

# The International Journal of Robotics Research

<http://ijr.sagepub.com>

---

## **Toward Cooperative Team-diagnosis in Multi-robot Systems**

Michael D.M. Kutzer, Mehran Armand, David H. Scheid, Ellie Lin and Gregory S. Chirikjian  
*The International Journal of Robotics Research* 2008; 27; 1069  
DOI: 10.1177/0278364908095700

The online version of this article can be found at:  
<http://ijr.sagepub.com/cgi/content/abstract/27/9/1069>

---

Published by:



<http://www.sagepublications.com>

On behalf of:



Multimedia Archives

**Additional services and information for *The International Journal of Robotics Research* can be found at:**

**Email Alerts:** <http://ijr.sagepub.com/cgi/alerts>

**Subscriptions:** <http://ijr.sagepub.com/subscriptions>

**Reprints:** <http://www.sagepub.com/journalsReprints.nav>

**Permissions:** <http://www.sagepub.co.uk/journalsPermissions.nav>

**Citations** <http://ijr.sagepub.com/cgi/content/refs/27/9/1069>

---

**Michael D.M. Kutzer<sup>1</sup>**  
**Mehran Armand<sup>1</sup>**  
**David H. Scheidt**

Johns Hopkins University Applied Physics Laboratory  
11100 Johns Hopkins Road  
Laurel, MD 20723  
{Michael.Kutzer, Mehran.Armand}@jhuapl.edu

**Ellie Lin<sup>2</sup>**

Carnegie-Mellon University, Robotics Institute  
5000 Forbes Avenue, Pittsburgh, Pennsylvania

**Gregory S. Chirikjian**

Johns Hopkins University, Department of Mechanical Engineering  
3400 North Charles Street,  
Baltimore, Maryland 21218  
gregc@jhu.edu

# Toward Cooperative Team-diagnosis in Multi-robot Systems

## Abstract

*Research in man-made systems capable of self-diagnosis and self-repair is becoming increasingly relevant in a range of scenarios in which in situ repair/diagnosis by a human operator is infeasible within an appropriate time frame. In this paper, we present an approach to the multi-robot team diagnosis problem that utilizes gradient-based training of multivariate Gaussian distributions. We then evaluate this approach using a testbed involving modular mobile robots, each assembled from four electromechanically separable modules. The diagnosis algorithm is trained on data obtained from two sources: (1) a computer model of the system dynamics and (2) experimental runs of the physical prototypes. Tests were then performed in which a fault was introduced in one robot in the testbed and the diagnostic algorithm was queried. The results show that the state predicted by the diagnostic algorithm performed well in identifying the fault state in the case when the model was trained using the experimental data. Limited convergence was also demonstrated using training data from an imperfect dynamic model and low data sampling frequencies.*

**KEY WORDS**—diagnosis, fault diagnosis, self diagnosis, team diagnosis, robot team repair, mobile robot, robotics, multi-robot system, modular robot, gradient-based training, particle filters.

## 1. Introduction

The ability to self-diagnose and self-repair is becoming necessary in a range of man-made systems for which a priori or in situ repair/diagnosis by a human operator is not feasible within an appropriate time frame. Interplanetary spacecraft, underwater vehicles and smart buildings are just a few examples of the many systems that can benefit from the behavioral flexibility and autonomy offered by the ability to self-diagnose and self-repair.

Cooperative multi-robot teams address critical problems that are beyond the capabilities of current, standalone robotic systems.

1. In a well designed multi-robot team, a failure in one member will not result in total system failure. For example, the distribution of a mission-critical sensor package over multiple independent robotic vehicles enables sampling of at least part of a desired dataset despite, in the worst possible case, the complete loss of one or more of the robotic delivery platforms. Moreover, the source of malfunction can be diagnosed by other members of the team, and possibly repaired in situ using spare parts carried by the robots. In the future, it is conceivable to

---

1. Also affiliated with: Johns Hopkins University, Department of Mechanical Engineering, Baltimore, MD

2. Also affiliated with: NASA Robotics Academy, NASA Goddard Space Flight Center, Office of Higher Education, Greenbelt, MD

take this concept even further by producing robots capable of complex repair procedures utilizing in situ resources. In both scenarios, the overall robustness and reliability of the system are greatly improved when multi-robot teams are used. In comparison, critical failures in a standalone robotic system can have a dire effect on its operational capabilities. This, in turn, can result in failures to achieve mission objectives.

2. In a team-diagnosis and team-repair scheme, using a combination of spare parts and in situ resource utilization (ISRU) robotic teams can, in principle, produce greatly extended mission lives. When applied with a high level of autonomy, repair strategies that require communication with a human operator can be minimized. This is especially applicable to the robotic exploration of space. These missions carry both a high cost and a high risk of failure. By sending multiple robots with capabilities including diagnosis and repair, the risk of failure following successful deployment in space or on planetary surfaces is drastically reduced. Furthermore, the resulting extended mission life and hardware sustainability can make missions like these far more effective.

The topic of robotic self-repair has been reported by several researchers. Various works have used the concept of modular redundancy to develop reconfigurable robots with repair capabilities. The work performed by Murata et al. (1994) produced a series of two-dimensional (2D) building blocks, each called a Fractum. Using electromagnets, these Fracta were able to arrange themselves into a series of geometric shapes. Yim et al. (2001) presented work on the distributed control of a class of 3D metamorphic robots named Proteo. Chirikjian et al. (1996) developed the concept of a metamorphic robotic system, and proposed methods of bounding the minimal required movements to achieve desired configurations. Rus (1998) presented a four-degree-of-freedom (DoF) 'robotic molecule' capable of creating various lattice structures in 3D space. Shen et al. (2002) presented a hormone-inspired method of active communication and active distributed control that was evaluated on the CONRO self-reconfigurable robot. Yoshida et al. (1999) addressed and experimented with both self-assembly and self-repair in two dimensions on the Fractum robot and in three dimensions on a large, DC-motor-driven robot. In essence, all of these homogeneous modular robots perform repair by discarding a failed module. Once the failed module is discarded, they return to their original configuration or function. We note that this repair strategy is specifically tailored to systems comprising homogeneous modules, thereby potentially limiting their scope.

To our knowledge, the only other report of cooperative robots attempting self-repair is the work of Bererton and Khosla (2000). In that work, the authors developed a cooperative team consisting of two robots. A fork-lift robot, using an ultrasonic

localization system, docks with a stationary robot to remove a failed module. The work focuses on the methodology of docking. As such, it does not address the problem of team-diagnosis, nor does it address the issues regarding complete disassembly/reassembly using a team of robots with similar capabilities.

A body of literature is also developing in the area of self-assembly and self-replication. Specifically, Gracias et al. (2001) and Bohringer (2003) have presented self-assembly applied to the meso- and micro-scales. These works rely heavily on passive elements driven by forces that cannot be realistically scaled to macro-systems such as those discussed in this work. Klavins (2007) demonstrated self-assembly in a controlled environment using programmable units floating on an air table. While this work is successful on the macro-scale, it relies heavily on a controlled environment for passive assembly. The research performed by Park et al. (2004) and Chirikjian et al. (2002) is a step towards the development of macro-scale self-replicating robots. Specifically, a robot in a structured environment replicated itself by correctly connecting a series of modules. As will be seen shortly, there are some similarities between the subsystems of the model developed in that work and the subsystems of the prototype robots developed here. Zykov et al. (2005) presented an overview of their work on reproducing robots made up of actuated modular cubes. While that work addresses autonomous replication, the use of homogeneous modular systems is similar to the previously mentioned topics of self-reconfiguration and self-repair.

Much of the existing research in the area of diagnosis is concentrated in systems containing a single entity. The use of particle filters for fault diagnosis specifically has gained popularity in the area. Dearden and Clancey (2001) present their application of particle filtering to data collected from the Marsokhod rover. Specifically, they applied particle filtering methods to determine the most likely values of system variables and modes of a Marsokhod wheel using sensor data. Washington (2000) presents the MaKSI method of state estimation and fault diagnosis, and its application to the Marsokhod rover. Specifically, he describes the MaKSI method as a combination of Kalman filtering to represent system dynamics and Markov model representations to estimate state probabilities.

Doucet et al. (2000) present their work on Rao-Blackwellized particle filtering. Here, they present a method of increasing efficiency by exploiting the structure of dynamic Bayesian Networks, but do not discuss its potential application to fault diagnosis. Blackmore et al. (2005) present their comparison of the Rao-Blackwellized particle filtering to a k-best enumeration method, and then present a hybrid combination of the two playing on the strengths of both methods. Verma et al. (2004) test a series of particle-filter-based methods to diagnose wheel states on the Hyperion robotic platform. While each of these particle-filter-based works presents a method or series of methods that are successful in the diagnosis of at least

part of a robotic platform, each focuses on the real-time or near real-time diagnosis of faults. Further, each requires a detailed model capable of estimating future sensor values used to weight particles. These methods are inherently different from our own in that they focus on real-time fault detection, and require detailed dynamic modeling that includes transient system behavior.

Other examples of methods for diagnosing systems containing a single entity include high-fidelity simulations as a foundation for diagnosis as described by Satish Kumar (2001) and diagnostic agent techniques described in Fröhlich et al. (1999) and Roos et al. (2003). A focused survey on single system diagnosis as applied to wheeled robots in unknown environments is documented by Duan et al. (2005). A more general review of fault detection is presented by Venkatasubramanian et al. (2003a, b, c).

Of the relatively few published works on fault diagnosis in multi-robot systems, a multi-robot work in the area of coordination failures is presented by Kalech et al. (2006). That work focuses on net team failures and does not specifically address the diagnosis of physical faults in any individual robot. In a separate example, Daigle et al. (2006) utilize distributed diagnosis techniques based on the analysis of transient signatures (the mapping of abrupt faults to transients in dynamic system behavior; Mosterman and Biswas (1999) in a multi-robot system. They exploit the physical coupling between robots that results from the cooperative task of moving a rigid object. Reliance on a cooperative task, physical coupling and high data sampling rates makes their approach and problem statement different from the present work.

Model-based approaches have been effective in performing diagnosis of complex systems. Our effort focuses on the application of a model-based technique to the domain of multi-robot teams in which each team member is itself a modular robot. Our motivation stems from a desire for cooperative systems to perform diagnosis and reconfiguration, assuming they exhibit the following properties: (1) they should be composed of constituent robots whose actions can be enumerated prior to deployment; (2) the behavior of the robots while performing a set action can be accurately modeled.

Considering team repair, the interactions between constituent robot pairs should be described through classical dynamics, electrical flow (when sharing power), fluid flow (when sharing propellant), network models (when sharing information) and contact mechanics. The combination of discrete nonlinear models and continuous linear models within the same overall model was partly inspired by the hybrid modeling approach described by Henzinger (2000). However, unlike methods relying on high-fidelity simulation where the goal is an accurate representation of the detailed behavior of the system as it undergoes a fault, our objective is to represent the minimum amount of behavioral information needed to make an effective diagnosis in the steady state (ignoring transients). That is, rather than addressing the inherent complexities associated

with modeling the transition to a fault state, our methods rely only on observations of the fault state after it has already occurred.

In this paper, we present a method as a first step toward cooperative team diagnosis. While our approach was developed for multiple robots, it was demonstrated using observations taken of a single robot. The full problem of cooperative team-diagnosis and team-repair will be addressed in future works.

## 2. Testbed

The testbed for this project consists of four identical two-wheel independent-drive robots controlled using a tethered computer. Each robot (Figure 1) consists of four individual modules physically connected using a series of rare-earth magnets. These magnetic connections, shown in Figure 2, each have two parts: *fixed docking sites* and *dynamic docking sites*. The combination of a fixed docking site with a dynamic docking site creates a magnetic connection. Our dynamic docking sites utilize a switched servo mechanism actuating a sliding series of the rare-earth magnets. The switched servo mechanism relies on a micro switch to trigger the sliding action of the rare-earth magnets. This micro-switch is strategically placed so that, when the gripper closes on a module, it will press the switch. The two set positions of the sliding magnets allow the site to repel and attract a fixed docking site (Figure 2) when the servo switch is pressed and released.

Our Central Processing Unit labeled CPU (Figure 1) consists of a microprocessor (Lego RCX), communication tower, a computer (not shown) and three fixed docking sites. The specific computer-RCX interaction was implemented to simulate an I/O data acquisition box controlled using Matlab. The onboard sensory package of each assembled robot consists of two rotation sensors (one on either wheel assembly) and two touch sensors on the gripper. The four identical robots were designed so that each can physically attach and remove modules from another robot as needed.

In the experiments that are reported in this paper, we employ two of the four robots with one ailing robot being diagnosed by one fully functioning robot. For future reference, we refer to the robot undergoing diagnosis as *ailing* even though it is not known if the robot is actually ailing or what its ailment might be until after the diagnosis is performed by the other robot.

We would expect that robots deployed in the field would be outfitted with a camera and vision system capable of producing observations equivalent to a camera placed overhead. As a proxy for this capability, any robot performing a diagnosis in our experimental trials has access to information obtained from an overhead camera and a computer running our diagnostic algorithm. The ailing robot consists of any one member of the previously mentioned testbed. Figure 3 shows a representative view of our experimental setup for diagnosis. The computer running the diagnostic algorithm begins diagnosis

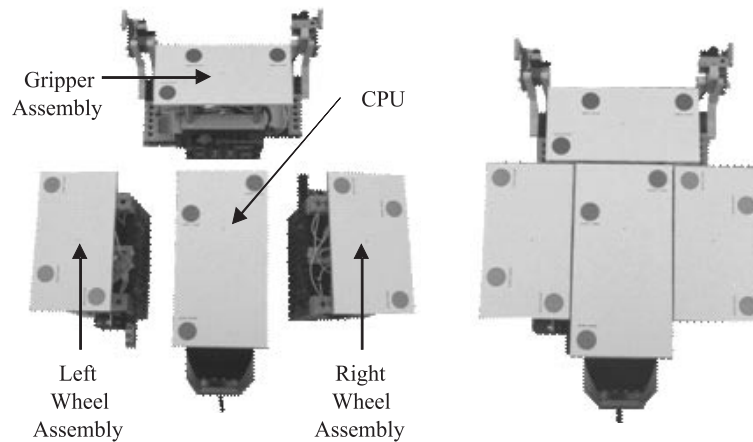


Fig. 1. Top view of our modular testbed; left: disassembled and right: assembled.

by sending a command to the computer controlling the member of the testbed. As a result, the computer controlling the member of the testbed sends a prescribed set of inputs  $X$  that are executed by the testbed. Simultaneously, the computer running the diagnostic algorithm begins viewing the scene using the overhead camera. Upon completion, the information taken from the overhead camera is processed by the computer running the diagnostic algorithm, and converted to a set of observations  $Y_i$  that are used to make the diagnosis.

### 3. General Approach

We define *cooperative team-diagnosis* as the act of multiple robots jointly attempting to identify faults in one or more robots performing a given task. Similarly, we define *cooperative team-repair* as the process of determining the correct course of action based on the identified problem, and completing the appropriate action to solve the problem. To facilitate in repair, robots should be modular (i.e. comprising a series of removable subsystems). A description of our modular robotic testbed is described in Section 2.

Our proposed approach to cooperative team-diagnosis is shown in Figure 4. For simplicity, only a two-robot system is shown. Robot A in the figure is a fully functioning robot, while Robot B is undergoing diagnosis as a result of a diagnostic command being given. We define the *diagnostic command* as a command given when there is a reason to suspect that a fault might have occurred (i.e. when a robot displays unexpected or erratic behavior that is observed by other robots). A *diagnostic maneuver* is the motion resulting from a prescribed set of inputs  $X$  executed by a robot suspected of not being fully functional.

In Figure 4, the *physical system* corresponds to the ailing robot executing the diagnostic maneuver. We define the *available sensors* of an individual robot to be those used to produce

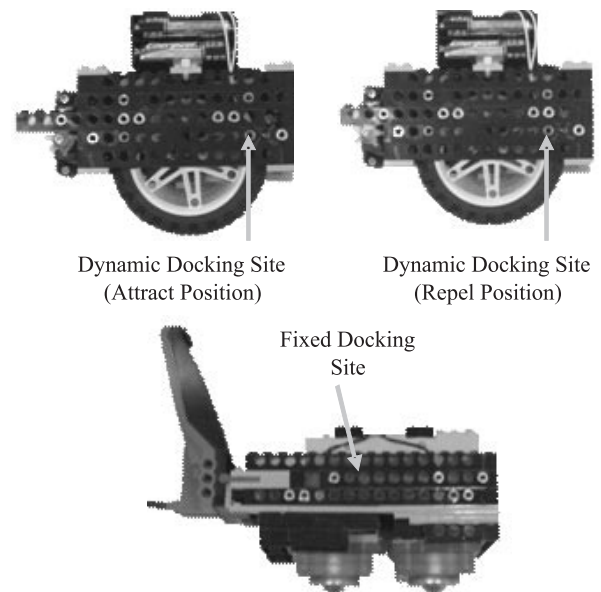


Fig. 2. Examples of fixed and dynamic docking sites of the modular robot. Top left: wheel module dynamic docking site in attract position and top right: repel position. Lower: right-fixed docking site of CPU.

a set of observations  $Y_i$  of the diagnostic maneuver. The *diagnostic algorithm* contains probabilistic inputs trained using estimates from the robot system model and/or experimental observations taken with a known state applied to the robot in a representative environment. Using observations  $Y_i$  from the diagnostic maneuver, the diagnostic algorithm is used to produce an estimate of the current state  $\xi_i$  of Robot B. Team fault diagnosis then combines the individual diagnoses using their corresponding probabilities to define the most likely state of



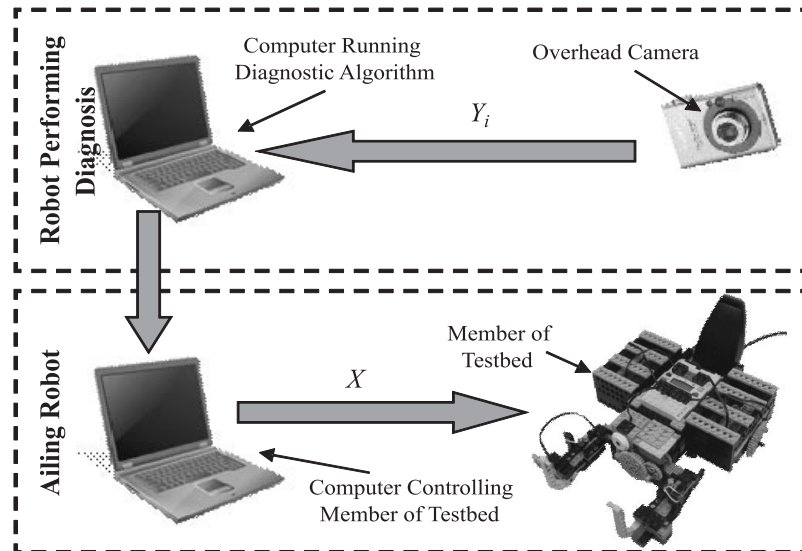


Fig. 3. Representative view of the experimental setup for diagnosis.

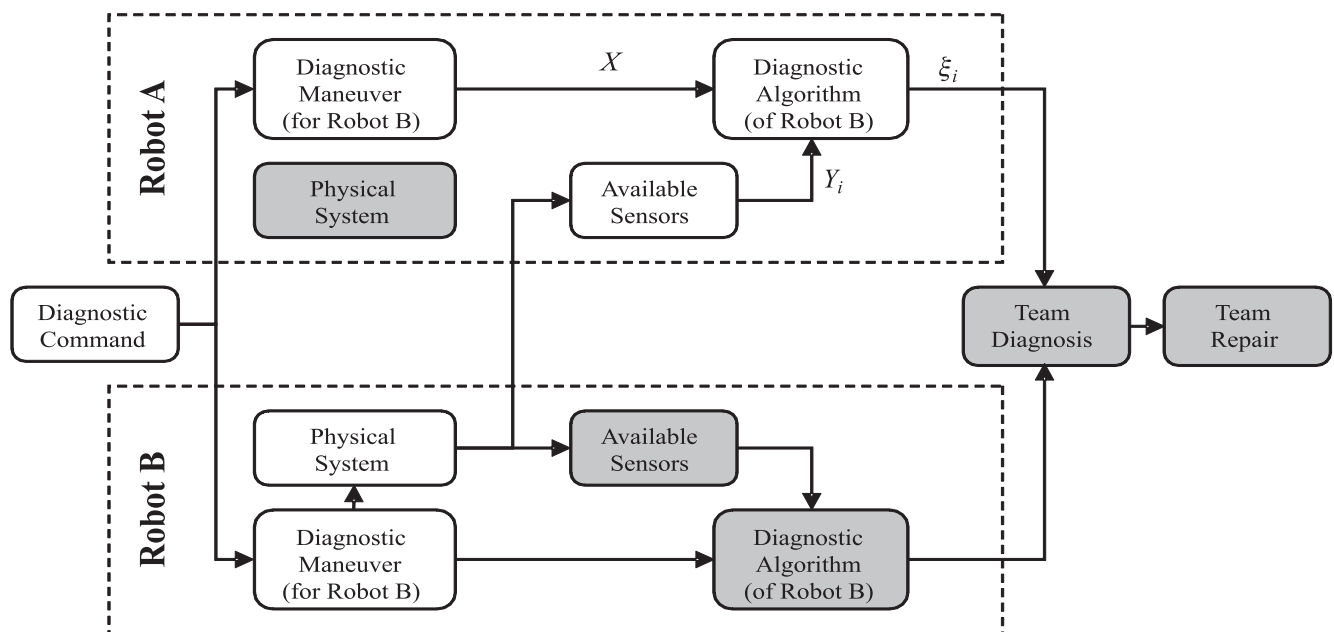


Fig. 4. Presented approach to team-diagnosis. Robot B is performing a diagnostic maneuver while performing a self-diagnosis and also being diagnosed by Robot A.

Robot B. Finally, team repair coordinates robots to remove applicable parts and replace them.

The white blocks in Figure 4 are specifically addressed in this paper, with the emphasis placed on the block that takes in  $X$  and  $Y_i$  and returns  $\xi_i$ . Robot A is used solely as an observer of the scene, and computationally to provide a diagnosis based on its observations. In our experimental testbed, Robot A com-

prises the overhead camera (the available sensors) and a computer running our diagnostic algorithm. These can be viewed as general resources available to any team member performing a diagnosis. Robot B represents any one robot from our aforementioned testbed. Note that, in the case of a homogeneous team such as our testbed, the need for robot-specific diagnostic maneuvers and algorithms as shown in Figure 4 (i.e. 'Diagnosis

tic Maneuver for Robot B' and 'Diagnostic Algorithm of Robot B') is not necessary. Instead, a single diagnostic maneuver and diagnostic algorithm can be used by each member of the team.

In the following subsections we describe our approach towards team-diagnosis. Briefly, the approach involves: (1) the development of a model of the robots and characterization of the system states, (2) finding an optimal diagnostic maneuver for the ailing robot given the available sensors on the diagnosing robots, (3) model-based training of the diagnostic algorithm using a gradient-descent method, and (4) diagnosis of the fault(s). In theory, this approach is not limited by the number or type of modules on each robot, the number of robots in the team, or the number of simultaneous faults of the ailing robot.

### 3.1. Modeling

Following Dearden and Clancey (2001) and Washington (2000), we create a system model tuned to correspond to each of the discrete states  $\xi_i$  of the system. As a concrete example, Appendix 2 lists all 43 discrete states that we have identified for our system. Each discrete state specifies a continuous model. Given a set of continuous system inputs ( $X = (x_1, x_2, \dots, x_m)^T$ ) applied from times  $t = t_0$  to  $t = t_F$ , a model will produce a series of system responses  $\{y_1, y_2, \dots, y_S\}$ . The value of  $S$  is specified by the method of observation (e.g. for a rangefinder  $S = 1$  corresponding to a single distance measurement and for an overhead camera  $S = 3$  for a planar pose measurement). Using a time step of  $\Delta t$  based on the slowest available sample rates, observations are obtained at the times  $t_k = t_0 + k\Delta t$  for  $k = 1, 2, \dots, f$  where

$$f = \left\lceil \frac{t_F - t_0}{\Delta t} \right\rceil.$$

Our  $S \cdot f$  dimensional vector of system observations relative to the initial values is:

$$Y_i = \left( (y_1|_{t_1} - y_1|_{t_0}) \quad (y_2|_{t_1} - y_2|_{t_0}) \quad \dots \quad (y_S|_{t_1} - y_S|_{t_0}) \quad \dots \quad (y_S|_{t_f} - y_S|_{t_0}) \right)^T.$$

Given a series of  $n$  system states ( $\xi_1, \xi_2, \dots, \xi_n$ ), a *global system model* contains  $n$  continuous models, each describing the system behavior in a specific state as shown in Figure 5.

For a system comprised of modular robots, we define a series of module states observable by the available sensors. Referencing our testbed (described in Section 2), and using an overhead camera as a sole source of observations, we only consider states that affect the movement of the robot. That is, we only consider module states that change the robot's ability to negotiate its environment (move forward, backward, turn

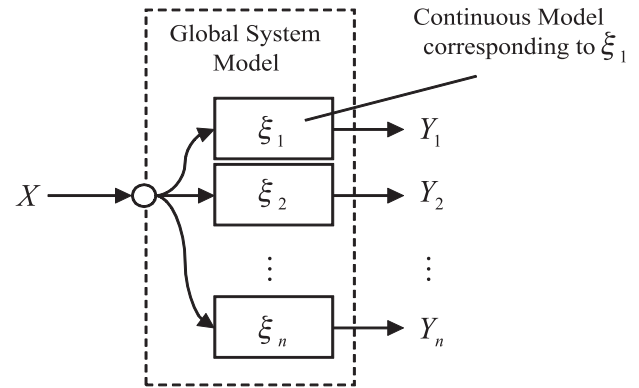


Fig. 5. The observations of the global hybrid system model depend on both system inputs  $X$  and discrete states  $\xi$ . It comprises a series of continuous models, one for each of the  $n$  system states ( $\xi_1, \dots, \xi_n$ ). The state-specific model corresponding to  $\xi_i$  is capable of simulating the observation  $Y_i$ .

clockwise, etc.) or articulate portions of itself (open and close its gripper). The set of all system states is then defined as the set of all possible combinations of the module states, lumping states that result in the same observable behavior together.

Due to the limited observations available with any sensing modality (and an overhead camera in particular), we note that the total number of diagnosable states is limited. Of the potentially observable module states (shown in Appendix 1), we note that certain states are indistinguishable using observations. For example, using only the observations from an overhead camera, it is impossible to differentiate between a wheel slipping on the ground such that it transfers no forces to the robot, a wheel that is physically stuck and a wheel whose motor is not responding to commands. As a result, we lump our observable module states into sets of diagnosable module states, or states that we can accurately predict using available observations.

System states are created using all possible combinations of module states. However, when creating these combinations we again see redundancies in observable behaviors that we lump into single *classes of states*. An example of this is the 'Unresponsive' system state. This state lumps together all combinations of module states containing any one of the following: {'CPU-Unresponsive'}, {'Left Wheel-Unresponsive' and 'Right Wheel-Unresponsive'}, {'Left Wheel-Removed' and 'Right Wheel-Removed'}, {'Left Wheel-Unresponsive' and 'Right Wheel-Removed'}, or {'Left Wheel-Removed' and 'Right Wheel-Unresponsive'}. Because we limit our observations to the position and orientation of the robot in plane, full diagnosis of all observable gripper states was impossible. This was treated as a non-issue as the later addition of a single system variable can produce additional gripper state information (e.g. distance between gripper arms). As such, the classes of

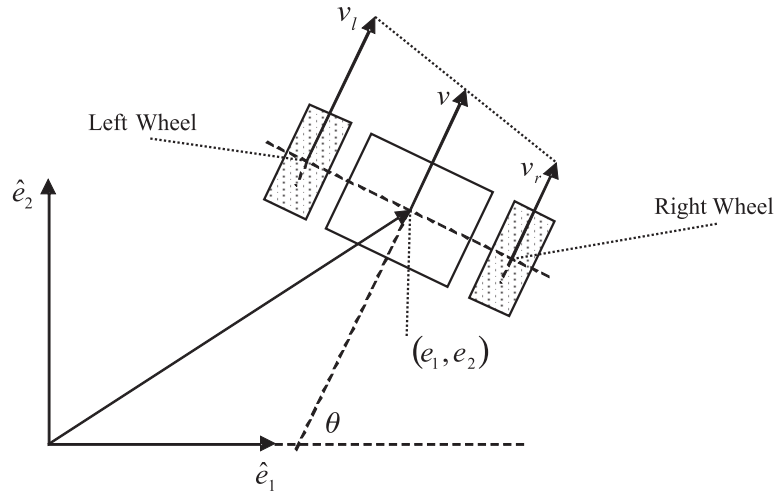


Fig. 6. Two-wheel independent-drive robot with position  $(e_1, e_2)$ , orientation  $\theta$  and velocities shown.

system states that we consider lump all gripper states by their effect on the robot's movement (i.e. only the nominal state and removed states are considered). A list of our system states is available in Appendix 2.

We define the state equation of our testbed using equations similar to Yamamoto et al. (1998) (Figure 6):

$$\begin{aligned}
 \dot{e}_1 &= v \cos(\theta) \\
 \dot{e}_2 &= v \sin(\theta) \\
 \dot{\theta} &= \dot{\phi} \\
 \dot{v}_r &= u_r \\
 \dot{v}_l &= u_l \\
 \dot{v} &= f_i(u_r, u_l) \\
 \ddot{\phi} &= g_i(u_r, u_l)
 \end{aligned} \tag{1}$$

where  $f_i$  and  $g_i$  represent the general state-dependent dynamics with  $u_r$  and  $u_l$  (right and left wheel accelerations) dependent on the continuous input states. For this system, we define our observations as

$$Y_i = \left( (e_1|_{t_1} - e_1|_{t_0}) \quad (e_2|_{t_1} - e_2|_{t_0}) \quad (\theta|_{t_1} - \theta|_{t_0}) \quad \dots \quad (\theta|_{t_f} - \theta|_{t_0}) \right)^T$$

where

$$e_1 = \int_{t_0}^{t_f} \dot{e}_1 \quad \text{and} \quad e_2 = \int_{t_0}^{t_f} \dot{e}_2$$

describe the absolute position of the robot relative to a fixed coordinate in space, and

$$\theta = \int_{t_0}^{t_f} \dot{\theta}$$

describes the rotation about the orthogonal  $(\hat{e}_1 \times \hat{e}_2)$  taken as an absolute measure of the angular displacement of the robot's heading relative to the positive  $e_1$  axis  $(\hat{e}_1)$ .

### 3.2. Diagnostic Maneuver

Given a global system model, the purpose of a *diagnostic maneuver* is to create a set of motions or procedures that will promote an accurate prediction of an unknown system state using observations taken during execution. That is, a diagnostic maneuver is ideally chosen to ensure unique observations  $(Y_1 \neq Y_2 \neq \dots \neq Y_n)$  for every system state  $(\xi_1, \xi_2, \dots, \xi_n)$  in the absence of noise.

We propose a method of deriving a prescribed set of inputs for diagnostic maneuver given a complex system. Following the Design of Experiment (DOE) techniques presented by Fisher (1990), our proposed algorithm applies a three-step process to determine which input factors are principal and to optimize the system response based on the setting of these principal system inputs. We define a *principal system input* as a continuous system input  $(x_i)$  that has a notable effect on the observations of the system.

1. *Screening Experiment.* In a case where all state-specific models produce monotonic (strictly increasing or decreasing) system responses  $(y_1, y_2, \dots, y_s)$ , define high (+1) and low (-1) values for every element of the system input,  $X$  (for models that are not monotonic, three or more values may need to be defined). Using these



high/low values, create a series of  $2^m$  experimental run iterations ( $X = \begin{pmatrix} x_1 & x_2 & \dots & x_m \end{pmatrix}^T$ ) where all possible high/low combinations are addressed. Applying all run iterations to a single state-specific model will produce a set of observations that can be used to statistically sort the relevance of the system inputs with respect to the observations produced. Running this step through each of the state-specific models or experimentally using a testbed with known states applied will create relevant statistical information for every state of the system. Combining this information, one can expound the statistical relevance of each input factor relative to the output over the entire generalized system model. This can be done using standard statistical analysis techniques available in many statistical software packages (e.g. JMP 6.0, SAS Institute, Cary, NC). Using this information, screening criteria (i.e. a designated statistical relevance above some critical value) can be used to select the set of principal system inputs.

2. *Response Optimization.* The goal of response optimization is to maximize the change in each observation from state-to-state by defining optimal system input functions. This should be done while applying inputs over as brief an interval as possible. A typical response optimization would be the application of a response surface method that includes a quadratic model with cross-product terms and three levels of variations for each input term. This is better described by Myers and Montgomery (1995). Noting that all input and output information is discretized when applied to the actual system, this step requires the consideration of the rates at which information is sampled and commands are executed. Specifically, the shortest interval over which the input is applied should be at least as large as the time taken for the system to respond.
3. *Verification Test.* There exists the possibility that in designing the global system model, one or more of the principal inputs may have been overlooked and therefore may not be included as a factor during the Screening Experiment. The verification test involves performing analysis of variance (ANOVA). Such a factor, with a significant effect on the output of the system, will result in a larger than necessary standard error and F ratio for the ANOVA test. In other words, the outputs resulting from our designated functions applied to principal inputs will not be significantly different for two or more of the system states. At this point the generalized system model must be modified to include the missing principal inputs and the previous two steps must be repeated.

Because of the simplicity of the testbed, we defined our diagnostic maneuver intuitively. Limiting our observations to

$$Y_i = \begin{pmatrix} (e_1|_{t_1} - e_1|_{t_0}) & (e_2|_{t_1} - e_2|_{t_0}) \\ (\theta|_{t_1} - \theta|_{t_0}) & \dots & (\theta|_{t_f} - \theta|_{t_0}) \end{pmatrix}^T$$

(the position and orientation of the robot in a plane as described in Section 3.1), we settled on a diagnostic maneuver consisting of a counterclockwise spin, followed by a clockwise spin of the robot. This limited movement uses only the left and right wheel motor inputs, making them the principal system inputs. We then defined the applied levels of our principal inputs and their durations based on the sample rate and field of view of our overhead camera. The input level was defined as the minimum motor voltage applied to a wheel module in the nominal state that consistently produced observable movement, regardless of the state of the gripper or opposite wheel. The input duration was defined such that three discrete position estimates could be taken for each spin.

### 3.3. Model-based Training

Given a complicated diagnostic maneuver, a method of model-based training was devised where the diagnostic maneuver can be separated into a set of simpler *trajectories*. The purpose of performing this separation is to break a large set of observation data into numerically more manageable subsets. Consider a system with  $n$  distinct system states ( $\xi_1, \xi_2, \dots, \xi_n$ ) with a well-defined set of prescribed inputs  $X$  consisting of a piecewise combination of  $p$  distinct input functions such that:

$$X = \begin{cases} X_1 & t_0 \leq t \leq \tau_1 \\ X_2 & \tau_1 < t \leq \tau_2 \\ \vdots & \vdots \\ X_i & \tau_{i-1} < t \leq \tau_i \\ \vdots & \vdots \\ X_p & \tau_{p-1} < t \leq \tau_p \end{cases} \quad (2)$$

Running one of these  $p$  distinct partial sets of inputs ( $X_i$ ) will produce just part of the diagnostic maneuver which we call a trajectory. If our diagnostic maneuver is optimally defined to meet the metrics discussed in Section 3.2, viewing a trajectory can only result in  $q$  unique observations ( $1 \leq q < n$ ). This overlap in observations divides the set  $\xi$  into  $q$  subsets diagnosable using observations from the trajectory. As a result, we say that  $X_i$  defines an equivalence relation on the set  $\xi$  such that:

$$\begin{aligned} & \{ \xi_1, \xi_2, \dots, \xi_n \} / X_i \\ &= \{ \hat{\xi}_1^i, \hat{\xi}_2^i, \dots, \hat{\xi}_j^i, \dots, \hat{\xi}_q^i \} \end{aligned} \quad (3)$$

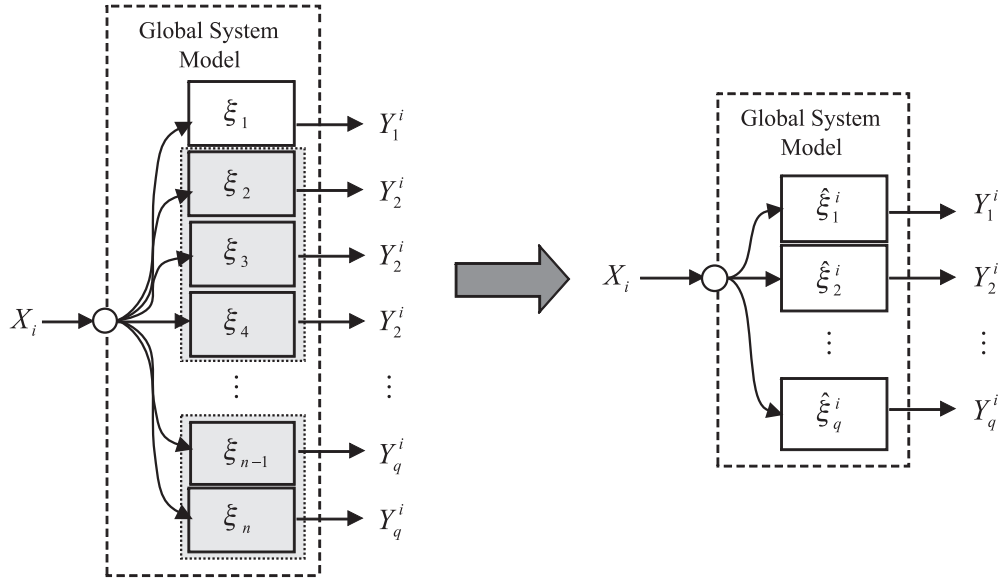


Fig. 7. Global system model where applied  $X_i$  produces a set of unique observations  $(Y_1^i, \dots, Y_q^i)$  where  $1 \leq q < n$ . The classes of system states  $(\hat{\xi}_1^i, \dots, \hat{\xi}_q^i)$  are defined where each is assigned to a unique observation. Each class of system states  $\hat{\xi}_j^i$  is defined such that it contains one or more element from the set of all system states  $\xi$ .

where

$$\bigcup_{j=1}^q \hat{\xi}_j^i = \{ \xi_1, \xi_2, \dots, \xi_n \} \quad \forall \quad i = 1, 2, \dots, p.$$

In other words, the diagnostic command  $X_i$  divides the set of discrete system states into subsets that are equivalent (i.e. indistinguishable) with respect to the incomplete and imperfect observations of the trajectories resulting from  $X_i$ . We denote  $\hat{\xi}_j^i$  as the  $j$ th class of system states induced by  $X_i$ . Each  $X_i$ -induced class is assumed disjoint from one another such that

$$\bigcap_{j,k=1}^q \hat{\xi}_j^i \hat{\xi}_k^i = \{ \}$$

(the empty set) for all  $j \neq k$ . In addition, the classes induced by  $X_i$  are defined in such a way that:

$$\hat{\xi}_{j_1}^1 \cap \hat{\xi}_{j_2}^2 \cap \dots \cap \hat{\xi}_{j_p}^p = \begin{cases} \xi_{j(j_1, j_2, \dots, j_p; 1, 2, \dots, p)} \\ \text{or} \\ \{ \} \end{cases} \quad (4)$$

In other words, whereas individual observations result in classes (subsets) of possible system states that are consistent with the observation of the trajectory resulting from the command  $X_i$ , if the whole diagnostic maneuver is properly defined and properly broken down into individual trajectories then in the end an unambiguous system state should result. The

definition of the  $q$  classes of system states induced by  $X_i$  is represented in Figure 7.

Note that an observation  $(Y_j^i)$  resulting from the  $j$ th class of system states  $(\hat{\xi}_j^i)$  induced by  $X_i$  results in the  $r$ -dimensional vector:

$$Y_j^i = \begin{pmatrix} (y_1|_{\tau_{i-1}+\Delta t} - y_1|_{\tau_{i-1}}) & (y_2|_{\tau_{i-1}+\Delta t} - y_2|_{\tau_{i-1}}) & \dots \\ (y_s|_{\tau_{i-1}+\Delta t} - y_s|_{\tau_{i-1}}) & \dots & (y_s|_{\tau_i} - y_s|_{\tau_{i-1}}) \end{pmatrix}^T$$

where

$$r = S \cdot \frac{\tau_i - \tau_{i-1}}{\Delta t}.$$

For every  $X_i$  ( $i = 1, \dots, p$ ), the set of induced classes  $(\hat{\xi}^i = \{ \hat{\xi}_1^i, \hat{\xi}_2^i, \dots, \hat{\xi}_q^i \})$  is defined and conditional distributions are trained, one for each element of  $\hat{\xi}^i$ . For each  $\hat{\xi}_j^i$ , a multi-variate Gaussian distribution is trained to estimate the probability density that an observation  $(Y_k^i)$  has resulted from a given state  $(\hat{\xi}_j^i)$  in terms of a  $\hat{\xi}_j^i$ -specific mean  $(\mu_j^i)$  and covariance  $(\Sigma_j^i)$ . The distribution for  $\hat{\xi}_j^i$  is given as:

$$p(Y_k^i; \hat{\xi}_j^i) = \frac{1}{(2\pi)^{r/2} |\Sigma_j^i|^{1/2}} \times \exp \left( -\frac{1}{2} (Y_k^i - \mu_j^i)^T (\Sigma_j^i)^{-1} (Y_k^i - \mu_j^i) \right). \quad (5)$$

where  $Y_k^i, \mu_j^i \in \mathbb{R}^r$ , and  $\Sigma_j^i \in \mathbb{R}^{r \times r}$  is symmetric and positive definite.

Note that the mean and covariance ( $\mu_j^i, \underline{\Sigma}_j^i$  respectively) are dependent upon  $\hat{\xi}_j^i$ . Here,  $Y_k^i$  is defined as the observations taken from the trajectory produced from the execution of the partial diagnostic inputs  $X_i$ . The probability density in Equation (5) should be read as ‘the probability density that a particular observation will occur if the robot state is in class  $\hat{\xi}_j^i$ ’.

Training of the diagnostic algorithm is based on the ideal that the probability density function (PDF) of the  $X_i$  induced class of system states  $\hat{\xi}_k^i$  when evaluated at an observation  $Y_k^i$  (attributed to the system with  $\hat{\xi}_k^i$  applied) will be larger than the PDF of the  $X_i$  induced class of system states  $\hat{\xi}_j^i$  when evaluated at the observation  $Y_k^i$ . That is, given an observation ( $Y_k^i$ ) from a known class of system states ( $\hat{\xi}_k^i$ ),

$$\max_{\hat{\xi}_j^i \in \hat{\xi}_1^i} \left( p \left( Y_k^i; \hat{\xi}_j^i \right) \right) = p \left( Y_k^i; \hat{\xi}_k^i \right)$$

where  $Y_k^i$  is the observation associated with the known class of system states  $\hat{\xi}_k^i$  when the partial diagnostic inputs  $X_i$  are applied.

Noting the dependence on  $\mu$  and  $\underline{\Sigma}$ , the training algorithm is defined as follows.

1. Assign initial guess to  $\mu_j^i$  and  $\underline{\Sigma}_j^i$  for all  $\hat{\xi}_j^i \in \hat{\xi}^i$  where  $\mu_j^i$  and  $\underline{\Sigma}_j^i$  must fit the criteria stated above.
2. For each state  $\hat{\xi}_k^i \in \hat{\xi}^i$ ,
  - (a) generate an observation  $Y_k^i$  using the system model associated with  $\hat{\xi}_k^i$  or experimental data from the system with  $\hat{\xi}_k^i$  applied;
  - (b) calculate  $p \left( Y_k^i; \hat{\xi}_j^i \right)$  for all  $\hat{\xi}_j^i \in \hat{\xi}^i$ ;
  - (c) if  $\max_{\hat{\xi}_j^i \in \hat{\xi}^i} \left( p \left( Y_k^i; \hat{\xi}_j^i \right) \right) = p \left( Y_k^i; \hat{\xi}_k^i \right)$ , do nothing;
  - (d) if  $\max_{\hat{\xi}_j^i \in \hat{\xi}^i} \left( p \left( Y_k^i; \hat{\xi}_j^i \right) \right) = p \left( Y_k^i; \hat{\xi}_h^i \right) \neq p \left( Y_k^i; \hat{\xi}_k^i \right)$ ,
    - (i) increase the probability density  $p \left( Y_k^i; \hat{\xi}_k^i \right)$  altering  $\mu_k^i$  and  $\underline{\Sigma}_k^i$ ;
    - (ii) decrease the probability density  $p \left( Y_k^i; \hat{\xi}_h^i \right)$  altering  $\mu_h^i$  and  $\underline{\Sigma}_h^i$ .

To increase and decrease probability density estimates given the defined probability density function, we utilize gradients with respect to the mean and covariance. Specifically, from the work of Jordan (2003), the gradient with respect to  $\mu_j^i$  is

$$\nabla_{\mu_j^i} p \left( Y_k^i; \hat{\xi}_j^i \right) = \left( ((Y_k^i)^T (\underline{\Sigma}_j^i)^{-1})^T - (\underline{\Sigma}_j^i)^{-1} \mu_j^i \right) \in \mathbb{R}^r \quad (6)$$

and the gradient with respect to  $(\underline{\Sigma}_j^i)^{-1}$  is

$$\begin{aligned} & \nabla_{(\underline{\Sigma}_j^i)^{-1}} p \left( Y_k^i; \hat{\xi}_j^i \right) \\ &= \left( \frac{1}{2} \left( \underline{\Sigma}_j^i - (Y_k^i - \mu_j^i)(Y_k^i - \mu_j^i)^T \right) \right) \in \mathbb{R}^{r \times r}. \end{aligned} \quad (7)$$

Probability density is increased by moving the mean and covariance up their respected gradient (using a positive step) and decreased by moving the mean and covariance down their respected gradient (using a negative step).

This algorithm is then iterated multiple times for every trajectory. For better results, the algorithm can be run with a large set of training data. Note that training data is produced from two possible sources: (1) from the model for each  $\hat{\xi}_j^i$  with bounded random noise added to either  $X_i$  or directly to  $Y_j^i$ , or (2) from experimental observations taken of the system with known states applied. Ideally, as the amount of training data increases, the values of  $\mu_k^i$  and  $\underline{\Sigma}_k^i$  will converge to a point such that

$$\max_{\hat{\xi}_j^i \in \hat{\xi}^i} \left( p \left( Y_k^i; \hat{\xi}_j^i \right) \right) = p \left( Y_k^i; \hat{\xi}_k^i \right)$$

for all possible  $Y_k^i$ .

### 3.4. Fault Diagnosis

We implement a generic particle filtering algorithm (van der Merwe et al. 2000) to develop fault diagnosis in multi-robot systems. For every trajectory, a fixed number of particles ( $N$ ) is initially distributed evenly among every class of system states with  $N_j$  designated as the number of particles found in  $\hat{\xi}_j^i$ . Each particle,  $\zeta_k$  ( $k = 1, \dots, N$ ), has an associated weight  $w_k$  that is initialized to  $1/N$  such that the total weight of all particles is normalized to

$$\sum_{k=1}^N w_k = 1.$$

Given an observation  $Y_h^i$  the weight of  $\zeta_k$  is updated based on the current class of system states ( $\hat{\xi}_j^i$ ) that the particle is tied to. Specifically,  $w_k = w_k \cdot p \left( Y_h^i; \hat{\xi}_j^i \right)$ . These weights are then renormalized such that

$$w_k = \frac{w_k}{\sum_{k=1}^N w_k}.$$

The set of particles is then resampled according to the total weight of particles associated with each  $\hat{\xi}_j^i$ . That is, the  $N$  particles are redrawn with probability proportional to the sum of normalized weights associated with each  $\hat{\xi}_j^i$ .

To eliminate sample impoverishment Dearden and Clancey (2001), an equal number of particles is fixed in each class of

system states such that  $\inf_{\hat{\xi}_j^i \in \hat{\xi}^i} (N_j) = N_j^F$  and  $N_h^F = N_j^F$  for all

$\hat{\xi}_h^i, \hat{\xi}_j^i \in \hat{\xi}^i$ . The remaining number of particles ( $N_j^D \equiv N_j - N_j^F$ ) are dynamic in that they are free to move between classes of system states. Accounting for fixed and dynamic particles,  $N^D$  particles are randomly redrawn from the set of  $N$  particles (rather than all  $N$  particles as described above). This leaves  $N^F$  particles fixed to their original class of system states. Note that by using this method we remove the possibility of any one class of system states reaching a zero probability.

The above method is then run for every available set of observations taken of a given trajectory. Once finished, we estimate the probability of a given class of system states  $\hat{\xi}_j^i$  based on the sum of the normalized particle weights tied to  $\hat{\xi}_j^i$ . We denote this discrete probability as

$$\tilde{p}(\hat{\xi}_j^i; \bar{Y}_j^i) = \frac{N_j^F + N_j^D}{N},$$

where  $\bar{Y}_j^i$  is the set of all observations used to calculate the probability. Note the change of probabilistic variable from the probability density function  $p(Y_j^i; \hat{\xi}_j^i)$  defined on  $Y_j^i$  to the probability distribution  $\tilde{p}(\hat{\xi}_j^i; \bar{Y}_j^i)$  defined on  $\hat{\xi}_j^i$ . Given a sufficient number of observations from the same trajectory, all dynamic particles should migrate to the most likely class of system states,  $\hat{\xi}_j^i$ . That is, the ideal scenario yields

$$\tilde{p}(\hat{\xi}_j^i; \bar{Y}_j^i) = \frac{N_j^F + N^D}{N}$$

and

$$\tilde{p}(\hat{\xi}_h^i; \bar{Y}_j^i) = \frac{N_h^F}{N}$$

for all  $h \neq j$ .

Once a probability is calculated for every class of system states for each trajectory in the diagnostic maneuver, we estimate the probability of each system state using the intersecting property of the classes of system states presented in Equation (4). Specifically, the intersection of  $X_i$  trajectory-induced classes over  $i = 1, 2, \dots, p$  will either yield a unique system state or the empty set. From this, we define the probability of being in any one system state as a joint probability  $\tilde{p}(\xi_j; \bar{Y}_{j_1}^1 \& \bar{Y}_{j_2}^2 \& \dots \bar{Y}_{j_p}^p)$  where  $\hat{\xi}_{j_1}^1 \cap \hat{\xi}_{j_2}^2 \cap \dots \cap \hat{\xi}_{j_p}^p = \xi_j$ .

We relate  $\tilde{p}(\xi_j; \bar{Y}_{j_i}^i) = \tilde{p}(\hat{\xi}_{j_i}^i; \bar{Y}_{j_i}^i)$  noting that the observations in the set  $\bar{Y}_{j_i}^i$  are dependent solely on the class of system states  $\hat{\xi}_{j_i}^i$  where  $\xi_j \in \hat{\xi}_{j_i}^i$ . Further consideration yields the joint probability:

$$\begin{aligned} \tilde{p}(\xi_j; \bar{Y}_{j_1}^1 \& \bar{Y}_{j_2}^2 \& \dots \bar{Y}_{j_p}^p) &= \prod_{i=1}^p p(\xi_j; Y_{j_i}^i) \\ &= \prod_{i=1}^p p(\hat{\xi}_{j_i}^i; Y_{j_i}^i) \end{aligned} \quad (8)$$

where  $\hat{\xi}_{j_1}^1 \cap \hat{\xi}_{j_2}^2 \cap \dots \cap \hat{\xi}_{j_p}^p = \xi_j$ .

Using this equation, final diagnosis is defined by finding the system state associated with maximum joint probability. Explicitly:

$$\begin{aligned} &\arg \max_{\xi_k \in \hat{\xi}} \left( \tilde{p}(\xi_k; \bar{Y}_{j_1}^1 \& \bar{Y}_{j_2}^2 \& \dots \bar{Y}_{j_p}^p) \right) \\ &= \arg \max_{\substack{\xi_k \in \hat{\xi} \\ \hat{\xi}_{k_1}^1 \cap \hat{\xi}_{k_2}^2 \cap \dots \cap \hat{\xi}_{k_p}^p = \xi_k}} \left( \tilde{p}(\hat{\xi}_{k_1}^1; \bar{Y}_{j_1}^1) \cdot \tilde{p}(\hat{\xi}_{k_2}^2; \bar{Y}_{j_2}^2) \right. \\ &\quad \cdot \dots \tilde{p}(\hat{\xi}_{k_p}^p; \bar{Y}_{j_p}^p) \left. \right). \end{aligned}$$

## 4. Testing Procedure

To evaluate our methods, we consider a set of thirty observations taken of each trajectory within the diagnostic maneuver with each of the system states applied. Of these thirty observations per trajectory per state, we allocate half (fifteen per trajectory per state) for training the diagnostic algorithm, and half to evaluate the diagnostic algorithm. Using this dataset, we train our algorithm using five, ten, and fifteen observations per trajectory per state. We then evaluate trained algorithms and compare the results to the ideal case.

We calculate  $\mu$  and  $\underline{\mu}$  for each class of system states, using the 1000 iterations of the training algorithm presented in Section 3.3. To evaluate training, we use the following two metrics.

1. *Number of Correct States.* The number of classes of states correctly predicted using the trained conditional distribution after every iteration of the algorithm. We define a class of states as correctly predicted if

$$\max_{\hat{\xi}_j^i \in \hat{\xi}^i} \left( p(Y_k^i; \hat{\xi}_j^i) \right) = p(Y_k^i; \hat{\xi}_k^i).$$

2. *Continuous Error Estimate.* The difference between the probability density of the applied class of system states ( $\hat{\xi}_k^i$ ) evaluated at  $Y_k^i$  and probability density of the most probable class of system states ( $\hat{\xi}_j^i$ ) evaluated at  $Y_k^i$  summed through every iteration is calculated, i.e.

$$\sum_{l=1}^{1000} \sum_{k=1}^q p(Y_k^i; \hat{\xi}_k^i) - p(Y_k^i; \hat{\xi}_j^i)$$

where  $q$  is the total number of classes of system states associated with the applied trajectory).

Ideally, we expect the number of correct states to converge to the total number of classes associated with each trajectory, and the continuous error estimate to converge to zero.

We first evaluate the trained diagnostic algorithms by diagnosing each known system state using the observations from training (i.e. five observations per trajectory per state for the five-sample training, ten observations per trajectory per state for the ten-sample training, etc.). The results are presented using a surface plot of the calculated probability of each system state versus the known state of the observations used in the diagnosis. Note that the ideal case will produce a consistent peak along the diagonal indicating that the predicted system state matches the applied or observed system state. Specifically, we used five fixed and twenty dynamic particles per class of system states associated with each trajectory. As a result, the highest possible probability of any class for each of our trajectories is therefore  $(20 \times 25)/(25 \times 25) = 0.8$ . Calculating the joint probability using Equation (8) yields a maximum possible probability of any state equal to 0.64. Similarly, the minimum possible probability of any state is equal to  $[5/(25 \times 25)]^2 = 6.4 \times 10^{-5}$ .

We then compare the performance of the five-sample, ten-sample and fifteen-sample trained algorithms, diagnosing using a single observation. To do so, we use the ratio of the total number of correctly predicted states to the total number of incorrectly predicted states. We define a correctly predicted state using the following criteria: (1) the state with the maximum probability matches the state of the applied observations, and (2) the maximum probability is at least 1.05 times the second highest probability. We define an incorrectly predicted state using similar criteria: (1) the state with the maximum probability does not match the state of the applied observations, and (2) the maximum probability is at least 1.05 times the second highest probability. All diagnoses not matching the criteria for either a correct or incorrect diagnosis are considered inconclusive. Each diagnosis is performed using a single observation. We calculate the total number of correct and incorrect diagnoses by summing every diagnosis applied over the entire set of observations allocated for evaluation.

Finally, we evaluate the ten-sample trained diagnostic algorithm performance using multiple observations. Specifically, we compare the ratio produced using a single observation (described above) to the performance using five and ten observations. To make our data comparable to the single observation testing, we use fifteen combinations of five observations per trajectory per state, and fifteen combinations of ten observations per trajectory per state from the evaluation data. We present the results using the same ratios described above.

To evaluate the global system model, we first trained a diagnostic algorithm using 100 observations per trajectory per state produced from the global system model. Over the 100

observations, reasonable noise was added to the model in an attempt to match the noise experimental. Once trained, we repeated the evaluation procedure from the ten-sample trained diagnostic algorithm performance using multiple observations.

## 5. Results

The metrics from five-sample training are presented in Figures 8 and 9. We observed an apparent exponential convergence of both metrics and subsequent plateau in training for both the clockwise and counterclockwise trajectories. The maximum number of correct states calculated for the clockwise trajectory was 19 out of the 25 classes of clockwise system states (Figure 8a). For the counterclockwise trajectory, the maximum number of correct states was 23 out of the 25 classes (Figure 9b). After 1000 iterations, the continuous error estimates were  $-5.72 \times 10^{-4}$  for the clockwise trajectory (Figure 8b) and  $-3.47 \times 10^{-4}$  for the counterclockwise trajectory (Figure 9b).

Figure 10 shows results of diagnoses performed using the five training observations per trajectory per state. This particular test shows 19 out of the 43 system states correctly predicted with an associated probability of  $> 1.6$  times the second most likely state. Of the 43 system states, 10 are effectively indistinguishable from one another with an associated probability of  $< 1.05$  times the second most likely state.

Applying the same procedures to ten-sample training shows notable improvement. The maximum number of correct states calculated for the clockwise trajectory was 25 out of 25 classes of clockwise system states (Figure 11a). For the counterclockwise trajectory, the maximum number of correct states was 23 out of 25 classes (Figure 12a). After 1000 iterations, the continuous error estimates were  $-1.58 \times 10^{-2}$  for the clockwise trajectory (Figure 11b) and  $-7.00 \times 10^{-3}$  for the counterclockwise trajectory (Figure 12b).

Figure 13 shows results of diagnoses performed using the ten training observations per trajectory per state. This test shows a far more consistent peak as 29 out of 43 system states are correctly predicted with an associated probability of  $> 1.6$  times the second most likely system state. Further, this test shows 34 out of the 43 system states correctly predicted with an associated probability of  $> 1.5$  times the second most likely system state. Finally, none of the 43 system states are indistinguishable, with an associated probability of  $< 1.05$  times the second most likely state.

For fifteen-sample training, the maximum number of correct states calculated for the clockwise trajectory was 25 out of the 25 classes of the clockwise system states. For the counterclockwise trajectory, the maximum number of correct states was 24 out of the 25 classes. After 1000 iterations, the continuous error estimates were  $-6.74 \times 10^{-2}$  for the clockwise trajectory and  $-1.03 \times 10^{-2}$  for the counterclockwise trajectory.



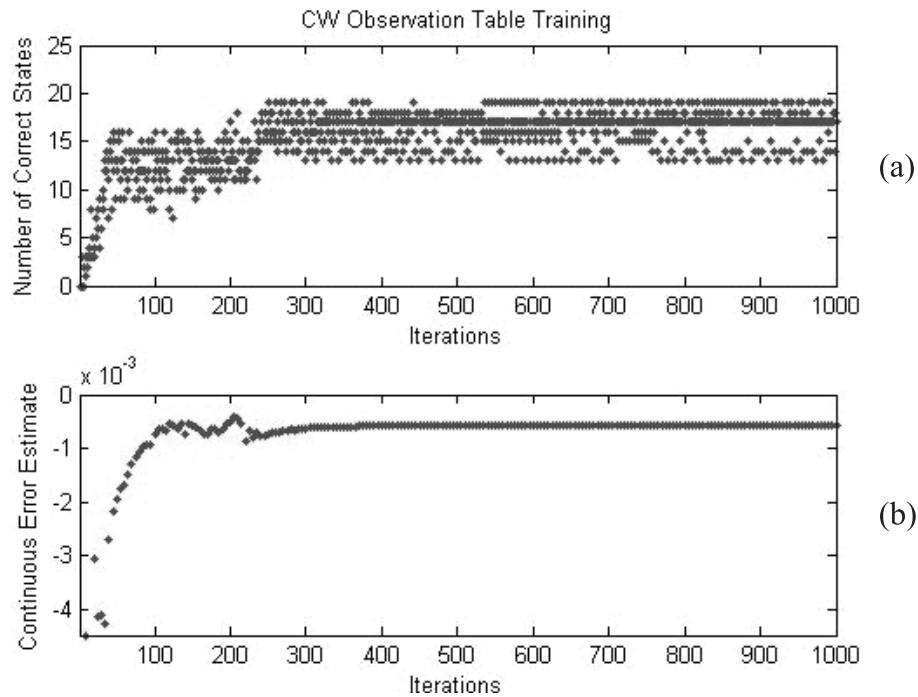


Fig. 8. Training metrics for five-sample training of the clockwise trajectory. (a) *Number of Correct States* metric plotted after every ten iterations of the algorithm and (b) *Continuous Error Estimate* metric plotted after every ten iterations of the algorithm.

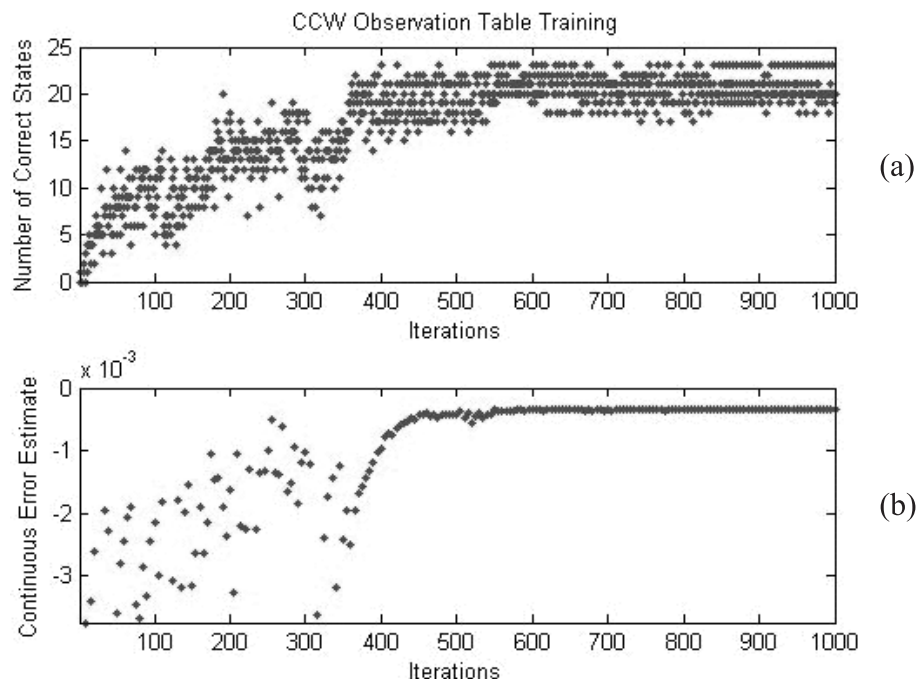


Fig. 9. Training metrics for five-sample training of the counterclockwise subroutine. (a) *Number of Correct States* metric plotted after every ten iterations of the algorithm and (b) *Continuous Error Estimate* metric plotted after every ten iterations of the algorithm.

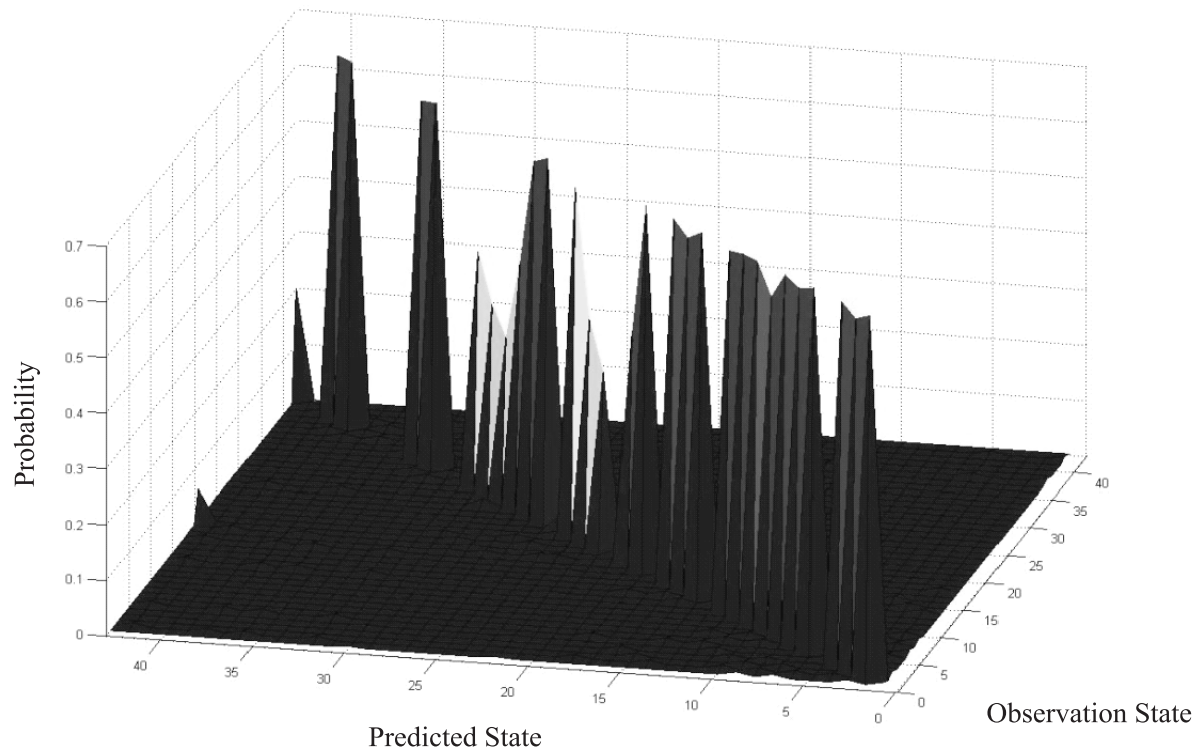


Fig. 10. Surface plot of predicted observation states probability using the five training observations per trajectory versus the known observation state.

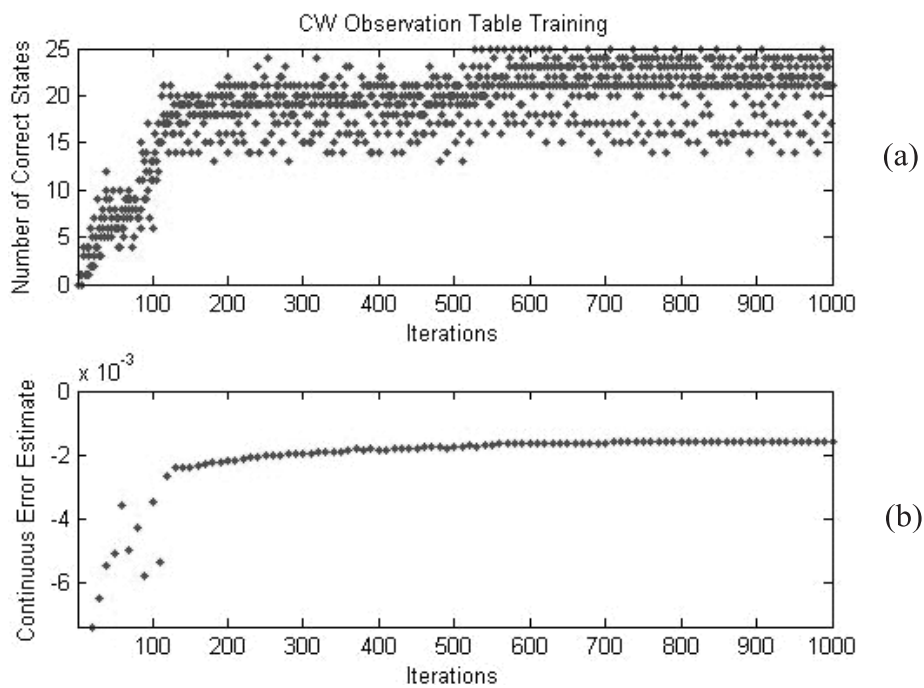


Fig. 11. Training metrics for ten-sample training of the clockwise trajectory. (a) *Number of Correct States* metric plotted after every ten iterations of the algorithm and (b) *Continuous Error Estimate* metric plotted after every 10 iterations of the algorithm.

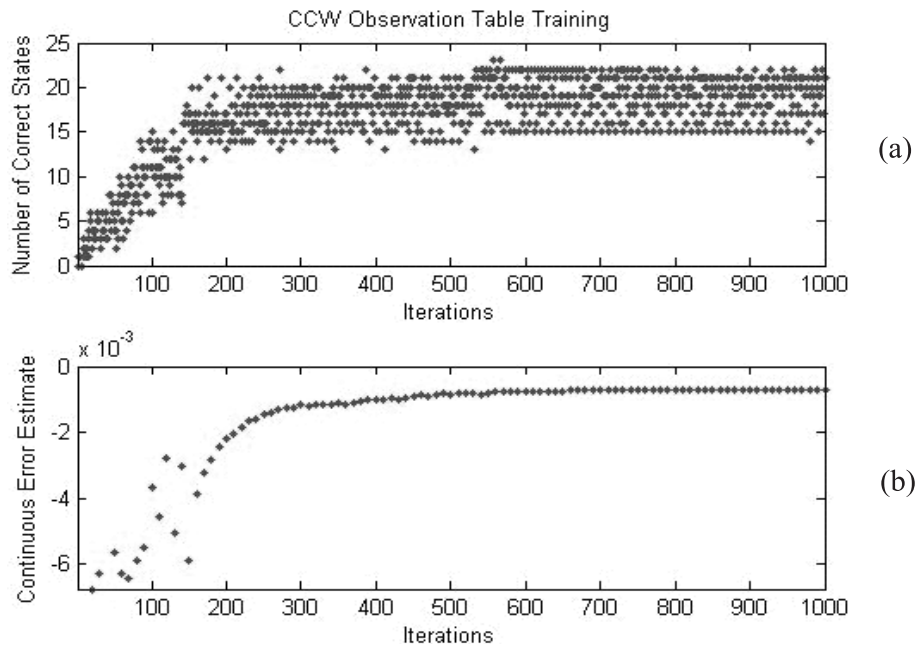


Fig. 12. Training metrics for ten-sample training of the counterclockwise subroutine. (a) *Number of Correct States* metric plotted after every ten iterations of the algorithm and (b) *Continuous Error Estimate* metric plotted after every ten iterations of the algorithm.

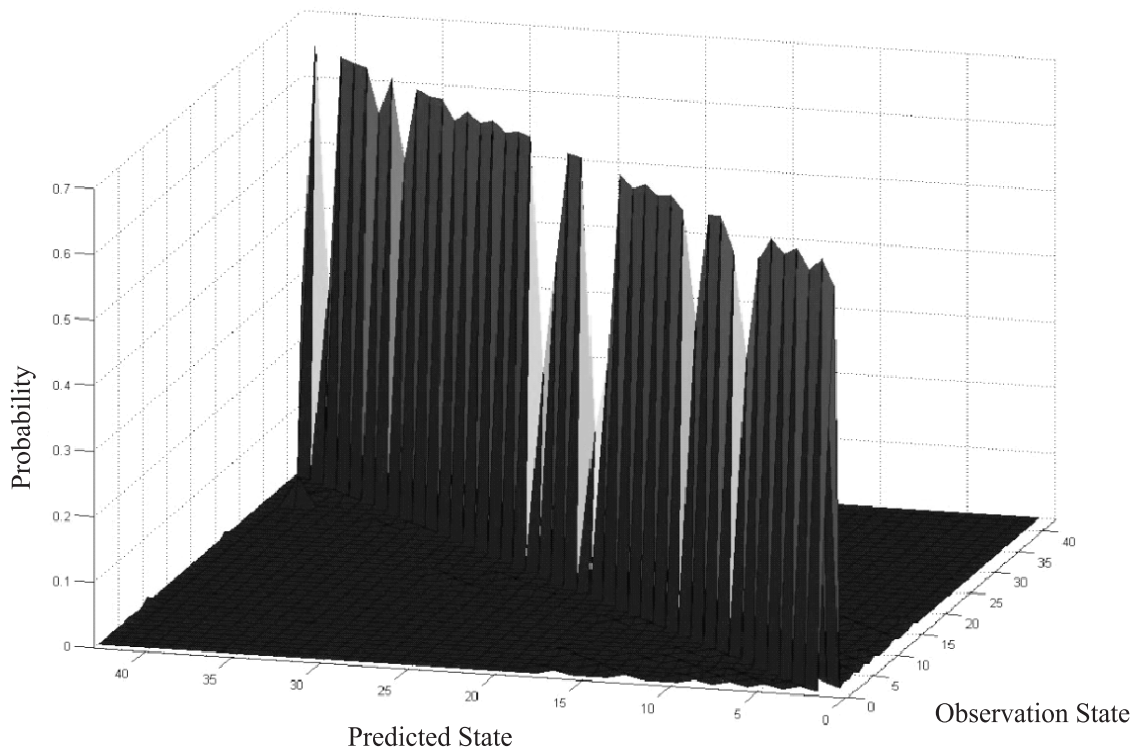


Fig. 13. Surface plot of predicted observation state probability using the ten training observations per trajectory versus the known observation state.

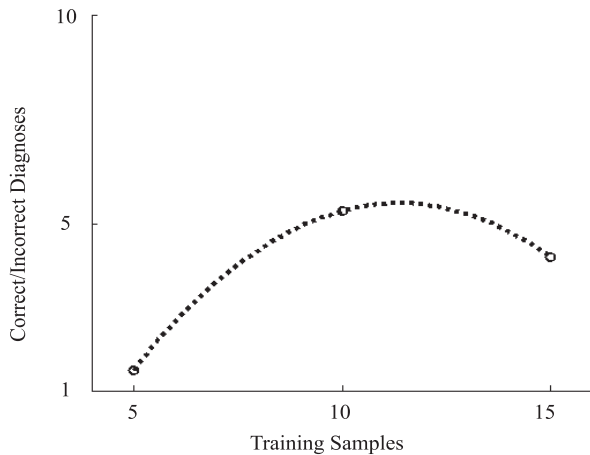


Fig. 14. The ratios of correct diagnoses to incorrect diagnoses for five-sample, ten-sample and fifteen-sample training.

The results of diagnoses performed using the fifteen training observations from each state were 31 out of 43 system states correctly predicted with an associated probability of  $>1.6$  times the second most likely system state. Further, this test produced 35 out of the 43 system states correctly predicted with an associated probability of  $>1.5$  times the second most likely system state. Finally, none of the 43 system states are indistinguishable with an associated probability of  $<1.05$  times the second most likely state.

The ratios of correct diagnoses to incorrect diagnoses for five-sample, ten-sample and fifteen-sample trained algorithms are shown in Figure 14. We note an initial increase from the five-sample ratio (approximately 1.5) to the ten-sample ratio (approximately 5.3), followed by decreased ratio (approximately 4.2) for fifteen-sample training.

The results from the ten-sample trained algorithm performing diagnosis using one observation per trajectory per state, five observations per trajectory per state and ten observations per trajectory per state are shown in Figure 15. Here, we note an initial increase from the single observation ratio (approximately 5.3) to the five observation ratio (approximately 9.2). This is then followed by an apparent plateau with the ten-observation ratio (approximately 8.8).

For 100-sample model-based training, the maximum number of correct states calculated for the clockwise trajectory was 25 out of the 25 classes of the clockwise system states. For the counterclockwise trajectory, the maximum number of correct states was 25 out of the 25 classes. After 1000 iterations, the continuous error estimates were  $-1.35 \times 10^{-1}$  for the clockwise trajectory and  $-9.74 \times 10^{-2}$  for the counterclockwise trajectory.

The results of diagnoses performed using the 100 model-based training observations from each state were 43 out of 43 system states correctly predicted with an associated probability of  $>1.6$  times the second most likely system state.

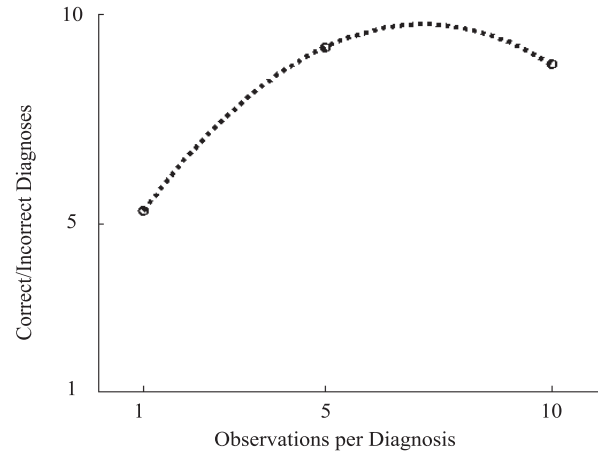


Fig. 15. The ratios of correct diagnoses to incorrect diagnoses using the ten-sample trained algorithm run using a single observation, five observations and ten observations.

The results from the algorithm trained using 100 model observations per trajectory per state performing diagnosis using one experimental observation per trajectory per state, five experimental observations per trajectory per state, and ten experimental observations per trajectory per state are as follows. Using a single experimental observation, the ratio was approximately 1.3. For the five experimental observations, the ratio was approximately 1.2. Finally, for the ten-experimental observation, ratio was approximately 1.3.

## 6. Discussion and Future Research

We have demonstrated a generalized approach for cooperative team-diagnosis in multi-robot systems. This approach is not limited by the number of system inputs nor is it limited by the number of system states.

Our experimental results, while promising, provide additional challenges to training and diagnosing the system. As noted in Section 5, we were unable to produce training capable of predicting all 43 designated system states. Rather, both increasing the experimental samples used for training and increasing the number of experimental observations used for diagnosis showed apparent plateaus in the ratio of the total correct diagnoses to the total incorrect diagnoses. Training using observations from the dynamic model produced a diagnostic algorithm that was capable of correctly diagnosing at least part of the experimental observations. However, the results were significantly worse than those produced by algorithms trained using experimental data.

The possible cause of the noted plateaus is the overlap of observations ( $Y_i \approx Y_j$ ) associated with specific sets of system states. These overlaps may also have caused fluctuations seen while training the diagnostic algorithms (Figures 8a, 9a, 11a and 12a). We attribute these overlaps to sub-optimal data

**Table 1. Example of system states that may produce an overlap in observations**

State 33	State 39
CPU – Nominal	CPU – Nominal
Left wheel – Stuck clockwise	Left wheel – Stuck clockwise
Right wheel – Removed	Right wheel – Stuck
Gripper – Nominal	Gripper – Nominal

collection. Specifically, the low observation sampling rate coupled with un-modeled dynamics that resulted in an overly conservative diagnostic routine can explain these results. We assume the majority of un-modeled dynamics are directly related to the twisting, bending and dragging forces of robot's power and communication tether. Admittedly, this was ignored in the derivation of the dynamic model as we assumed its contribution would be negligible.

As an example of two states that produce overlapping observations, consider states 33 and 39 shown in table 1.

Experimental results indicate that system state 39 shows the expected decrease in the number of incorrect diagnoses when we increase from five-sample to ten-sample training. Out of the 15 diagnoses, we see 8 incorrect diagnoses associated with five-sample training and 4 incorrect diagnoses associated with ten-sample training. The result of increasing the number of observations used for a given diagnosis, however, produces a sharp increase in the number of incorrect diagnoses. With ten-sample training, we see 7 incorrect diagnoses from the 15 diagnoses where each uses 5 observations. Likewise, of the 15 diagnoses where each uses 10 observations, we see 10 incorrect diagnoses. Consistently, each of these incorrect diagnoses predicts system state 33. This erroneous recurrence results in a high probability associated with state 33 when we increase the number of observations used in a diagnosis. In the case where 10 observations are used to diagnose observations from state 39, the estimated probability associated with each incorrect diagnosis (probability of state 33) becomes consistently  $>0.60$  while the probability associated with the applied state (probability of state 39) is consistently  $<0.10$ .

When applying system states 33 and 39 to the dynamic model, the discrepancy between states produces observations that are clearly different even with substantial noise applied to the motor inputs. Applying these states to the testbed, however, produces many observations that are impossible to differentiate. As mentioned previously, we attribute this undesirable behavior to the un-modeled forces associated with the robot's power and communication tether. We further note that consistent overlapping of testbed observations with specific states applied occurs regularly between all states with motion similar to a state containing a missing wheel module. The effect of tether interaction may be further amplified by the application of a vision system with a very low sampling frequency.

Note that the low sampling frequency was intentionally used to increase the diagnosis challenge.

Clearly, the possibility of system states being masked by other system states is a potential issue with this method of diagnosis. Specifically, this problem arose because of what appears to be a suboptimal diagnostic maneuver. With that said, the solution to this issue has already been presented in Section 3.2 in that, by definition, an optimal diagnostic maneuver should effectively eliminate this large overlap between the observations associated with system states. Because our diagnostic maneuver was intuitively defined based on an understanding derived from the global system model, we effectively overlooked some key contributors to the robot's behavior.

As mentioned, we attribute the majority of these overlapping observations to un-modeled dynamics resulting from the robot's power and communication tether. Had we used experimental data to derive the diagnostic maneuver in lieu of the global system model, this masking phenomenon may never have arisen. To further reduce the risk of overlapping observations, a longer diagnostic maneuver can be used. That is, if we relax the 'applied over as short an interval as possible' metric described in Section 3.2, we will effectively make longer observations that will convey more than the minimum amount of information necessary to make a diagnosis. By doing so, we will reduce the possible overlap between observations, but increase the time and energy required to perform a diagnosis.

While our experimental results focus on what can effectively be considered a two-robot system, we specifically considered diagnosis using multiple observations as an initial proof of concept for diagnosis using multiple robots simultaneously. Specifically, we assume that five robots capable of observing a diagnostic maneuver and submitting their observations to one another is equivalent to a single robot taking an observation of five diagnostic maneuvers. The main assumption is that the source of observations differs in a way that will add information to the diagnosis. That is, each robot sees things in a slightly different way (from a different angle or distance).

We have demonstrated an approach for team-diagnosis that is not limited by the number of system inputs and/or system states. To further develop this generalized approach, we suggest that the following topics be addressed.

1. *Unknown System States.* The addition of one or more unknown or unassigned system state(s) to the diagnostic algorithm may improve the applicability of this method. That is, this method will become far more applicable to complex systems if the diagnostic algorithm is improved to account for one or more system states that may have been overlooked during development.
2. *Robust Strategies for Separating Indistinguishable States.* This may require the use of multiple, heterogeneous sensors on each robot, or distributed amongst members of a team of robots. Specifically, a set of selection criteria for the types of sensors and observa-



tions that will provide useful information to distinguish between indistinguishable states will increase the presented methods utility.

3. *Environment Compensation*. Methods of accounting for various changes in the environment where diagnosis is performed will make the presented method far more robust. Methods of accounting for environmental changes may include training using data from multiple environments and estimating behavioral changes relative to environmental information.
4. *Diagnoses from Faulty Robots*. For cooperative team-diagnosis to become a useful technique, methods of accounting for undetected faults should be developed. Specifically, the possibility exists that one or more of the diagnosing robots could contain a fault. In this case, the diagnoses contributed by these robots cannot necessarily be trusted. Accounting for this may require techniques for detecting the outliers attributed to diagnoses submitted by unreliable robots.
5. *Combining Diagnoses*. Given a team of robots, the issue of combining diagnoses from various robots needs to be addressed. Consider the scenario described above where one or more robots performing diagnosis is unreliable. In this case, how can the available diagnoses be compiled, and where will the compiling occur? Compensating for this issue may require techniques involving distributed decision-making or SWARM techniques.

## 7. Towards Team Repair

Previously, we defined team repair as the act of determining the correct course of actions based on an identified problem and completing the appropriate action to solve the problem. Applying this definition to a team of modular robots, team repair becomes a coordinated effort between robots to remove and replace modules diagnosed as faulty. To do so, we propose a structured plan for performing repair that is defined by a compiled set of prescribed module-specific procedures corresponding to each system state. Once all module-specific procedures are completed, the team will consider the ailing robot repaired. We consider a rather simple repair paradigm applied to our testbed.

1. Robots cooperatively diagnose the faulty module(s) of the *Ailing Robot*.
2. Each robot plans and reports a shortest path to the faulty module(s) of the *Ailing Robot* and a storage area of replacement parts based on its collective knowledge of the position of the other robots in the working environment.
3. For each faulty module, the robotic team will repeat the following stages.
  - (a) The robot with the shortest path to the storage area (the *Replacement Robot*) will move to collect the required replacement module.
  - (b) The robot with the shortest path to the faulty module (the *Removal Robot*) will move to and remove the faulty module from the ailing robot.
  - (c) The *Removal Robot* will transport the faulty module from the *Ailing Robot* and deposit it in a designated part of the storage area.
  - (d) The *Replacement Robot* will move to the *Ailing Robot* and replace the faulty module.
  - (e) The *Removal Robot* will move and assume the original location of the *Replacement Robot* and the *Replacement Robot* will move and assume the original location of the *Removal Robot*.

The team-repair discussed above is a simple paradigm within a structured environment. In order for robots to act on the information obtained from team-diagnosis, schemes to effectively implement team-repair procedures must be considered. Specifically, both the individual subsystems of the robot and the team-repair protocols must be robust to errors and uncertainty. Clearly, many sources of potential disorder exist. The error associated with sensing, manipulation and environmental uncertainty combine to yield a high potential for disorder. Moreover, the structured team-diagnosis as discussed in the results ignores time-dependent motions of objects/obstacles during the planning and execution of the repair procedure. A more realistic scenario may include moving objects/obstacles as well as robots in the environment during this procedure.

In addition, for optimal planning of the shortest paths, the repair objective function may also consider the task-in-hand for each robot at the time of repair. For example, if one robot does not have an important or time-sensitive task, it may be more efficient for that robot to perform all of the six repair steps (described above) by itself. Moreover, the example discussed here includes homogenous robots with similar modules. In a more general problem, the cooperative team may involve heterogeneous modular robots where robots carry different payloads and perform different functions. As a result, the task of team-repair will become more complicated and the objective function must therefore include metrics for the task being performed, including the importance of the task and the capabilities of each robot.

The extension of team-repair to an unstructured environment may be addressed by applying behavior-based swarm algorithms to the system (Scheidt et al. 2005). The development and implementation of behavior-based swarm algorithms has been reported on numerous hardware platforms including unmanned air, ground and sea-surface vehicles (Hawthorne et al. 2007). These experiments have shown the ability of these algorithms to perform well in scenarios that have intermittent

## Appendix 1

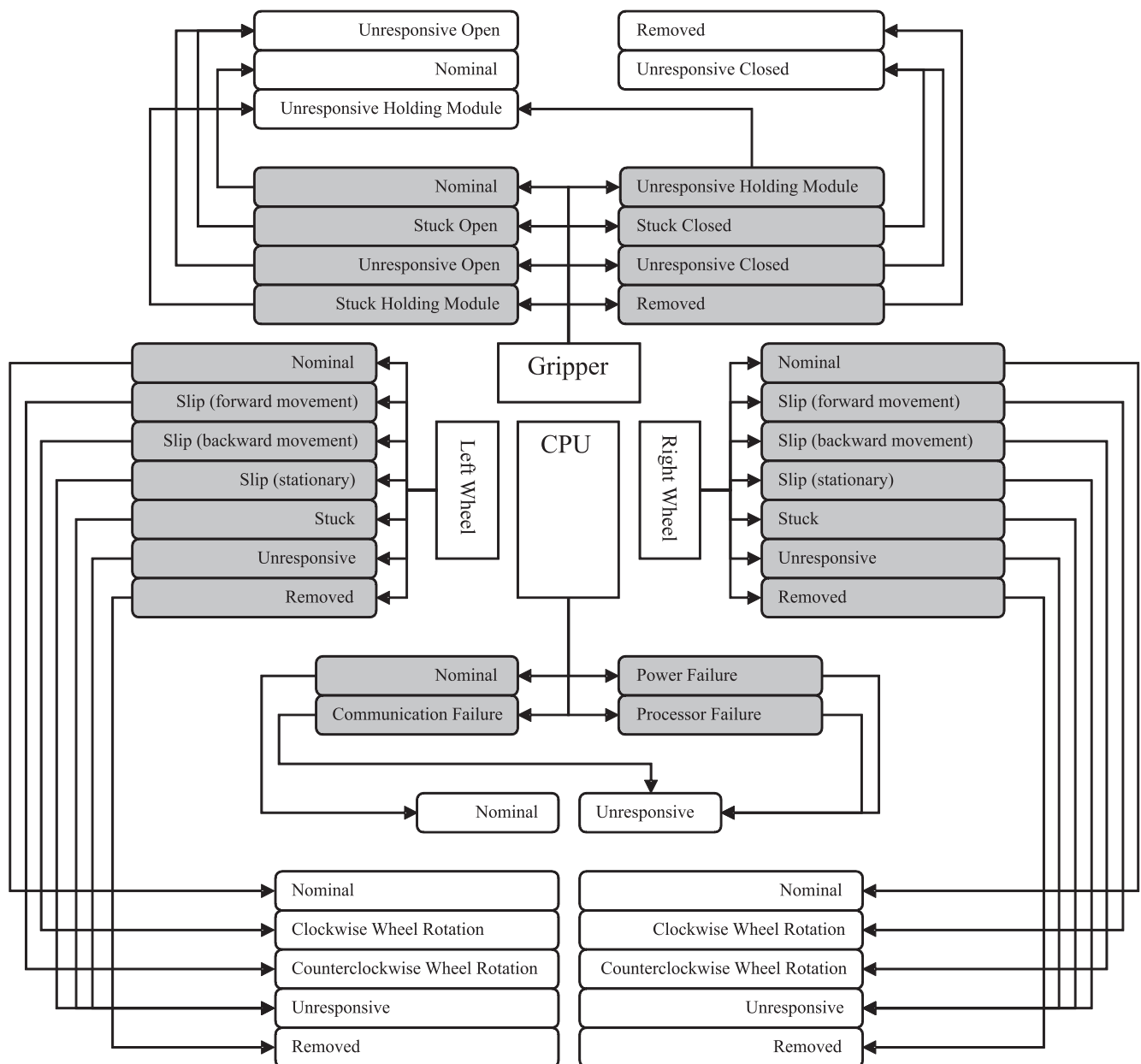


Fig. 16. Observable module states (gray) and diagnosable module states (white).

communications, contain heterogeneous mix of vehicles and sensors and require extended operation without direct human supervision. The theoretical bases of these algorithms are dynamic co-fields, a swarm behavior that uses localized decision processes to generate emergent sophisticated maneuvers (Scheidt et al. 2005, Hawthorne et al. 2007). The future extension of this work will involve the implementation of these behavior-based swarm algorithms to the problem of cooperative team-repair in unstructured environments.

## Acknowledgements

The authors thank the 2006 NASA Robotics Academy interns Elan Hourticolon-Retzler, Brad Smith and Anahita Karimi for their contributions to this work. In addition, the authors thank Kevin Wolfe for his significant feedback regarding the presentation of this work. This work was supported by Internal Research and Development funds provided by the Johns Hopkins University Applied Physics Laboratory.

## Appendix 2

**Table 2. Observable system states**

State	CPU	Left wheel	Right wheel	Gripper
1	Nominal	Nominal	Nominal	Nominal
2	Nominal	Nominal	Nominal	Removed
3	Nominal	Clockwise Rotation	Nominal	Nominal
4	Nominal	Clockwise Rotation	Nominal	Removed
5	Nominal	Counterclockwise Rotation	Nominal	Nominal
6	Nominal	Counterclockwise Rotation	Nominal	Removed
7	Nominal	Nominal	Clockwise Rotation	Nominal
8	Nominal	Nominal	Clockwise Rotation	Removed
9	Nominal	Clockwise Rotation	Clockwise Rotation	Nominal
10	Nominal	Clockwise Rotation	Clockwise Rotation	Removed
11	Nominal	Counterclockwise Rotation	Clockwise Rotation	Nominal
12	Nominal	Counterclockwise Rotation	Clockwise Rotation	Removed
13	Nominal	Nominal	Counterclockwise Rotation	Nominal
14	Nominal	Nominal	Counterclockwise Rotation	Removed
15	Nominal	Clockwise Rotation	Counterclockwise Rotation	Nominal
16	Nominal	Clockwise Rotation	Counterclockwise Rotation	Removed
17	Nominal	Counterclockwise Rotation	Counterclockwise Rotation	Nominal
18	Nominal	Counterclockwise Rotation	Counterclockwise Rotation	Removed
19	Nominal	Removed	Nominal	Nominal
20	Nominal	Removed	Nominal	Removed
21	Nominal	Unresponsive	Nominal	Nominal
22	Nominal	Unresponsive	Nominal	Removed
23	Nominal	Removed	Clockwise Rotation	Nominal
24	Nominal	Removed	Clockwise Rotation	Removed
25	Nominal	Unresponsive	Clockwise Rotation	Nominal
26	Nominal	Unresponsive	Clockwise Rotation	Removed
27	Nominal	Removed	Counterclockwise Rotation	Nominal
28	Nominal	Removed	Counterclockwise Rotation	Removed
29	Nominal	Unresponsive	Counterclockwise Rotation	Nominal
30	Nominal	Unresponsive	Counterclockwise Rotation	Removed
31	Nominal	Nominal	Removed	Nominal
32	Nominal	Nominal	Removed	Removed
33	Nominal	Clockwise Rotation	Removed	Nominal
34	Nominal	Clockwise Rotation	Removed	Removed
35	Nominal	Counterclockwise Rotation	Removed	Nominal
36	Nominal	Counterclockwise Rotation	Removed	Removed
37	Nominal	Nominal	Unresponsive	Nominal
38	Nominal	Nominal	Unresponsive	Removed
39	Nominal	Clockwise Rotation	Unresponsive	Nominal
40	Nominal	Clockwise Rotation	Unresponsive	Removed
41	Nominal	Counterclockwise Rotation	Unresponsive	Nominal
42	Nominal	Counterclockwise Rotation	Unresponsive	Removed
43	Unresponsive	Unresponsive	Unresponsive	Unresponsive

## References

- Bererton, C., and Khosla, P.K. (2000). Toward a team of robots with repair capabilities: A visual docking system. *Proceedings of the ISER 7<sup>th</sup> International Symposium on Experimental Robotics*, pp. 333–342.
- Blackmore, L., Funiak, S., and Williams, B. (2005). Combining stochastic and greedy search in hybrid estimation. *Proceedings of the 20<sup>th</sup> National Conference on Artificial Intelligence*, pp. 282–287.
- Bohringer, K.F. (2003). Surface modification and modulation in microstructures: controlling protein adsorption, monolayer desorption, and micro-self-assembly. *IOP Journal of Micromechanics and Microengineering (JMM)* **13**(4): S1–S10.
- Chirikjian, G.S., and Suthakorn, J. (2002). Toward self-replicating robots. *Proceedings of the 8<sup>th</sup> International Symposium on Experimental Robotics (ISER)*, pp. 392–401.
- Chirikjian, G.S., Pamecha, A. and Ebert-Uphoff, I. (1996). Evaluating efficiency of self-reconfiguration in a class of modular robots. *Journal of Robotic Systems* **13**(5): 317–338.
- Daigle, M., Koutsoukos, X. and Biswas, G. (2006). Distributed diagnosis of coupled mobile robots. In *Proceedings of IEEE International Conference on Robotics and Automation*, pp. 3787–3794.
- Dearden, R., and Clancey, D. (2001). Particle filters for Real-Time Fault Detection in Planetary Rovers. Research Institute for Advanced Computer Science, NASA Ames Research Center.
- Doucet, A., de Freitas, J., Murphy, K., and Russell, S. (2000). Rao-blackwellised particle filtering for dynamic Bayesian networks. *Proceedings of the Conference on Uncertainty in Artificial Intelligence*, pp. 176–183.
- Duan, Z., Cai, Z., and Yu, J. (2005). Fault diagnosis and fault tolerant control for wheeled mobile robots under unknown environments: A survey. *Proceeding of IEEE International Conference on Robotics and Automation*, 3428–3433.
- Fisher, R.A. (1990). *The Design of Experiments*. Oxford University Press: New York.
- Fröhlich, P., de Almeida Móra, I., Nejd, W., and Schroeder, M. (1999). Diagnostic Agents for Distributed Systems. *Selected papers from the ESPRIT Project ModelAge Final Workshop on Formal Models of Agents*, pp. 173–186.
- Gracias, D.H., Choi, I., Weck, M., and Whitesides, G.M. (2001). Meso scale self-assembly. *Algorithmic and Computational Robotics: New Direction* B. Donald, K. Lynch and D. Rus (eds), A.K. Peters, Ltd, pp. 1–9.
- Hawthorne, R.C., Stripes, J.A., Chalmers, R.W., and Scheidt, D.H. (2007). Demonstration of effects-based operations using fully autonomous heterogeneous vehicle swarms. In *Proceedings of the IEEE International Conference on Robotics and Automation*, pp. 3124–3125.
- Henzen, T.A. (2000). The theory of hybrid automata. *Verification of Digital and Hybrid Systems*, M.K. Inan, R.P. Kurshan (eds), NATO ASI Series F: Computer and Systems Sciences **170**: 265–292.
- Jordan, M.I. (2003). *An Introduction to Probabilistic Graphical Model*. Chapter 13, University of California, Berkeley (To be published).
- Kalech, M., Kaminka, G.A., Meisels, A., and Elmaliach, Y. (2006). Diagnosis of multi-robot coordination failures using distributed CSP algorithms. In *Proceedings of the Twenty-First National Conference on Artificial Intelligence (AAAI-06)*, pp. 970–975.
- Klavins, E. (2007). Programmable self-assembly. *Control Systems Magazine* **24**(4): 43–56.
- Mosterman, P., and Biswas, G. (1999). Diagnosis of continuous valued systems in transient operating regions. *IEEE Transactions on Systems, Man and Cybernetics, Part A* **29**(6): 554–565.
- Murata, S., Kurokawa H., and Kokaji S. (1994). Self-assembling machine. *Proceedings of the 1994 IEEE International Conference on Robotics and Automation*, vol. 1; pp. 441–448.
- Murata, S., Yoshida, E., Kurokawa, H., Tomita, K., and Kokaji, S. (2001). Self-repairing mechanical systems. *Autonomous Robots* **10**(1): 7–21.
- Myers, R.H., and Montgomery, D.C. (1995). *Response Surface Methodology: Process and Product Optimization Using Designed Experiments*. John Wiley & Sons: New York.
- Park, W., Albright, D., Addleston, C., Won, W.K., Lee, K., and Chirikjian, G.S. (2004). Robotic self-repair in a semi-structured environment. *Proceedings of Robosphere2004*.
- Roos, N., ten Teije, A., and Witteveen, C. (2003). A protocol for multi-agent diagnosis with spatially distributed knowledge. *Proceedings of the first International Joint Conference on Autonomous Agents and Multiagent Systems* **2**: 655–661.
- Rus, D. (1998). Self-reconfiguring robots. *IEEE Intelligent Systems and Their Applications* **13**(4): 2–4.
- Satish Kumar, T.K. (2001). QCBFS: Leveraging qualitative knowledge in simulation-based diagnosis. *Proceedings of the 15<sup>th</sup> International Workshop on Qualitative Reasoning*.
- Scheidt, D., Neighoff, T., and Stipes, J. (2005). Cooperating unmanned vehicles. In *Proceedings of the IEEE International Conference on Networking, Sensing and Control*, pp. 326–331.
- Shen, W.M., Salemi, B., and Will, P. (2002). Hormone-inspired adaptive communication and distributed control for CONTRO self-reconfigurable robots. *IEEE Transactions on Robotics and Automation* **18**(5): 700–712.
- van der Merwe, R., Doucet, A., de Freitas, N., and Wan, E. (2000). *The Unscented Particle Filter*. Technical Report CUED/F-INFENG/TR 380, Cambridge University Engineering Department, Cambridge, England.

- Venkatasubramanian, V., Rengaswamy, R., Yin, K., and Kavuri, S. N. (2003a). A review of process fault detection and diagnosis; Part I: Quantitative model-based methods. *Computer and Chemical Engineering* **27**: 293–311.
- Venkatasubramanian, V., Rengaswamy, R., and Kavuri, S. N. (2003b). A review of process fault detection and diagnosis; Part II: Qualitative models and search strategies. *Computer and Chemical Engineering* **27**: 313–326.
- Venkatasubramanian, V., Rengaswamy, R., Kavuri, S. N., and Yin, K. (2003c). A review of process fault detection and diagnosis; Part III: Process history based methods. *Computer and Chemical Engineering* **27**: 327–346.
- Verma, V., Gordon, G., Simmons, R., and Thrun, S. (2004). Real-Time Fault Diagnosis: Tractable Particle Filters for Robot Fault Diagnosis. *IEEE Robotics and Automation Magazine* **11**(2): 56–66.
- Washington, R. (2000). On-board real-time state and fault identification for rovers. In *Proceedings of the IEEE International Conference on Robotics & Automation*, pp. 1175–1181.
- Yamamoto, M., Iwamura M., and Mohri, A. (1998). Time-optimal motion planning of skid-steer mobile robots in the presence of obstacles. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 32–37.
- Yim, M., Zhang, Y., Lamping, J., and Mao, E. (2001). Distributed control for 3D metamorphosis. *Autonomous Robots* **10**(1): 41–56.
- Yoshida, E., Murata, S., Tomita, K., Kurokawa, H., and Kokaji, S. (1999). An experimental study on a self-repairing modular machine. *Robotics and Autonomous Systems* **29**(1): 79–89.
- Zykov, V., Mytilinaios, E., Adams, B., and Lipson, H. (2005). Self-reproducing machines. *Nature* **435**(7038): 163–164.