On Fault Detection and Diagnosis in Robotic Systems

ELIAHU KHALASTCHI, College of Management Academic Studies, Rishon LeZion, Israel
MEIR KALECH, Ben-Gurion University of the Negev, Beersheba, Israel

The use of robots in our daily lives is increasing. Different types of robots perform different tasks that are too dangerous or too dull to be done by humans. These sophisticated machines are susceptible to different types of faults. These faults have to be detected and diagnosed in time to allow recovery and continuous operation. The field of Fault Detection and Diagnosis (FDD) has been studied for many years. This research has given birth to many approaches and techniques that are applicable to different types of physical machines. Yet the domain of robotics poses unique requirements that are very challenging for traditional FDD approaches. The study of FDD for robotics is relatively new, and only few surveys were presented. These surveys have focused on traditional FDD approaches and how these approaches may broadly apply to a generic type of robot. Yet robotic systems can be identified by fundamental characteristics, which pose different constraints and requirements from FDD. In this article, we aim to provide the reader with useful insights regarding the use of FDD approaches that best suit the different characteristics of robotic systems. We elaborate on the advantages these approaches have and the challenges they must face. To meet this aim, we use two perspectives: (1) we elaborate on FDD from the perspective of the different characteristics a robotic system may have and give examples of successful FDD approaches, and (2) we elaborate on FDD from the perspective of the different FDD approaches and analyze the advantages and disadvantages of each approach with respect to robotic systems. Finally, we describe research opportunities for robotic systems' FDD. With these three contributions, readers from the FDD research communities are introduced to FDD for robotic systems, and the robotics research community is introduced to the field of FDD.

CCS Concepts: • **Computer systems organization → Robotic components**; **Robotic control**; **Robotic autonomy**;

Additional Key Words and Phrases: Fault detection, fault diagnosis, robots

ACM Reference format:
Eliahu Khalastchi and Meir Kalech. 2018. On Fault Detection and Diagnosis in Robotic Systems. *ACM Comput. Surv.* 51, 1, Article 9 (January 2018), 24 pages.
https://doi.org/10.1145/3146389

1 INTRODUCTION

Robots are physical systems with varying degrees of autonomy that operate in different and dynamic physical environments, e.g., satellites, Martian rovers, unmanned aerial, ground, or underwater vehicles. The use of robots in our daily lives is increasing. Based on data published in World Robotics [1], Guizzo [2] points to an increase in the world robot population from 4.5 million in 2006 to 8.6 million in 2008. Recent reports [3, 4] by the International Federation of Robotics (IFR) from 2016 describe yearly increases of 15% and 25% in sales of service and industrial robots,

Authors' addresses: E. Khalastchi and M. Kalech; emails: eliahukh@colman.ac.il, kalech@bgu.ac.il.

https://doi.org/10.1145/3146389

respectively. The use of robots is appealing for tasks that can be referred to as the four Ds—too Dangerous, too Dull, too Dirty, and too Difficult—to be done by humans. Such tasks include surveillance and patrolling [5], aerial search [6], rescue [7], and mapping [8]. Like any physical system, these intricate and sometimes expensive machines are prone to different types of faults such as wear and tear, noise, or control failures [9].

Fortunately, there are various types of Fault Detection and Diagnosis (FDD) approaches that may be applied to robotic systems. The motivation to use FDD is to facilitate recovery from a damage caused by faults. In the robotics domain, faults have the potential to affect the robot's efficiency, cause failures, or even jeopardize the safety of the robot or its surroundings [10]. When a fault is detected, it is imperative to proceed with a diagnosis process to identify which internal components are involved. This diagnostic information can be used for recovery or for decision-making purposes such as using undamaged redundant systems or re-planning [11]. The ability to recover successfully allows the system to be reliable, robust, and efficient, which is ultimately the reason for applying FDD.

While the field of FDD has been studied for many years, the study of FDD for robotics is relatively new. In recent years, a few surveys have been presented [12, 13]. These surveys focus on traditional FDD approaches and how these approaches may broadly apply to generic types of robots (e.g., wheeled mobile robots). Yet there is a wide variety of robotic systems. The aim of this article is providing the reader useful insights regarding the use of FDD approaches that best suit the different robotic systems.

To meet this aim, we survey FDD approaches from two perspectives, which form our main and novel contributions: (1) the different types of robotic systems and (2) the different FDD approaches. To generally capture the variety of robotic systems, we discuss different characteristics of robotic systems, where each characteristic has a great influence on the selection of a suitable FDD approach. Each characteristic is treated as a degree, where having a high degree is typically associated with greater FDD-challenges. For instance, a robot that is characterized by having a high degree of autonomy would have greater FDD-challenges than its less autonomous counterpart. For each characteristic, we discuss the challenges and provide examples of successful FDD approaches. The second perspective focuses on the different FDD approaches, their advantages and disadvantages, and how each approach may suit the different constraints and requirements of robotic systems. A third contribution is the description of research opportunities in the field of FDD for robotics. With these three contributions, readers from the FDD research communities are introduced to FDD for robotic systems, and the robotics research community is introduced to the field of FDD, and both are provided with useful insights regarding the association of robotic systems with suitable FDD approaches.

The rest of the article is organized as follows. In Section 2, we provide a background regarding the description of robotic systems, fault taxonomy, and FDD approaches taxonomy. In Section 3, we describe the characteristics of robotic systems and their impact on the selection of FDD approaches. In Section 4, we elaborate on the different FDD approaches. Section 5 provides a summary, which maps different aspects and characteristics of robotic systems to suitable FDD approaches. In Section 6 we describe different research opportunities in the field of FDD for robotics, and Section 7 concludes the article.

2 BACKGROUND

A robotic system is a set of physical and virtual (hardware and software) components that are embodied within a physical machine. These components work together in order to achieve certain goals within a physical environment [14]. In a general sense, a robotic system is engaged in three main processes [15]: sensing, thinking, and acting. The local environment and the robot's own

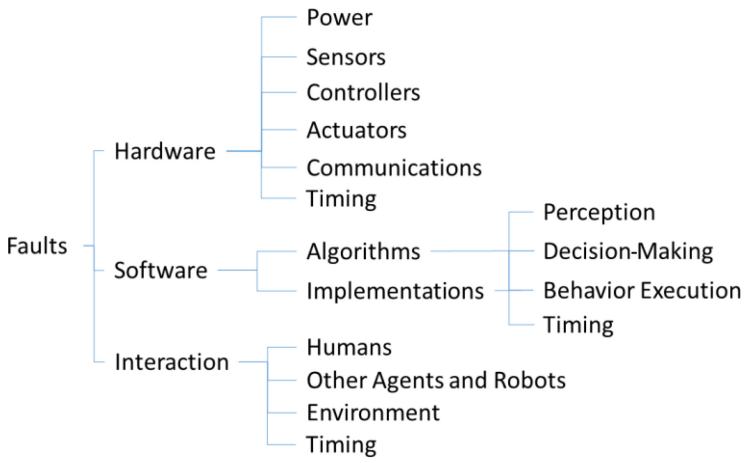

Fig. 1. Fault taxonomy for robotic systems.

body are **sensed** by different sensors. The "**thinking"** component involves (a) the extraction of information from the sensor-data, and (b) the use of acquired knowledge to decide on a course of action(s) to achieve the goal(s). There are many types of architectures for implementing the "thinking;" some architectures may integrate Artificial Intelligence (AI) [16] techniques such as planning [14] and learning [17]. The "thinking" is executed by programmable devices onboard the robot. Then, according to the decisions made by the robot, instructions are issued to different actuators that, in turn, **act** and affect the robot and its local environment. These effects are sensed by the robot's sensors and this cycle continues until the desired goals are achieved.

Unfortunately, these sophisticated and sometimes expensive machines are susceptible to different kinds of faults. Figure 1 depicts the fault taxonomy of robotic systems. Figure 1 combines and extends notions presented in Refs. [9], [18], and [19]. Faults are subdivided into hardware, software, and interaction related faults. Hardware faults might occur to any physical component of the robotic system, namely, the low-level sensing and acting. Such faults influence the information-feed of the robot and its ability to perform instructions. Software faults may regard faulty algorithms and/or faulty implementations of correct algorithms. Such faults influence the cognitive behavior of the robot, i.e., perception, decision making, and behavior execution. Interaction related faults may be the result of internal faults of the robot, or the result of exogenous events. Timing related faults, e.g., missed deadlines, might occur at the levels of the hardware, software, or interaction.

Fortunately, many FDD approaches exist. Figure 2 depicts the taxonomy of FDD approaches. Figure 2 combines notions presented in Refs. [12] and [20]. FDD approaches are typically divided into three categories: data-driven, model-based, and knowledge-based. Data-driven approaches are model free. Online data is usually used to statistically differentiate a potential fault from historically observed normal behavior, e.g., via Principle Component Analysis [21]. Model-based approaches [22] typically use analytical redundancy to detect and diagnose faults. The correct behavior of each component in the robotic system is modeled analytically, and the expected output can be compared to the observed output. Quantitative models involve mathematical equations, which typically describe the functionality of components. Qualitative models involve logic-functions, which typically describe the behavior of components by describing qualitative relations between the observed variables. Knowledge-based [23] approaches typically associate recognized behaviors with predefined known faults and diagnoses.

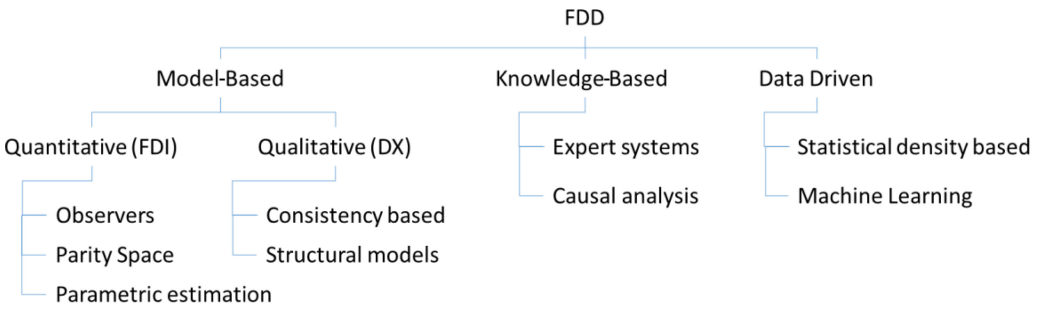

Fig. 2. Taxonomy for FDD approaches.

Fig. 3. Different robotic systems with an increasing degree of autonomy (left to right).

There are different types of robotic systems with different FDD constraints and requirements, and there are many FDD approaches that can be borrowed from other fields of study. In this article, we aim to provide useful insights that will allow the reader to associate different robotic systems with suitable FDD approaches, given the advantages these approaches have and the challenges they must face. To meet this aim, we use two perspectives: (1) In Section 3, we elaborate on FDD from the perspective of the different characteristics the robotic system may have. (2) In Section 4, we elaborate on FDD from the perspective of the different FDD approaches.

3 FDD—DIFFERENT ROBOTIC SYSTEMS PERSPECTIVE

Nowadays, defining what constitutes a robotic system has become an increasingly difficult question to answer. First, there is a broad range of systems that can be considered robotic. Second, the appearance of sophisticated machines that may fit the general description in the previous section has blurred the line between robotic and non-robotic systems. As a result, one may think that there is nothing unique about robotic systems that stand out in the context of FDD. Indeed, hardware components (such as sensors and actuators) and software components can be diagnosed for faults using exiting FDD approaches taken from other fields of study. Yet, the combination of certain characteristics of robotic systems challenges FDD approaches in different ways. Thus, to generally capture the different types of robotic systems, we believe it is preferable to focus on such characteristics.

It is important to note that systems may depict different degrees of each robotic-characteristic. For instance, "autonomy" is a robotic-characteristic, and yet not all robots are fully autonomous. Figure 3 depicts different robotic systems with an increasing degree of autonomy (from left to right). A robotic exoskeleton or suit [24] (left) is fully controlled by the operator and thus, can be considered with a zero degree of autonomy. Next (center) is an Unmanned Ariel Vehicle (UAV). Such a vehicle may be auto-piloted, but missions are controlled by human operators. On the right is the NASA Martian-laboratory curiosity. Communication is depended on Mars

Table 1. The Characteristics of Robotic Systems

Characteristic	FDD Challenges	Typically Addressed by
Dependency on exteroceptive sensors	Predicting sensor readings for sensor fault detection	Sensor fusion techniques Machine learning techniques
	Exploiting the information of exteroceptive sensors for FDD	Fault injection and supervised learning Particle filtering
Autonomous control	No operator Generating expectations	Model-based approaches Machine-learning approaches (to some extent)
	No remote supporting systems Creating a computationally light FDD mechanism	
	Continuous operation Online fault detection Quick fault detection	
Deliberation	Utilizing the plan for FDD	Model-based reasoning (plan is used as the model)
Dynamic context of operation	Recognizing or distinguish different contexts Detection of contextual faults	Windowing Clustering
Interaction with the environment	Unknown faults FDD for interaction related faults: Human-robot Robot-robot Social diagnosis Team-plan faults Coordination faults Scalability	Research opportunities: Human-robot Ad hoc teams FDD for robotic teams is typically handled with model-based diagnosis

rotation and orbit and thus may take 3 to 22 minutes. Hence, curiosity has to take some real-time decisions autonomously and cannot wait for instructions from human operators or remote supporting systems. Having a higher degree of certain robotic-characteristics might challenge FDD approaches in different ways. Table 1 lists five characteristics of robotic systems. As a system displays a higher degree of a characteristic, different FDD challenges emerge. The table depicts the main key points which are discussed in the next sections. For each characteristic, we discuss the challenges it imposes on FDD and the typical approaches of FDD used to address these challenges.

3.1 Dependency on Exteroceptive Sensors

In every system, sensors can be added for diagnosis purposes (e.g., car diagnostic sensors) with a certain cost (e.g., price, weight, complexity). Yet there are many existing robotic systems that do not possess such sensors or the cost for adding them is too high. Thus, FDD approaches for robotic systems are challenged to exploit existing sensors to determine the occurrence of faults and their root causes. Internal sensors, which are designed to provide the state of internal components (e.g., thermometer, battery level), can be very useful for FDD. Yet the challenge arises

from exteroceptive sensors, which are thus the focus of this section. Robotic systems are heavily dependent on exteroceptive sensors, which provide them with the state of the environment (e.g., camera, range detectors). Exteroceptive sensors form two main challenges for FDD: (1) detecting faulty exteroceptive sensors and (2) utilizing the information of exteroceptive sensors for fault detection.

3.1.1 Faulty Exteroceptive Sensors. In every sensor-based system, it is important to detect sensor faults. Indeed, faulty sensors might lead to incorrect beliefs, decisions, and unexpected behaviors of a robotic system. In order to detect sensor faults, sensor readings should be compared to some predefined prediction or expectation. Exteroceptive sensors sense a physical environment, which is dynamic and highly uncertain. Hence, it is challenging to predict the readings of exteroceptive sensors. This challenge is increased when there is no external source of perception, which can be compared to the robot's perception (e.g., a human operator, a radar station, other robots). FDD approaches are thus challenged to generate expectations for the robot's sensors in order to detect sensor faults.

A model-based approach for predicting the sensors' readings under normal or faulty conditions is by using *a priori* models (analytical or stochastic). However, such an approach may only apply to internal sensors since robotic systems may operate in unknown environments where it is hard for such *a priori* models to predict the readings of exteroceptive sensors [25]; an unpredictable environment (and its impact on sensors) can be hardly modeled, if at all. Fortunately, sensor fusion [26] techniques apply data-driven approaches that may overcome this challenge and can be utilized for exteroceptive sensor fault detection [27]. When multiple sensors sense different aspects of the (unknown) environment, their readings can be fused to form a consensus. Any inconsistency between the readings of a sensor and the fused consensus may indicate the occurrence of a fault. Since there is no need for an external perception source to compare with, and no need to predict the readings of exteroceptive sensors, then sensor-fusion-based fault detection approaches are very attractive for robotic systems. Sensor-fusion-based fault detection approaches for robotic systems include different algorithms such as: Kalman filters [28–31], Dempster-Shafer [32, 33], correlation and distribution-based [34–36], and Bayesian networks [37, 38].

3.1.2 Utilize Exteroceptive Sensors for FDD. Exteroceptive sensors sense the environment. As such, these sensors can only implicitly and indirectly provide useful information for diagnosing faults in internal parts of the robot. FDD mechanisms for robotic systems are thus challenged to exploit the information of exteroceptive sensors. Specifically, the challenge is to capture fault expressions as they befall in the environment and associate them with faulty internal components.

A data-driven approach can address this challenge for robotic systems via fault injection and supervised learning, e.g., Refs. [39–42]. Injected faults carry expressions in the robot's environment. These expressions are sensed by the exteroceptive sensors and their readings are registered in a training data set. A supervised learning algorithm is used then to create an FDD model from the training data set. However, when using fault injection (simulated or real), one cannot confidently account for all possible faults. To be able to detect new faults, over-fitting the training data must be avoided. In addition, fault injection is delicate; injecting a fault directly to a data stream might not imitate how the fault propagates. Injecting faults in a simulation provides more accurate results, enabling the fault to propagate through the system. However, simulated faults might differ (subtly or significantly) from real faults. While injecting a real fault to a physical robot might be preferable, this is accompanied by the risk of damaging the robot.

Another data-driven approach for utilizing exteroceptive sensors for FDD is to use particle filters, e.g., see Refs. [43–46]. Particle filtering is a statistical approach that is attractive for fault detection, in general, due to its ability to handle noisy sensing and motion [43]. This ability is

particularly useful for exploiting exteroceptive sensors of robotic systems, which provide noisy and partial information about the environment. Particles are used to indicate posterior probability distribution over a state space comprised of a normal state and different fault modes to different components. As the robot operates, only the particles that are consistent with the sensor readings are allowed to remain. Gradually, the particles indicate the most probable states and eventually reside on one state indicating a normal or a faulty behavior. Yet, the number of maintained particles is an issue. To enable the detection of rare events or low-probability fault states, a large number of particles is often needed, which might be impractical for a computationally limited robot. Verma et al. [43] have suggested several promising techniques to reduce the number of particles.

3.2 Autonomous Control

Robotic systems can range from being totally controlled (e.g., a robotic exoskeleton or suit [24]) to systems with a high degree of autonomy. Fully autonomous robots do exist; some of these robots carry out a simple task (e.g., vacuum cleaning) or operate within a very narrow context under the strict confines of their direct environment (e.g., a robotic arm in an assembly line). Other more complex systems may have "a human in the loop" with varying degrees of control and monitoring (e.g., a rover on Mars). A system is considered more autonomous when it is (a) less dependent on human operators and (b) remote supporting systems, and (c) able to operate for extensive periods of time without intervention. These three attributes challenge FDD approaches in different ways.

As the dependency on a human operator and remote support systems decreases, thus, it becomes the role of the robot to monitor itself and detect faults. The first challenge is to be able to generate expectations using nothing else but the robot's own perceptions; there is no external source to compare with (e.g., human operator, ground radar station) as we have already discussed in Section 3.1.1. The next challenges for an FDD mechanism is to detect faults quickly and online. A robot has to continue its operation autonomously even in the presence of faults. This means that faults have to be quickly detected (and handled) as they occur, rather than retrospectively after the operation has ended (and the data of the entire operation is available). The final challenge is to implement an FDD mechanism that requires low computational resources (CPU, Memory). Since there is less dependency on remote supporting systems, the FDD process has to be carried out onboard the robot. The robot has a limited computational power, most of which is dedicated to mission oriented processes. A computationally demanding FDD process might interfere with other processes and ironically be the cause for faults. The main difficulty in creating an FDD mechanism for an autonomous robot is to address all of these challenges together, i.e., generating expectations, quickly, online, with a low computational burden.

One approach to address these challenges is with model-based diagnosis [47–50]. The diagnostic process relies on an explicit *a priori* model of the normal system behavior, its structure, and/or its known faults. Behavioral models depict the normal behavior of the system by a set of analytical equations or logical formulas. Given the (online) system input (e.g., instructions), the model can quickly produce an expectation for the robot's behavior (e.g., expected sensor readings). Inconsistencies between the observed behavior and the produced expected behavior are suspected to be caused by faults. Fault isolation is the process of selecting (minimal) sets of components which, if regarded as faulty, may explain these inconsistencies [51]. The computational burden is depended on the type and fidelity of the model.

Statistical data-driven approaches, such as outlier-detection [52], may not be an attractive choice for an FDD mechanism as large amounts of data are to be processed online, carrying a heavy computational load, and might not detect the fault in time [12, 20]. Yet, some techniques do improve traditional approaches [35, 53] and address these challenges. Another data-driven approach is to utilize machine learning [39, 42, 54, 55]. Learning (supervised, semi-supervised or unsupervised)

produces an FDD model, which can be quickly used online. Learning offline reduces the online computational load, but produces a static model, which may not fit new behaviors. Learning online increases the computational load but produces a dynamic model. An attractive compromise is to add the detected faults to a dynamic model [56].

3.3 Deliberation

Fault tolerance is generally considered in a very limited way, and it is typically not integrated within the control architecture of robotic systems [57, 58]. This is unfortunate since aspects like deliberation can be exploited for FDD. The robotic control paradigms can be divided into three categories: reactive (behavioral), deliberative (planning), and hybrid (both reactive and deliberative) [15]. As opposed to a reactive architecture, deliberative architectures involve planning. Sensor data and stored knowledge (e.g., the outcomes of actions) are used to maintain a model of the world. Given this information, the robot "thinks ahead" and comes up with the best plan for achieving its goals. Since planning is a time-consuming process, hybrid architectures are used to add reactivity to the system. Typically, hybrid architectures include three layers: a reactive layer for fast reactions (e.g., obstacle avoidance), an intermediate supervisory layer, and a deliberative layer for planning. For a comprehensive survey about deliberation approaches for autonomous robots, you may refer to Ref. [59].

Having a plan, can be very useful for FDD since it provides context (i.e., the task at hand), expectations (i.e., possible outcomes of actions), and focus (i.e., which monitored attributes are currently important?). In particular, to form a plan, the planner has to have information regarding the expected outcome of each action; the outcome of an action forms the preconditions of the next step of the plan. Note that in the field of robotics, there is a distribution of expected outcomes for each action [60] due to the highly dynamic and uncertain physical environment in which the robot may operate. Given a plan, execution monitoring [12, 61] can compare the observed outcome of an action to the expected outcome. Any discrepancies may indicate (a) an inadequate choice of an action, or (b) a fault to the robot, which did not allow it to carry out the action successfully (or observe the expected outcome). In the highly dynamic environment in which robotic systems may operate, the inadequate choice of an action does not necessarily indicate a fault in the plan. For instance, during a flight, a UAV may detect obstacles that necessitate re-planning its flight path [62]. Thus, we focus on the utilization of a plan for FDD purposes.

Note that execution monitoring for low-level components can be done by either model-based, Knowledge-based, data-driven approaches or a hybrid combination of the three [12]. Execution monitoring for higher-leveled cognitive components requires reasoning at an abstract level, i.e., beliefs and intentions the robot has [59]. This reasoning can be applied by a qualitative model-based approach where the plan is used as the model.

The following are examples of such plan related model-based reasoning approaches. An early example can be found in PLANEX [63], which exploited plans from the well-known STRIPS [64] to detect faults in a robot. In Ref. [61], the authors present the utilization of FLUX [65] for high-level reasoning. When discrepancies arise between expectations and actual perception, a reasoning process is triggered, and diagnoses and recovery plans are generated. Another approach applies execution monitoring via Temporal Action Logic (TAL) [66]—a logic for reasoning about action and change. For instance, in Ref. [67], TAL is utilized for execution monitoring a plan generated by a TAL planner [68] for a rotor-based UAV. In Ref. [59], the authors give examples of plan-invariant approaches. Namely, conditions that have to hold during the entire execution of a plan are either given or extracted. Then, these conditions are monitored where a violation of these invariants may indicate a fault. In addition, the authors in Ref. [59] argue that goal reasoning is also essential

where a robotic system is expected to continuously operate for an extended period of time; if goals are not being achieved, then this may indicate a fault.

For multi-robot systems, diagnosing deliberation related faults is more challenging. Each robot has an individual plan, which is expected to support the global plan. The global plan and the individual plans of other robots may be only partially observable. Yet, an FDD mechanism is expected to detect a malfunctioning multi-robot system and differentiate global-plan faults from individual-plan related faults or faulty robots. Model-based approaches can potentially handle partially observable plans [69, 70] and uncertainty [71]. In later work, Micalizio and Torasso [72] presented a novel methodology, named Cooperative Weak-Committed Monitoring (CWCM). CWCM exploits nondeterministic action models to carry out two main tasks: detecting action failures and the reconstruction of possible beliefs a robot has had about the environment. They show that CWCM is effective in identifying and explaining action and interaction failures even in scenarios where the system observability is significantly reduced.

Since the decision-making of the robot is dependent on the beliefs or world model the robot has, it is very important to monitor the perception process. The perception process fuses raw sensor readings into abstract beliefs, e.g., filtered range sensor readings and camera inputs into a relative distance to a recognized object. Model-based diagnosis can be used for belief management and to find inconsistencies between beliefs that might indicate faults [73]. In addition, beliefs can be backtraced to the low-level sensors' components that were used to construct the beliefs [74], which is very useful for diagnosis.

It is our belief that the issue of deliberation related faults in robotic systems is not as researched as other aspects of robotic systems, and research opportunities may exist.

3.4 Dynamic Context of Operation

A context of operation is comprised of the robot's state, the environment state, and the task at hand. For instance, a robot that is on the move, in an outdoor environment, foraging for food. Naturally, the context in which the robot operates is dynamic. An FDD mechanism must consider that observations which are legitimate under one context might not be legitimate under another; these are contextual faults. For instance, a zero airspeed value is not legitimate while the UAV is flying. The context in which the robot operates might not be given as an input for an FDD mechanism. Thus, FDD approaches are challenged to distinguish between contexts such that contextual faults can be detected.

The transition between contexts may be discrete (e.g., from "driving" to "parking") or continuous (e.g., gradually from "hot" to "cold" temperature). A data-driven approach can process the robot's data as a time-series (sensor readings, beliefs, instructions, communications, etc.) in an online sliding window fashion [35, 75, 76]. The data in the window may implicitly indicate the current context even if the transition between contexts is continuous; the underlying assumption is that the robot behaves differently under different contexts and it is expressed in the robot's data. Another approach to implicitly recognize a context is via clustering techniques (e.g., KNN). Past records of the robot's (fused) data are clustered. Each cluster may indicate a different context. Online, a (fused) data instance of the robot is associated with the closest cluster (e.g., via Mahalanobis Distance metric [77]) and thus is assigned with a context. A cluster may also present the context of a faulty behavior [78].

Once the current context is recognized, the associated FDD approach should be triggered. Specifically, machine-learning-based FDD approaches might be susceptible to inaccuracies when the data of different contexts is mixed. Separating the training data set into sets of different contexts allows the use of different learned models, which best suit each context (e.g., a decision tree for context A

and a neural network for context B). Online, the context is recognized and the associated learned model is used for context-based FDD [75].

3.5 Interaction with the Environment

Robotic systems interact with the highly dynamic and uncertain physical environment. Robots may interact with objects, other robots [79], and humans [80, 81]. Naturally, these interactions carry a high degree of uncertainty, where unexpected outcomes might lead to unknown faults and failed interactions. An FDD mechanism is challenged to detect unknown faults and to distinguish between failed interactions that resulted from internal faults and failed interactions that resulted from exogenous events.

The literature on Human-Robot Interaction (HRI) is mainly concerned with safety [82–85]. Namely, the literature is concerned with safety protocols and standards, risk assessment techniques, collision avoidance, and recovery from accidental encounters. Works about safety in HRI emphasize the need for safety even in the presence of faults. Yet, to the best of our knowledge, there is a knowledge gap regarding the use of FDD approaches that are dedicated to detect and diagnose HRI-related faults.

On the other hand, the interaction between robots is extensively researched, even in the context of interaction related faults [86]. Interaction with other robots introduces new challenges as well as new opportunities for FDD. The main opportunity is to exploit the observations and behaviors of other robots such that social fault detection and diagnosis can be achieved. In a multi-robot scenario there are two main types of interactions: interaction between robots that are unfamiliar with each other and yet work together to achieve collaboration without pre-coordination, as in ad hoc teams [87], and a predesigned interaction between robots that are a part of the same team [88].

The setting of an ad hoc scenario appeals to real-world scenarios such as when unfamiliar agents suddenly have to collaborate in order to handle an emergency, e.g., put out a fire. Robots that were not predesigned to collaborate with each other are expected to form a team. Each robot should assume the role that best fit its abilities with respect to the performance of the other robots. Social FDD is very relevant to ad hoc teams as one robot may need to detect a faulty member and replace its role. To the best of our knowledge, the challenge of social FDD for robots without preconditions has yet to be addressed.

When teamwork is predesigned, social diagnosis can be achieved [89]. A team member has an expectation of how the other team members should act or what they are trying to achieve. If a team member acts in an unexpected manner, then other team members may recognize this as a fault. Nevertheless, teamwork introduces a large variety of weak points that an FDD mechanism should take into account; in particular, different FDD approaches tackle: team-plan related faults [69–72, 90], coordination faults [91–96], and implementation issues such as scalability [97] for large size teams and decentralization [98, 99].

A special case involving a large number of robots in a multi-robot scenario is a swarm robotic system. Research on swarm systems has attracted much attention in recent years [100]. A robotic swarm system is a system that consists of multiple, not very intelligent, interconnected individual robots, where swarm capabilities and intelligence emerge from their simple topologic behaviors. Early research considered swarm robotics to be fault tolerant due to the inherit redundancy in such a system [101]. Recent research [102] shows that faulty robots can and will significantly affect healthy robots and lead to task failure. Since an individual robot reacts to the observed behavior of its neighbors, fault symptoms are transferred among neighboring robots and cascade along the swarm. Faults in swarm systems can be classified into two types based on their influences: (1) *topology faults*—change the swarm topology, and (2) *component faults*—only affect a

robot's actuators, sensors, or controllers [100]. Robotic swarms present two main challenges for FDD mechanisms. The first challenge is to detect and diagnose a single individual robot among thousands in the swarm, identifying the one seeding a topological fault that might lead to overall faulty swarming behavior. The second challenge applies to the implementation of a scalable and reliable FDD mechanism. To be reliable, an FDD mechanism would have to collect a lot of local information from individual robots and construct a global swarm view. However, it is challenging for the FDD mechanism to be scalable while doing so. Comprehensive surveys for fault detection and diagnosis in a robotic swarm can be found in the Ph.D. dissertation of Hui Keng Lau [101] and also in Ref. [100].

4 FDD—DIFFERENT APPROACHES PERSPECTIVE

In the previous section, we have discussed FDD from the perspective of robotic systems. In particular, we discussed how different characteristics of robotic systems impact FDD, and we gave examples of relevant works. In this section, we look into FDD for robotic systems from the perspective of the different FDD approaches. We start with the citation of relevant surveys, and then delve into data-driven, model-based, and knowledge-based approaches. This section is very relevant for those of the robotics research community who wish to research FDD.

4.1 FDD for Robotic Systems—Related Surveys

Pettersson [12] surveys execution monitoring approaches in robotics. He uses the knowledge and terminology from an industrial control field to classify different execution monitoring approaches applied to robotics with an emphasis on mobile robots. In this survey, model-based approaches are referred to as "analytical approaches" and are subdivided into the following categories: parameter estimation, parity relations, and observers. Data-driven approaches are subdivided into univariate and multivariate statistical monitoring. Knowledge-based approaches are subdivided into the following categories: casual analysis, expert systems, and artificial neural networks. The advantages and disadvantages of each approach are comprehensively discussed.

Zhuo-hua et al. [13] presented a survey of the state-of-the-art in fault detection and diagnosis for fault tolerant control of wheeled mobile robots (WMRs) under unknown environments. They introduce the main components, typical kinematics models, and fault models of WMRs and error models of inertial navigation sensors. The survey presents main approaches such as multiple model-based, particle filter-based, and sensor fusion-based approaches, as well as layered fault tolerant architecture. The survey discusses the main challenges and difficulties of these approaches.

Steinbauer [9] conducted a survey about the faults of autonomous robots in the context of RoboCup [103]. He presents an adapted fault taxonomy suitable for autonomous robots and provides information regarding the nature, relevance, and impact of faults in robot systems that are beneficial for researchers dealing with fault mitigation and management in autonomous systems. A survey was sent to 68 research groups around the world participated in RoboCup. The survey questions RoboCup participants about the hardware and software used, hardware and software faults, the faults of algorithms, as well as counter-measures used. The major observations drawn from this survey are: (1) Faults in sensors have a similar frequency of occurrence as faults in the robot platform, but their negative impact on the success rate of the mission is much greater. (2) In general, algorithms are more affected by faults than hardware. (3) The failure of algorithms caused by missed deadlines or faulty configuration is a major problem.

Crestani and Godary-Dejean [57] present a survey of fault tolerance concepts, means, and implementation in mobile robotics control architectures. Their overview discusses works of fault tolerance in the field of control engineering, i.e., model-based and data-based methods, and deems these approaches as efficient but not sufficient for robotics. Their main reasons are (1) real-time

requirements are difficult to meet, (2) faults related to high-level knowledge are much more challenging than faults to actuators or sensors, and (3) recovery mechanisms are not flexible enough to manage the situations arising during complex missions. The survey discusses 11 different algorithms for fault tolerance in the control architecture of different robots and concludes with a discussion about the applicability of fault tolerance for mobile robots.

For robotics in the UAV domain, a very extensive survey can be found in the work of Marzat et al. [104]. This survey of model-based fault diagnosis focuses on those methods that are applicable to aerospace systems. These systems are similar to the platforms of UAVs. The survey highlights the characteristics of aerospace models, recalls generic nonlinear dynamical modeling from flight mechanics, and presents a unifying representation of sensor and actuator faults. An extensive bibliographical review supports a description of the key points of fault detection methods that rely on analytical redundancy. The approaches that best suit the constraints of the field are emphasized and recommendations for future developments in in-flight fault diagnosis are provided.

In the next section, we describe each of the main FDD approaches more fully.

4.2 Data-Driven Approaches

Data-driven approaches rely on sampled data to extract useful information that enables fault detection and possibly even diagnosis. Some data-driven approaches statistically compare online data to known fault expressions or historically observed normal behaviors. Statistical-filtering approaches such as Kalman filters [28–31] or Particle Filters [43–46] are very suitable for filtering statistical noise and providing a good estimation for the expected (normal) value. A statistical match (small deviation) classifies the online data as normal, and outliers (large deviations) are classified as anomalies or unknown faults [105].

Other data-driven approaches apply machine learning [106] techniques to produce a fault detection or diagnosis model. For example, a Neural Network (NN) is a supervised learning algorithm that is appealing for modeling nonlinear systems such as robotic manipulators (arms with several degrees of freedom) [107, 108]. Informative features are selected or extracted from a dataset. According to these features, an FDD model is learned from a dataset. The learned model provides the ability to classify new unseen data. In particular, the FDD model is used to classify online data that is produced in real time by the robot, determining whether or not the data expresses a fault and even associates its diagnosis. Supervised approaches require a classified dataset for model creation, while unsupervised approaches do not. The model learning can take place offline, e.g., on past observations that were recorded from previous operations, as well as online, e.g., applied on current data that is consumed in a sliding window fashion [36].

Data-driven approaches can be further subdivided to univariate or multivariate statistical analysis. That is, whether the data stream of one measured attribute is dependent on another single attribute (univariate) or it is dependent on several other attributes (multivariate). For instance, the altitude measurement of a UAV is dependent on the actions of the elevators, ailerons, and throttle, and thus, it is a multivariate scenario. All data-driven approaches are heavily dependent on the quality of the data. Supervised approaches rely on the existence of sufficient examples in the training data and generalization of examples is sometimes questionable. In the domain of robotics in which previously unseen faults might occur, unsupervised approaches are more suitable, yet it can be claimed that unsupervised approaches are generally less accurate than supervised approaches for known fault detection.

Some faults can only be detected within the context of another data instance; therefore, multivariate approaches are preferable to univariate approaches [12]. For example, if an altimeter of a UAV reports a drifting value, it is not necessarily a fault, for, in fact, the UAV may be climbing.

However, in the context of reports of leveled altitude by the GPS, it can be assumed that the contradiction is due to some fault. Fault detection must consider the multivariate context.

The great advantage of data-driven approaches is that they are model free. Robotic systems can be found in a variety of platforms, e.g., static, mobile, UAVs, and Unmanned Ground Veihcle (UGV). A model-free approach is general in nature and can be applied to different robotic systems. Furthermore, some robotic systems are very complex, thus attempting to analytically model the correct behavior of each component, and its integrated interactions with the system and the environment can be quite difficult and sometimes even impractical. In this context, a model-free approach is easier to implement. An additional benefit of data-driven approaches is the detection of unknown faults via outlier detection. Data-driven approaches may be applied for prognostics as well, i.e., predicting the time at which a system or a component will no longer perform its intended function. Schwabacher and Goebel [109] provided a survey of AI techniques applied to prognostics. The survey focuses on data-driven prognostics, prognostic architectures, and applications of prognostics.

Data-driven approaches cope with some challenges as well: (1) Typically, knowledge about the structure of the system or data dependencies is available and could potentially contribute to fault detection and diagnosis; pure, data-driven approaches do not usually exploit existing and available knowledge about the robotic system. (2) Robotic systems may provide a lot of data. Processing the entire data quickly and online for fault detection purposes may be impractical, and consequently, faults might be detected too late. An FDD approach must apply some form of dimension reduction by feature selection and extraction. Deciding which features are essential is challenging. (3) Historical data of robotic systems from manufacturers is not typically classified in terms of faulty/normal examples. Moreover, since not all faults necessarily cause an observable unexpected behavior, even data that is considered to be fault-free might contain hidden faults and anomalies. The lack of data classification results in one of two scenarios: (1) The use of unsupervised or clustering methods, which might be less accurate than supervised approaches. (2) The use of supervised methods that rely on fault injection (simulated or real) in which one cannot confidently account for all possible faults. In addition, fault injection is delicate. Injecting a fault directly to a data stream might not imitate how the fault propagates. Injecting faults in a simulation provides accurate results, enabling the fault to propagate through the system; however, simulated faults might differ (subtly or significantly) from real faults. Therefore, while injecting a real fault to a physical robot might be preferable, this is accompanied by the risk of damaging the robot. Such issues challenge the use of supervised approaches. (4) Diagnosis is a great challenge in a purely data-driven approach, which operates without additional information regarding the components of the system. Because of this, some model representation of the robotic system should be used or the association of fault expressions and their diagnoses should be learned in a supervised fashion.

The claims above are supported by our previous works. In Ref. [35], we introduced an unsupervised, online multivariate data-driven fault detection approach. Mahalanobis-distance [77] calculation is utilized to compare correlated streams of data with previously observed data. If the distance crosses a dynamically updated threshold, a fault is declared. In later work [36], we introduced an unsupervised approach utilizing *a priori* knowledge about the system. The approach produces greater fault detection accuracy. The fault detection accuracy was improved even further in our most recent work [42]. In a broader sense, the combination of different data-driven approaches may result in a better, more accurate approach – by taking the advantages of each [12, 20].

4.3 Model-Based Approaches

As opposed to data-driven approaches, model-based approaches are totally dependent on *a priori* knowledge about the system. The diagnostic process relies on an explicit model of the normal system behavior, its structure, and/or its known faults. Behavioral models depict the normal behavior

of the system by a set of analytical equations or logical formulas. Inconsistencies between the observed behavior and the expected behavior, produced by the model, are suspected to be caused by faults. Fault isolation is the process of selecting (minimal) sets of components which, if regarded as faulty, may explain these inconsistencies [51].

When such a model is available, some Model-Based Diagnosis (MBD) approaches possess the advantages of being able to detect unknown faults and be applied quickly and online. These are essential requirements in robotic domains. As expected, the main disadvantage of MBD approaches is the overhead cost of constructing such analytical models [12]. In robotic domains the construction of such models is even more challenging, because the models need to take into account the dynamic context of the robots, i.e., the environment and the task at hand, as well as the robot's complexity.

Structural or dependency models depict dependencies among components. Relative to behavioral models, structural models are easier to construct, since it is easier to describe associations between components than describing their normal behavior analytically. These structural models cannot typically help with the task of fault detection, because they cannot extrapolate an expected behavior with which to compare with an observed behavior. Rather, these models are used in the process of diagnosis in that information regarding interdependencies in the system allows the diagnosis process to look for the root cause of a fault.

Some MBD approaches use fault models. Fault models are models that depict the behavior of (known) faults. If these behaviors are recognized, then the fault can be reported. An FDD approach that only relies upon fault models might not be able to detect unknown faults. Robotic systems operate in dynamic physical environments and are certain to encounter unknown faults. Hence, the use of fault models alone is incomplete in the domain of robotics.

Model-based approaches for fault detection and diagnosis are studied by two distinct and parallel research communities, the Fault Detection and Isolation (FDI) community [110] and the DX community [111, 112]. FDI approaches derive from control theory and statistical decision making. DX approaches derive from computer science and AI. Both approaches rely on a model that describes the system; however, the concepts, assumptions, and techniques of the two approaches are different.

4.3.1 DX Model-Based Diagnosis. DX model-based diagnosis approaches (hereinafter referred to as DX MBD) are consistency-based. These approaches primarily address diagnosis and less fault detection. The system model is described with logic formulas. If these formulas are not satisfiable given the system observations and the assumption that all components are healthy, a fault is detected. Diagnoses are produced by selecting minimal sets of components that, if assumed to express certain fault modes, satisfy the system description and output observations [51].

There are very few studies about fault detection and diagnosis for robots in the DX community. We believe that the reason for this is that the classical DX MBD approach cannot be easily or directly implemented for robots. The main challenges are: (1) Constructing the model in a logical language—sampled data that is used as the system's observations is continuous in nature and affected by noise, and therefore, it is challenging to describe these observations in logical formulas. (2) Presenting the dynamic nature of the robot. (3) Representing the behavior of a robot in a temporal model. (4) Representing the interaction between a robot and the environment. (5) Handling fault propagation within the *sense-think-act* cycle.

Steinbauer and Wotawa [73] discuss and demonstrate the importance of detecting, diagnosing, and repairing faults in the beliefs of a robot when faced with uncertainties and/or exogenous events. The robot's beliefs are assumed to be maintained in first-order logic forms, and a DX MBD approach is used to demonstrate how inconsistencies in the beliefs can be found. However,

a pure diagnosis based on the content of the knowledge base, i.e., literals, is not sufficient for diagnosing all types of faults; models of faults in other control levels of the robotic system have to be incorporated as well.

Akhtar and Kuestenmacher [113] address the challenge of diagnosing unknown faults that are related to exogenous events by the use of native physical knowledge. The native physical knowledge is represented by the physical properties of objects which are formalized in a logical framework. Upon the detection of an unknown fault, this DX MBD approach uses reasoning methods to associate the fault with possible exogenous events. However, formalizing the physical laws demands great *a priori* knowledge about the context in which the robot operates (environment and task). It would be quite challenging to apply native physical knowledge formalization to the dynamic context nature of robots.

Some attempts to automatically generate a model that can be used by a DX MBD approach in robots have been introduced, and Sadov et al. [114] automatically produced a partial model. A robot's action is invoked in minimal repetition while different components are switched off. The model produced formally describes which components are essential for which actions. In recent work [74], we semi-automatically derived a model from a robot's control software. The model associates a behavior with the invoked sensors and actuators. Zaman et al. [48] exploit the properties of an underlying Robotic Operating System (ROS) [115] to automatically diagnose and repair software and hardware faults. Niggemann et al. [116] presented a taxonomy of learning problems related to model construction tasks and a learning algorithm that is able to construct a behavioral model of a hybrid system with scalable precision. These approaches eliminate the issue of the overhead for manual model construction, but are still unable to produce a model which takes into account the dynamic environment in which the robotic system interacts. In Ref. [117], the environment of a vehicle is modeled. Struss and Dobi [117] simplified the problem of modeling the environment by using the road as a reference, thus transforming a three-dimensional problem into a single-dimensional problem in which aspects such as the slope or road curvature are abstracted into Boolean attributes.

4.3.2 FDI Model-Based Approaches. Isermann [22] presents a short introduction into FDI model-based fault detection and diagnosis. As opposed to DX approaches, FDI approaches use numeric analytical behavioral models [12]. These behavioral models are mainly useful for fault detection but can also derive a diagnosis. Constraints between the system's inputs and outputs are analytically modeled, typically as a function of time. Models can either be continuous-time or discrete-time, deterministic or stochastic, linear or nonlinear. Typically, FDI approaches are divided into three categories: (1) *parameter estimation*—where the value of a parameter represents a physical feature of the system. (2) *Observers*—that rely on estimation of unknown variables. (3) *Parity relations*—that rely on the elimination of unknown variables [118].

The observations are compared to the modeled estimations, and a residual is computed. The residual should tend to zero in the normal case and increase otherwise. The residual should also be robust to disturbances to the system, modeling errors and noise. A structured residual is sensitive to some faults and not to others. A directed residual is a vector oriented along a particular direction of each fault. According to the computed residual, an isolation of the fault can take place.

FDI approaches are very suitable for fault detection and isolation in internal components within the robot, e.g., a motor. Internal components are structured within the robotic system, and hence, they can be modeled as having inputs and outputs, which connect them to the other components. These internal components have a low level of interaction with the environment, if any, and exogenous effects can be regarded as disturbances to the component. Hence, the behavior of these internal components is relatively easier to model.

FDI approaches are less suitable for a robot as a whole, as an entire set of complex behaviors under a larger variety of contexts, some of which are unknown, would have to be analytically modeled. While hardware components can be modeled, the control software is not very easy to model analytically; it is very complex and sometimes nondeterministic. Furthermore, the environment in which the robot operates is being discovered on an ongoing basis, as the robot operates. One would need to avoid modeling the environment or use a very high level of abstraction. Finally, the robot operates in a *sense-think-act* cycle. It may operate independently according to the sensed environment, affect it, and sense it again in a repeating fashion. This cycle is quite challenging to analytically model as input-output relations.

For example, consider a wheeled mobile robot with an electrical motor. The motor is relatively easy to model—we can model the expected output of rounds per minute (RPM) as a function of the electrical charge given to the motor (input). However, it will be very difficult to analytically model why the robot decided to use a particular charge and whether it made the right choice under the given context in which it operates.

FDI is an extensively researched field, and works in the robotics domain can be found in abundance, for example, mobile robots [119], actuators of robots [120], and sensors of robots [121]. In her extensive survey bridging FDI and DX, Travé-Massuyès [118] presents diagnosis as it is understood in the control (FDI) and AI (DX) fields and exemplifies how different theories of these fields can be synergistically integrated to provide better diagnostic solutions and achieve improved fault management in different environments. These FDI and DX model-based approaches rely on manual construction of analytical models. These models become harder to construct as the robotic system becomes more complex or increases its interaction with the environment.

4.4 Knowledge-Based Approaches

Knowledge-based approaches typically mimic the behavior of a human expert. Symptoms are quickly associated with diagnoses. The strength of knowledge-based approaches is the opportunity to combine model-based and data-driven approaches in a hybrid FDD approach [12], e.g., a data-driven approach can detect a fault, and a model-based approach can associate the fault to a diagnosis. The knowledge-based approaches can be divided into two categories: causal analysis and expert systems.

Causal analysis methods typically rely on causal modeling of fault-symptom relationships and are used mostly for fault isolation [122]. For example, a signed direct graph (SDG) can be used to represent the topology of a robotic system as causal relationships between the system's components and known faults' symptoms. Faulty expressions of both hardware and software components can be represented as nodes in the graph including sensors values, process variables, actuators feedbacks, etc. Each node stores a value and a limit above/below which the node is considered to be faulty. Assuming that a single fault affects only a single node and the fault does not change the causal relationships in the graph, the causal edges will link the fault symptoms with the fault's origin. A knowledge-based FDD approach can use heterogeneous causal models as well. In Ref. [123], the knowledge was categorized into five different models: design, sensor, historical, mission, and fault knowledge. The fault isolation knowledge, gathered by human experts, can be combined with the robot's knowledge (e.g., sensor knowledge). The main disadvantage of causal analysis methods is their reliance on *a priori* fault isolation knowledge. Robots might encounter unknown faults, and thus, no *a priori* knowledge is complete; some faults will be left undetected or undiagnosed.

The second category of knowledge-based systems is expert systems. Expert systems imitate the reasoning of human experts while isolating faults. Typically, IF-THEN rules are used to represent the knowledge of the expert system [12]. These rules can be found from first principles or the structural description of the robotic system. Expert systems typically rely on Boolean values,

Table 2. Summary of Advantages and Challenges FDD Approaches Have When Applied to Robotic Systems

Approaches	Advantages	Challenges
Data-driven	Model free Detection of unknown faults	Being quick and online
Model-based	Quick & Online Detection of unknown faults	Modeling complex robotic systems Modeling context (especially environment)
Knowledge-based	Quick & Online Low computational burden	Detection of unknown faults

which might make the system sensitive to uncertainties [12]. Fuzzy logic [124] provides a way to overcome this problem [125]. The main disadvantage of expert systems is the handling of unknown faults; it is challenging to extrapolate new diagnoses from the fixed set of experts' rules.

4.5 Summary

Table 2 summarizes the general advantages and challenges the architypes of FDD approaches have when applied to robotic systems. Note that this is a general summary and exceptions can be found. We can see that each approach has a certain challenge that is an advantage of the other approaches. Thus, constructing a hybrid approach that combines elements of these approaches may overcome these challenges [12, 36, 42].

5 MAPPING FDD AND ROBOTIC SYSTEMS—A SUMMARY

To summarize the previous sections, Figure 4 depicts a mapping of FDD for robotic systems.

Figure 4 depicts a robotic system as a set of hardware (bottom) and software (top) components. The hardware components might encounter different hardware faults, which are typically diagnosed by model-based or knowledge-based diagnosis approaches. Exteroceptive sensors (bottom left) might encounter different sensor faults and are not easily exploited for FDD. As the dependency on exteroceptive sensors increases, data-driven approaches become attractive for FDD; in particular, sensor fusion for sensor fault detection, and machine learning and particle filtering for utilizing sensor-readings for diagnosis.

The computing components (middle), e.g., Memory, CPU, and Storage, support the software components (top). These components might encounter hardware faults. The executed programs may encounter different software faults. These types of faults were not the focus of this article since they are not unique to robotics, and can be diagnosed by approaches from other fields of study.

Figure 4 depicts the major control software components: perception, deliberation, and actuation. Perception involves turning sensor readings into abstract beliefs or a world model. Model-based approaches can be used for belief management where inconsistencies may indicate faults, and can be associated to low-leveled components (typically sensors) as possible diagnoses. The decision-making process of the robot may include deliberation (planning). Model-based reasoning may allow the diagnosis of deliberation-faults at an abstract level, i.e., when the expected outcome of a robot's action is not being observed and thus may indicate a fault. Actuation, or behavior execution, involves the transition of high-leveled instructions onto low-leveled controller-actuator commands. At this level, execution monitoring can be done by either data-driven, model-based, or knowledge-based approaches.

The control mechanism for the robot must consider the context of operation (top left). As the robot is intended to operate in more contexts or in a dynamic context, FDD should consider

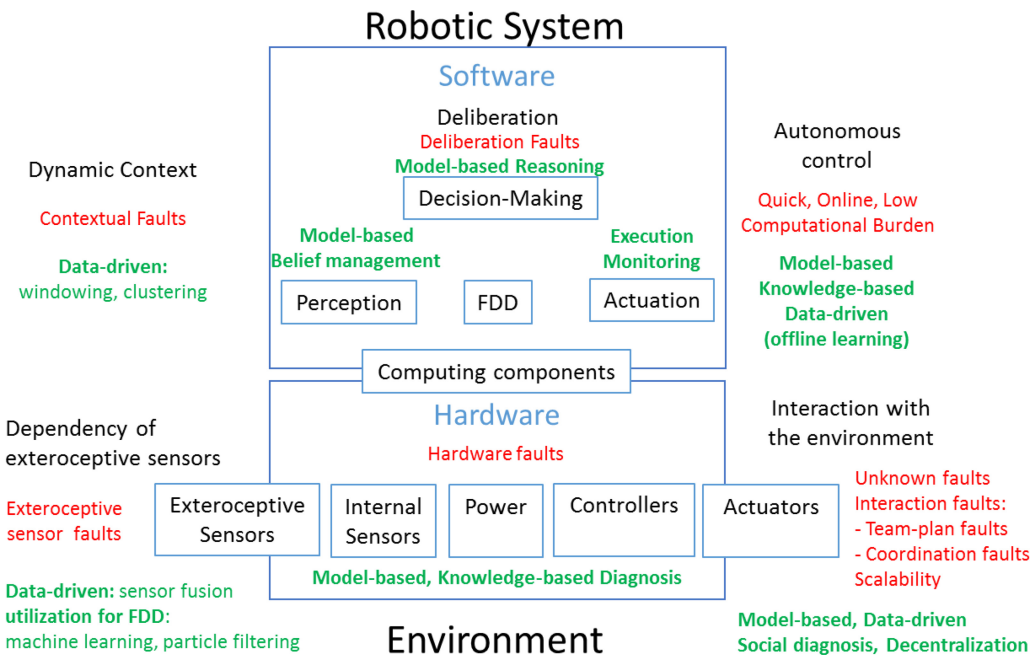

Fig. 4. FDD for robotic systems – mapping it all together.

contextual faults. If the context is not given to the FDD mechanism, the transition between contexts can be detected by data-driven approaches, namely, windowing or clustering.

As the robotic system becomes more autonomous (top right), the FDD mechanism is constrained to being quick, online, and computationally light. This is due to the fact that FDD must be carried onboard the robot, and faults have to be detected quickly and online to allow continued operation. Model-based and knowledge-based approaches can meet these constraints. Data-driven approaches, on the other hand, are less attractive in this aspect as they might require the long processing of large data and not meet fault detection deadlines. An exception can be made for machine-learning approaches where the learning is done offline, and the learned model is used online.

Finally, the actions of the robotic system interact with the environment. As the interaction with the environment grows, the robotic system might encounter environment-related faults, i.e., unknown faults and interaction faults. Knowledge-based approaches are typically unattractive for unknown fault detection. The diagnosis of teamwork faults such as team-plan or coordination faults can be done by model-based or data-driven approaches. Human-robot interaction faults and FDD for ad hoc robotic teams are two subjects that are open for research.

Next, we discuss research opportunities in greater detail.

6 RESEARCH OPPORTUNITIES

FDD for human-robot interaction related faults: As we have stated in Section 3.4, the literature on HRI is mainly concerned about safety. We believe that FDD approaches concerning interaction-related faults between humans and robots should be researched. Several interesting challenges may arise. First, humans are robust, and may compensate for a faulty behavior of a robot during interaction. This might challenge fault detection as faults are prevented by the human.

Second, to a robot, humans are very complex. Reasoning about the behavior of a human during interaction might be very challenging. Thus, it will be challenging for an FDD mechanism to detect fault-preventing behaviors of humans, or diagnose the reason for a failed interaction.

FDD for an ad hoc team of robots: Another research opportunity is mentioned in Section 3.4—FDD for ad hoc teams. Since no *a priori* knowledge is assumed, detecting faulty interactions between robots is very essential and very challenging. First, a robot would need to apply goal/ behavior detection to the other robots, recognizing and modeling the system that has just formed. Then, the next challenge is to detect which robot might be faulty or less adequate for the role it has assumed, or causing failed interactions. This requires sharp observations and reasoning at a very abstract level.

FDD for deliberation-related faults: As presented in Section 3.3, some work has been done regarding FDD for deliberation-related faults. Yet, we believe this issue can still be considered as a research opportunity. The main challenge, in our view, is the fact that there is no other plan to compare with the one the robot has. If the plan has a faulty step, then the result may be detected but it is hard to diagnose, i.e., differentiate between internal faults and a faulty step in the plan. This can potentially be solved in a multi-robot scenario, where robots (with different knowledge, beliefs, and world models) may apply goal-recognition algorithms to monitor the behaviors of their teammates. When the desired goal is not recognized, perhaps it is due to a faulty plan-step. For a single robot scenario, this is naturally more challenging.

FDD for improved performance: Even a (predesigned) team of robots may display degraded performance due to a number of reasons. Private misbeliefs of individual robots, unfitting role assignment, misuse of resources, and so on, can all contribute to degraded performance. In a team of robots (or agents), one robot may assume the role of monitoring the team and give correcting instructions aimed at improving the team's performance. For example, one robot may assume the role of a coach [126]; a coaching robot's role is to improve the team via communications. It is not a centralized coordinator that instructs the agents exactly what to do at each point of time. Rather, it advises the robots on how to improve by giving them useful and limited information, general instructions, or altering the team plan. Works about a coaching robot [127] in the Robocup competition typically adjust the strategy of the team or find weaknesses about the opponent team such that they could be exploited.

A new research opportunity lies in the use of FDD techniques to isolate behaviors or beliefs, which might degrade the performance of the team. In particular, model-based diagnoses may have the potential to recognize private misbeliefs of individual robots about their own role and about the roles of their teammates. The coaching robot may provide the necessary information to correct these beliefs and, in turn, improve the team's performance.

FDD during development-time of robotic systems: As most FDD approaches and control architectures for robotic systems focus on fault diagnosis or tolerance during operation time, a new research opportunity lies in the use of FDD techniques to reduce the development time of robotic systems. Under development, especially during testing, many types of faults can occur, e.g., incomplete control code or software bugs, faults in the planning policy, hardware wear, and communication faults. The occurrence of such faults slows down the progress of development as developers typically manually diagnose such faults. A development-time FDD mechanism should be embedded in the development environments. There are several works that compare different development environments for robotic systems, e.g., see Refs. [128] and [129]. From these works, we can see that some attention was given to fault tolerance during development. This attention is typically in the form of open source philosophy, which allows debugging, and layered application,

which allows to focus on mission-oriented code. To the best of our knowledge, there has yet to be an attempt to investigate the correlation between using an active FDD mechanism and the reduction of development time.

7 CONCLUSION

In this article, we have surveyed FDD for robotic systems. We aimed to provide the reader with useful insights regarding the use of FDD approaches, which best suit the different characteristics of robotic systems. We elaborated on the advantages these approaches have and the challenges they must face. We used two perspectives: (1) we elaborate on FDD from the perspective of the different characteristics robotic system may have and gave examples of successful FDD approaches, and (2) we elaborated on FDD from the perspective of the different FDD approaches, and analyzed the advantages and disadvantages of each approach with respect to the characteristics of the robotic systems. Finally, we described research opportunities for robotic systems FDD. With these three contributions, it is the authors' hope that readers from the FDD research communities are introduced to FDD for robotic systems, and the robotics research community is introduced to the field of FDD.

REFERENCES

[1] World Robotics. 2010. [En ligne]. Retrieved from http://www.worldrobotics.org/.
[2] E. Guizzo. 2010. World robot population reaches 8.6 million. *IEEE Spectrum*.
[3] IFR. 2016. *Executive Summary World Robotics. 2016 Service Robot*. The International Federation of Robotics (IFR).
[4] IFR. 2016. *Executive Summary World Robotics. 2016 Industrial Robots*. The International Dederation of Robotics (IFR).
[5] N. Agmon, S. Kraus, and G. A. Kaminka. 2008. Multi-robot perimeter patrol in adversarial settings. In *Proceedings of the IEEE International Conference on Robotics and Automation*.
[6] M. A. Goodrich, B. S. Morse, D. Gerhardt, J. L. Cooper, M. Quigley, J. A. Adams, and C. Humphrey. 2008. Supporting wilderness search and rescue using a camera-equipped mini UAV. *Journal of Field Robotics* 25, 1–2 (2008), 89–110.
[7] A. Birk and S. Carpin. 2006. Rescue robotics - a crucial milestone on the road to autonomous systems. *Advanced Robotics* 20, 5, 2006.
[8] S. Thrun. 2002. Robotic mapping: A survey. *Exploring Artificial Intelligence in the New Millennium*. Morgan Kaufmann, 1–35.
[9] G. Steinbauer. 2013. A survey about faults of robots used in robocup. In *RoboCup 2012: Robot Soccer World Cup XVI*. Springer, Berlin, 344–355.
[10] B. S. Dhillon. 1991. *Robot Reliability and Safety*. Springer.
[11] J.-H. Shin and J.-J. Lee. 1999. Fault detection and robust fault recovery control for robot manipulators with actuator failures. In *Proceedings of the IEEE International Conference on Robotics and Automation*.
[12] O. Pettersson. 2005. Execution monitoring in robotics: A survey. *Robotics and Autonomous Systems* 53, 2, 73–88.
[13] D. Zhuo-hua, C. Zi-xing, and Y. Jin-xia. 2005. Fault diagnosis and fault tolerant control for wheeled mobile robots under unknown environments: A survey. In *Proceedings of the 2005 IEEE International Conference on Robotics and Automation (ICRA'05)*.
[14] B. Siciliano, L. Sciavicco, L. Villani, and G. Oriolo. 2010. *Robotics: Modelling, Planning and Control*. Springer Science & Business Media.
[15] R. Brooks. 1986. A robust layered control system for a mobile robot. *IEEE Journal of Robotics and Automation* 2, 14–23.
[16] R. Murphy. 2000. *Introduction to AI Robotics*. MIT Press.
[17] J. Kober, J. A. Bagnell, and J. Peters. 2013. Reinforcement learning in robotics: A survey. *The International Journal of Robotics Research* 32, 11 (2013), 1238–1274.
[18] J. Carlson and R. R. Murphy. 2003. Reliability analysis of mobile robots. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA'03)*.
[19] B. Shim, B. Baek, S. Kim, and S. Park. 2009. A robot fault-tolerance approach based on fault type. In *Proceedings of the 9th International Conference on Quality Software (QSIC'09)*.
[20] V. Chandola, A. Banerjee, and V. Kumar. 2009. Anomaly detection: A survey. *ACM Computing Surveys (CSUR)* 41, 15.
[21] I. Jolliffe. 2005. *Principal Component Analysis*. Wiley Online Library.

[22] R. Isermann. 2005. Model-based fault-detection and diagnosis—status and applications. *Annual Reviews in Control* 29, 71–85.
[23] R. Akerkar and P. Sajja. 2010. *Knowledge-Based Systems*. Jones and Bartlett Publishers.
[24] G. Zeilig, H. Weingarden, M. Zwecker, I. Dudkiewicz, A. Bloch, and A. Esquenazi. 2012. Safety and tolerance of the ReWalk™ exoskeleton suit for ambulation by people with complete spinal cord injury: A pilot study. *Journal of Spinal Cord Medicine* 35, 96–101.
[25] J. Gage and R. R. Murphy. 2010. Sensing assessment in unknown environments: A survey. *IEEE Transactions on Systems, Man, and Cybernetics* 40, 1–12.
[26] H. B. Mitchell. 2007. *Multi-Sensor Data Fusion: An Introduction*. Springer Science & Business Media.
[27] R. C. Luo, C.-C. Yih, and K. L. Su. 2002. Multisensor fusion and integration: Approaches, applications, and future research directions. *IEEE Sensors Journal* 2, 107–119.
[28] S. I. Roumeliotis, G. S. Sukhatme, and G. A. Bekey. 1998. Sensor fault detection and identification in a mobile robot. In *Proceedings of the International Conference on Intelligent Robots and Systems (RSJ'98)*.
[29] M. H. Amoozgar, A. Chamseddine, and Y. Zhang. 2013. Experimental test of a two-stage kalman filter for actuator fault detection and diagnosis of an unmanned quadrotor helicopter. *Journal of Intelligent & Robotic Systems* 70, 107–117.
[30] K. Bader, B. Lussier, and W. Schon. 2016. A fault tolerant architecture for data fusion: A real application of Kalman filters for mobile robot localization. *Robotics and Autonomous Systems* 88, 11–23.
[31] M. Van, D. Wu, S. S. Ge, and H. Ren. 2016. Fault diagnosis in image-based visual servoing with eye-in-hand configurations using kalman filter. *IEEE Transactions on Industrial Informatics* 12, 16, 1998–2007.
[32] R. R. Murphy. 1998. Dempster-shafer theory for sensor fusion in autonomous mobile robots. *IEEE Transactions on Robotics and Automation* 14, 197–206.
[33] X. Yuan, M. Song, F. Zhou, Z. Chen, and Y. Li. 2015. A novel mittag-leffler kernel based hybrid fault diagnosis method for wheeled robot driving system. *Computational Intelligence and Neuroscience* 2015, Article 65.
[34] R. Lin, E. Khalastchi, and G. A. Kaminka. 2010. Detecting anomalies in unmanned vehicles using the mahalanobis distance. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA'10)*.
[35] E. Khalastchi, M. Kalech, G. A. Kaminka, and R. Lin. 2015. Online data-driven anomaly detection in autonomous robots. *Knowledge and Information Systems* 43, 13, 657–688.
[36] E. Khalastchi, M. Kalech, and L. Rokach. 2013. Sensor fault detection and diagnosis for autonomous systems. In *Proceedings of the 12th International Conference on Autonomous Agents and Multi-Agent Systems (AAMAS'13)*.
[37] A. Rezaee, A. A. Raie, A. Nadi, and S. S. Ghidary. 2011. Sensor fault tolerance method by using a bayesian network for robot behavior. *Advanced Robotics* 25, 2039–2064.
[38] N. Mehranbod, M. Soroush, and C. Panjapornpon. 2005. A method of sensor fault detection and identification. *Journal of Process Control* 15, 321–339.
[39] A. L. Christensen, R. O'Grady, M. Birattari, and M. Dorigo. 2008. Fault detection in autonomous robots based on fault injection and learning. *Autonomous Robots* 24, 49–67.
[40] F. Wotawa. 2016. Testing self-adaptive systems using fault injection and combinatorial testing. In *Proceedings of the IEEE International Conference on Software Quality, Reliability and Security Companion (QRS-C'16)*.
[41] G. Jäger, S. Zug, T. Brade, A. Dietrich, C. A. M. C. Steup, and A.-M. Cretu. 2014. Assessing neural networks for sensor fault detection. In *Proceedings of the IEEE International Conference on Computational Intelligence and Virtual Environments for Measurement Systems and Applications (CIVEMSA'14)*.
[42] E. Khalastchi, M. Kalech, and L. Rokach. 2017. A hybrid approach for improving unsupervised fault detection for robotic systems. *Expert Systems with Applications* 81, 372–383.
[43] V. Verma, G. Gordon, R. Simmons, and S. Thrun. 2004. Real-time fault diagnosis [robot fault diagnosis]. *IEEE Robotics & Automation Magazine* 11, 56–66.
[44] Y.-S. Sun, X.-R. Ran, Y.-M. Li, G.-C. Zhang, and Y.-H. Zhang. 2016. Thruster fault diagnosis method based on gaussian particle filter for autonomous underwater vehicles. *International Journal of Naval Architecture and Ocean Engineering* 8, 3, 243–251.
[45] M. Zając. 2014. Online fault detection of a mobile robot with a parallelized particle filter. *Neurocomputing* 126, 151–165.
[46] Z. Duan, Z. Cai, and J. Yu. 2006. Adaptive particle filter for unknown fault detection of wheeled mobile robots. In *Proceedings of the Proceedings of the 2006 IEEE/RSJ International Conferencer on Intelligent Robots and Systems*. 1312–1315.
[47] R. Isermann. 2005. Model-based fault-detection and diagnosis—status and applications. *Annual Reviews in Control* 29, 71–85.
[48] S. Zaman, G. Steinbauer, J. Maurer, P. Lepej, and S. Uran. 2013. An integrated model-based diagnosis and repair architecture for ROS-based robot systems. In *Proceedings of the International Conference on Robotics and Automation (ICRA'13)*.

[49] D. Stavrou, D. G. Eliades, C. G. Panayiotou, and M. M. Polycarpou. 2016. Fault detection for service mobile robots using model-based method. *Autonomous Robots* 40, 383–394.
[50] M. Hashimoto, H. Kawashima, and F. Oba. 2003. A multi-model based fault detection and diagnosis of internal sensors for mobile robot. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS'03)*.
[51] R. Reiter. 1987. A theory of diagnosis from first principles. *Artificial Intelligence* 32, 57–95.
[52] V. J. Hodge and J. Austin. 2004. A survey of outlier detection methodologies. *Artificial Intelligence Review* 22, 85–126.
[53] D. Pokrajac, A. Lazarevic, and L. J. Latecki. 2007. Incremental local outlier detection for data streams. In *Proceedings of the IEEE Symposium on Computational Intelligence and Data Mining (CIDM'07)*. 504–515.
[54] P. Goel, G. Dedeoglu, S. I. Roumeliotis, and G. S. Sukhatme. 2000. Fault detection and identification in a mobile robot using multiple model estimation and neural network. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA'00)*.
[55] F. Baghernezhad and K. Khorasani. 2016. Computationally intelligent strategies for robust fault detection, isolation, and identification of mobile robots. *Neurocomputing* 171, 335–346.
[56] G. Fagogenis, V. De Carolis, D. M. Lane. 2016. Online fault detection and model adaptation for underwater vehicles in the case of thruster failures. In *Proceedings of the International Conference on Robotics and Automation (ICRA'16)*.
[57] D. Crestani and K. Godary-Dejean. 2012. Fault tolerance in control architectures for mobile robots: Fantasy or reality? In *Proceedings of the 7th National Conference on Control Architectures of Robots (CAR'12)*.
[58] D. Crestani, K. Godary-Dejean, and L. Lapierre. 2015. Enhancing fault tolerance of autonomous mobile robots. *Robotics and Autonomous Systems* 68, 140–155.
[59] F. Ingrand and M. Ghallab. 2017. Deliberation for autonomous robots: A survey. *Artificial Intelligence* 247 (2017), 10–44.
[60] A. Bouguerra, L. Karlsson, and A. Saffiotti. 2008. Monitoring the execution of robot plans using semantic knowledge. *Robotics and Autonomous Systems* 56, 942–954.
[61] M. Fichtner, A. Grossmann, and M. Thielscher. 2003. Intelligent execution monitoring in dynamic environments. *Fundamenta Informaticae* 57, 371–392.
[62] Q. Xue, P. Cheng, and N. Cheng. 2014. Offline path planning and online replanning of UAVs in complex terrain. In *Proceedings of the Guidance, Navigation and Control Conference (CGNCC'14)*.
[63] R. E. Fikes. 1971. Monitored execution of robot plans produced by STRIPS. In *Proceedings of the IFIP Congress*.
[64] R. E. Fikes and N. J. Nilsson. 1971. STRIPS: A new approach to the application of theorem proving to problem solving. *Artificial Intelligence* 2, 189–208.
[65] M. Thielscher. 2005. FLUX: A logic programming method for reasoning agents. *Theory and Practice of Logic Programming* 5, 533–565.
[66] P. Doherty and J. Kvarnstrom. 2008. Temporal action logics. *Foundations of Artificial Intelligence* 3, 709–757.
[67] P. Doherty, J. Kvarnstrom, and F. Heintz. 2009. A temporal logic-based planning and execution monitoring framework for unmanned aircraft systems. *Autonomous Agents and Multi-Agent Systems* 19, 332–377.
[68] J. Kvarnstram. 2005. Linköping studies in science and technology, dissertation. In *TALplanner and Other Extensions to Temporal Action Logic*.
[69] N. Roos and C. Witteveen. 2009. Models and methods for plan diagnosis. *Autonomous Agents and Multi-Agent Systems* 19, 30–52.
[70] F. De Jonge, N. Roos, and C. Witteveen. 2009. Primary and secondary diagnosis of multi-agent plan execution. *Autonomous Agents and Multi-Agent Systems* 18, 267–294.
[71] R. Micalizio. 2009. A distributed control loop for autonomous recovery in a multi-agent plan. In *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI'09)*. 1760–1765.
[72] R. Micalizio and P. Torasso. 2014. Cooperative monitoring to diagnose multiagent plans. *Journal of Artificial Intelligence Research* 50, 1–70.
[73] G. Steinbauer and F. Wotawa. 2010. On the way to automated belief repair for autonomous robots. In *Proceedings of the 21st International Workshop on Principles of Diagnosis (DX'10)*.
[74] E. Khalastchi, M. Kalech, and L. Rokach. 2012. Multi-layered model based diagnosis in robots. In *Proceedings of the 23rd International Workshop on Principles of Diagnosis (DX'12)*.
[75] M. Kalisch. 2016. Fault detection method using context-based approach. In *Advanced and Intelligent Computations in Diagnosis and Control*. 383–395.
[76] M. Van, S. S. Ge, and H. Ren. 2017. Finite time fault tolerant control for robot manipulators using time delay estimation and continuous nonsingular fast terminal sliding mode control. *IEEE Transactions on Cybernetics* 47, 17, 1681–1693.
[77] P. C. Mahalanobis. 1936. On the generalised distance in statistics. *Proceedings of the National Institute of Sciences of India* 2, 49–55.

[78] R. Hornung, H. Urbanek, J. Klodmann, C. Osendorfer, and P. van der Smagt. 2014. Model-free robot anomaly detection. In *Proceedings of the International Conference on Intelligent Robots and Systems (IROS)*, 2014.
[79] P. U. Lima and L. M. Custodio. Multi-robot systems. In *Innovations in Robot Mobility and Control*. 1–64.
[80] M. A. Goodrich and A. C. Schultz. 2007. Human-robot interaction: A survey. *Foundations and Trends in Human-computer Interaction* 1, 3 (2007), 203–275.
[81] T. B. Sheridan. 2016. Human–robot interaction status and challenges. *Human Factors: The Journal of the Human Factors and Ergonomics Society* 58, 525–532.
[82] R. Alami, A. Albu-Sch{affer, A. Bicchi, R. Bischoff, R. Chatila, A. De Luca, A. De Santis, G. Giralt, J. Guiochet, and G. Hirzinger. 2006. Safe and dependable physical human-robot interaction in anthropic domains: State of the art and challenges. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (RSJ'06)*. 1–16.
[83] A. Bicchi, M. A. Peshkin, and J. E. Colgate. 2008. Safety for physical human–robot interaction. In *Springer Handbook of Robotics*. 1335–1348.
[84] M. Vasic and A. Billard. 2013. Safety issues in human-robot interactions. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA'13)*.
[85] T. S. Tadele, T. de Vries, and S. Stramigioli. 2014. The safety of domestic robotics: A survey of various safety-related publications. *IEEE Robotics & Automation Magazine* 21, 134–142.
[86] L. Iocchi, D. Nardi, and M. Salerno. 2000. Reactivity and deliberation: A survey on multi-robot systems. In *Proceedings of the Workshop on Balancing Reactivity and Social Deliberation in Multi-Agent Systems*. 9–32.
[87] P. Stone, G. A. Kaminka, S. Kraus, and J. S. Rosenschein. 2010. Ad hoc autonomous agent teams: Collaboration without pre-coordination. In *Proceedings of the AAAI*.
[88] G. A. Kaminka and I. Frenkel. 2005. Flexible teamwork in behavior-based robots. In *Proceedings of the National Conference on Artificial Intelligence*.
[89] M. Kalech and G. A. Kaminka. 2003. On the design of social diagnosis algorithms for multi-agent teams. In *Proceedings of the International Joint Conferences on Artificial Intelligence (IJCAI'03)*.
[90] S. B. Stancliff, J. Dolan, and A. Trebi-Ollennu. 2009. Planning to fail - reliability needs to be considered a priori in multirobot task allocation. In *Proceedings of the IEEE International Conference on Systems, Man and Cybernetics (SMC'09)*.
[91] R. Micalizio, P. Torasso, and G. Torta. 2004. On-line monitoring and diagnosis of multi-agent systems: A model based approach. In *Proceedings of the European Conference on Artificial Intelligence (ECAI'04)*.
[92] N. Roos, A. ten Teije, and C. Witteveen. 2003. A protocol for multi-agent diagnosis with spatially distributed knowledge. In *Proceedings of the 2nd International Joint Conference on Autonomous Agents and Multiagent Systems (AAMAS'03)*.
[93] M. Kalech. 2012. Diagnosis of coordination faults: A matrix-based approach. *Journal of Autonamous Agents and Multi-Agent Systems* 24, 11, 69–103.
[94] M. Kalech and G. A. Kaminka. 2007. On the design of coordination diagnosis algorithms for teams of situated agents. *Artificial Intelligence Journal* 71, 491–513.
[95] X. Li and L. E. Parker. 2007. Sensor analysis for fault detection in tightly-coupled multi-robot team tasks. In *Proceedings of the IEEE International Conference on Robotics and Automation*.
[96] X. Li and L. E. Parker. 2008. Design and performance improvements for fault detection in tightly-coupled multi-robot team tasks. In *Southeastcon*.
[97] M. Kalech and G. A. Kaminka. 2011. Coordination diagnostic algorithms for teams of situated agents: Sclaing-up. *Computational Intelligence* 27, 13, 393–421.
[98] M. Kalech, G. A. Kaminka, A. Meisels, and Y. Elmaliach. 2006. Diagnosis of multi-robot coordination failures using distributed csp algorithms. In *Proceedings of the National Conference on Artificial Intelligence*.
[99] M. Daigle, X. Koutsoukos, and G. Biswas. 2006. Distributed diagnosis of coupled mobile robots. In *Proceedings of the IEEE International Conference on Robotics and Automation*.
[100] L. Qin, X. He, and D. Zhou. 2014. A survey of fault diagnosis for swarm systems. *Systems Science & Control Engineering: An Open Access Journal* 2, 13–23.
[101] H. K. Lau. 2012. *Error Detection in Swarm Robotics: A Focus on Adaptivity to Dynamic Environments*. University of York.
[102] A. F. Winfield and J. Nembrini. 2006. Safety in numbers: Fault-tolerance in robot swarms. *International Journal of Modelling, Identification and Control* 1, 30–37.
[103] H. Kitano, M. Asada, Y. Kuniyoshi, I. Noda, and E. Osawa. 1997. Robocup: The robot world cup initiative. In *Proceedings of the 1st International Conference on Autonomous Agents*. ACM, 340–347.
[104] J. Marzat, H. Piet-Lahanier, F. Damongeot, and E. Walter. 2012. Model-based fault diagnosis for aerospace systems: A survey. *Proceedings of the Institution of Mechanical Engineers, Part G: Journal of Aerospace Engineering* 226, 1329–1360.

[105] V. Hodge and J. Austin. 2004. A survey of outlier detection methodologies. *Artificial Intelligence Review* 22, 85–126.
[106] J. R. Anderson, R. S. Michalski, J. G. Carbonell, and T. M. Mitchell. 1986. *Machine Learning: An Artificial Intelligence Approach*. Morgan Kaufmann.
[107] M. Van and H.-J. Kang. 2015. Robust fault-tolerant control for uncertain robot manipulators based on adaptive quasi-continuous high-order sliding mode and neural network. *Proceedings of the Institution of Mechanical Engineers, Part C: Journal of Mechanical Engineering Science* 229, 18, 1425–1446.
[108] M. Van, S. S. Ge, and H. Ren. 2017. Robust fault-tolerant control for a class of second-order nonlinear systems using an adaptive third-order sliding mode control. *IEEE Transactions on Systems, Man, and Cybernetics: Systems* 7, 12, 221–228.
[109] M. Schwabacher and K. Goebel. 2007. A survey of artificial intelligence for prognostics. In *Proceedings of the AAAI Fall Symposium*. 107–114.
[110] I. Hwang, S. Kim, Y. Kim, and C. E. Seah. 2010. A survey of fault detection, isolation, and reconfiguration methods. *IEEE Transactions on Cntrol Systems Technology* 18, 636–653.
[111] http://www.dx-2013.org/, International Workshop on Principles of Diagnosis, 2013.
[112] W. Hamscher, L. Console, and J. de Kleer. 1992. *Readings in Model-Based Diagnosis*. Morgan Kaufmann.
[113] N. khtar and A. Kuestenmacher. 2011. Using naive physics for unknown external faults in robotics. In *Proceedings of the 22nd International Workshop on Principles of Diagnosis (DX-2011),* 2011.
[114] V. Sadov, E. Khalastchi, M. Kalech, and G. A. Kaminka. 2010. Towards partial (and useful) model identification for model-based diagnosis. In *Proceedings of the 21st International Workshop on Principles of Diagnosis (DX'10)*.
[115] M. Quigley, K. Conley, B. Gerkey, J. Faust, T. Foote, J. Leibs, R. Wheeler, and A. Y. Ng. 2009. ROS: An open-source robot operating system. In *Proceedings of the ICRA Workshop on Open Source Software*.
[116] O. Niggemann, B. Stein, A. Vodencarevic, A. Maier, and H. K. Buning. 2012. Learning behavior models for hybrid timed systems. In *Proceedings of the 26th AAAI Conference on Artificial Intelligence*. AAAI.
[117] P. Struss and S. Dobi. 2013. Automated functional safety analysis of vehicles based on qualitative behavior models and spatial representations. In *Proceedings of the 24th International Workshop on Principles of Diagnosis (DX'13)*.
[118] L. Travé-Massuyès. 2014. Bridging control and artificial intelligence theories for diagnosis: A survey. *Engineering Applications of Artificial Intelligence* 27, 1–16.
[119] M. Yu, D. Wang, M. Luo, and D. Zhang. 2010. FDI and fault estimation based on differential evolution and analytical redundancy relations. In *Proceedings of the 11th International Conference on Control Automation Robotics & Vision (ICARCV'10)*.
[120] T. Hsiao and M.-C. Weng. 2012. A hierarchical multiple-model approach for detection and isolation of robotic actuator faults. *Robotics and Autonomous Systems* 60, 154–166.
[121] X. Zhang. 2011. Sensor bias fault detection and isolation in a class of nonlinear uncertain systems using adaptive estimation. *IEEE Transactions on Automatic Control* 56, 1220–1226.
[122] L. H. Chiang, R. D. Braatz, and E. L. Russell. 2001. *Fault Detection and Diagnosis in Industrial Systems*. Springer Science & Business Media.
[123] K. Hamilton, D. M. Lane, N. K. Taylor, and K. Brown. 2001. Fault diagnosis on autonomous robotic vehicles with recovery: An integrated heterogeneous-knowledge approach. In *Proceedings of the International Conference on Robotics and Automation (ICRA'01)*.
[124] C. Negoita, L. Zadeh, and H. Zimmermann. 1978. Fuzzy sets as a basis for a theory of possibility. *Fuzzy Sets and Systems* 1, 3–28.
[125] R. Patton, F. Uppal, and C. Lopez-Toribio. 2000. Soft computing approaches to fault diagnosis for dynamic systems: A survey. In *Proceedings of the 4th IFAC Symposium on Fault Detection Supervision and Safety for Technical Processes*. 198–211.
[126] P. Riley, M. Veloso, and G. Kaminka. 2002. Towards any-team coaching in adversarial domains. In *Proceedings of the 1st International Joint Conference on Autonomous Agents and Multiagent Systems*.
[127] G. Kuhlmann, W. B. Knox, and P. Stone. 2006. Know thine enemy: A champion robocup coach agent. In *Proceedings of the National Conference on Artificial Intelligence*.
[128] A. Elkady and T. Sobh. 2012. Robotics middleware: A comprehensive literature survey and attribute-based bibliography. *Journal of Robotics* 2012 (2012), Article ID 959013, 15 pages.
[129] J. Kramer and M. Scheutz. 2006. Development environments for autonomous mobile robots: A survey. *Autonomous Robots* 22, 12, 101–132.

Received June 2016; revised August 2017; accepted September 2017