS-Comm controllers to the construction task shown in Figure 2. The synthesized action and communication functions are given in Table I. As can be seen, the graph coloring approach for minimizing communication messages was able to reduce the number of unique messages needed from 5 to 4. The reduction in this case was minimal as many observations can be made in a large proportion of the task states. However, this technique can quite significantly reduce the number of unique messages needed in many task domains. Figure 5 shows how the action and communication functions are integrated into the controller. Since the Avoid and Random Walk behaviors do not change the world state, they do not impact the controller synthesis procedure. The controllers for the construction task shown in Figure 2 were implemented on a group of 8 simulated robots. A total of 300 simulation runs were conducted. As expected, due to significant uncertainty in sensing and actions, each trial did not result in correct task execution. Over the 300 experiments, correct task execution was achieved in 29.4% of the trials. This represents a significant improvement over the non-communicative DAct-NoIS-NoComm controller, which resulted in only 0.9% of trials being correctly executed. We note that if there was no uncertainty in sensing and action, the synthesized DAct-NoIS-Comm controller would be *gauranteed* to correctly execute the task, whereas no such gaurantee could be made for the non-communicative DAct-NoIS-NoComm controller. For real-robot verification of the feasibility of the synthesized DAct-NoIS-Comm controller, we also implemented it on a group of three actual Pioneer 2DX mobile robots. A limited number of real-world trials were correctly executed which verified the feasibility of the DAct-NoIS-Comm controller in the real-world. We emphasize that the real-robot experiments were performed in order to show that our formalism and synthesis method are not merely abstract concepts but successfully capture the difficult issues involved in real-world embodied MRS, thus providing a grounded and pragmatic tool for the description, synthesis, and analysis of coordinated MRS.

We note that our robots do not have the ability to manipulate bricks in simulation or with physical robots. To address this issue in simulation, when a robot wants to execute a brick placement action, it commands the simulator to place a brick of a given color at a given location relative to the robot's current pose. In real-robot experiments, we manually placed the appropriate brick in response to the robot's command (e.g., "Place yellow brick in the corner formed by the red and blue bricks directly in front of my position").

VI. Conclusions

The successful deployment of a task-achieving MRS depends on effective mechanisms for coordinating the robots' actions and interactions. To date, demonstrated coordination mechanisms have largely been designed in a task-specific manner with little formal analysis of the fundamental issues involved in MRS coordination. In this paper, we presented a formally-grounded method for designing coordinated MRS. Specifically, we introduced a systematic method for synthesizing a specific type of robot controller we call a DAct-NoIS-Comm controller, one that is stateless but capable of inter-robot communication. This controller, executed by all robots in a MRS, achieves coordinated execution of a given task. From the broader perspective of our over-arching research goal aimed at a formal investigation of coordination in MRS, the synthesis of a coordinated MRS with each robot executing a DAct-NoIS-Comm controller provides insight into how and why inter-robot communication can be used to facilitate coordination. Through experimental results in a multi-robot construction domain, we have shown how the use of communication can significantly improve MRS coordination.

References

[1] *Webster's Revised Unabridged Dictionary*. C. & G. Merriam Co., 1996.

[2] R. Alur, R. Grosu, Y. Hur, V. Kumar, and I. Lee. Modular specification of hybrid systems in charon. In *Proceedings of the 3rd International Workshop on Hybrid Systems: Computation and Control*, pages 6–19, Pittsburgh, PA, Mar 2000.

[3] R. Arkin and T. Balch. Aura: Principles and practive in review. *Journal of Experimental and Theoretical Artificial Intelligence*, 9:175–189, 1997.

[4] T. Balch. Measuring robot group diversity. In T. Balch and L. E. Parker, editors, *Robot Teams: From Diversity to Polymorphism*, pages 93–135. AK Peters, 2002.

[5] B. R. Donald. Information invariants in robotics. *Artifical Intelligence*, 72(1–2):217–304, 1995.

[6] G. Dudek, M. Jenkin, E. Milios, and D. Wilkes. A taxonomy for multi-agent robotics. *Autonomous Robots 3*, (4):375–397, 1996.

[7] D. Floreano. Patterns of interactions in shared environments. In *Proceedings of the Second European Conference on Artificial Life*, pages 347–366, Brussels, Belgium, May 1993.

[8] B. Gerkey and M. Matarić. Multi-robot task allocation: Analyzing the complexity and optimality of key architectures. In *Proceedings of the IEEE International Conference on Robotics and Automation*, pages 3862–3867, Taipei, Taiwan, Sep 2003.

[9] B. Gerkey, R. Vaughan, K. Stoey, A. Howard, G. Sukhatme, and M. Matarić. Most valuable player: A robot device server for distributed control. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 1226–1231, Maui, Hawaii, Oct 2001.

[10] C. Jones and M. Matarić. Towards a multi-robot coordinaton formalism. In *Proc. of the Workshop on the Mathematics and Algorithms of Social Insects*, pages 60–67, Atlanta, Georgia, 2003.

[11] C. Jones and M. Matarić. Synthesis and analysis of non-reactive controllers for multi-robot sequential task domains. In *Proceedings of the International Symposium on Experimental Robotics*, Singapore, Jun 2004.

[12] Nathan Koenig and Andrew Howard. Design and use paradigms for gazebo, an open-source multi-robot simulator. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 2149–2154, Sendai, Japan, Sep 2004.

[13] K. Lerman and A. Galstyan. Macroscopic analysis of adaptive task allocation in robots. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 1951–1956, Las Vegas, Nevada, Oct 2003.

[14] A. Martinoli, A. Ijspeert, and F. Mondada. Understanding collective aggregation mechanisms: From probabilistic modeling to experiments with real robots. *Robotics and Autonomous Systems*, 29:51–63, 1999.

[15] M. Matarić. Designing and understanding adaptive group behavior. *Adaptive Behavior*, 4:51–80, Dec 1995.

[16] M. Matarić. Issues and approaches in the design of collective autonomous agents. *Robotics and Autonomous Systems*, 16(2–4):321–331, Dec 1995.

[17] R. Nelson and R. Wilson. *Graph Colourings*. Longman Scientific and Technical, 1990.

[18] L. Parker. Alliance: An architecture for fault-tolerant multi-robot cooperation. *Transactions on Robotics and Automation*, 14(2):220–240, Apr 1998.

[19] L. Parker. Toward the automated synthesis of cooperative mobile robot teams. In *Proceedings of SPIE Mobile Robots XIII*, volume 3525, pages 82–93, Boston, MA, Nov 1998.