A Hybrid Approach for Fault Detection in Autonomous Physical Agents

Eliahu Khalastchi, Meir Kalech, Lior Rokach
Information Systems Engineering, Ben-Gurion University of the Negev, Israel
{khalastc,kalech,liorrk}@bgu.ac.il

ABSTRACT
One of the challenges of fault detection in the domain of autonomous physical agents (or Robots) is the handling of unclassified data, meaning, most data sets are not recognized as normal or faulty. This fact makes it very challenging to use collected data as a training set such that learning algorithms would produce a successful fault detection model. Traditionally unsupervised algorithms try to address this challenge. In this paper we present a hybrid approach that combines unsupervised and supervised methods. An unsupervised approach is utilized for classifying a training set, and then by a standard supervised algorithm we build a fault detection model that is much more accurate than the original unsupervised approach. We show promising results on simulated and real world domains.

Categories and Subject Descriptors
I.2.9 [**Artificial Intelligence**]: Robotics – *Autonomous vehicles, Sensors.*

General Terms
Reliability, Experimentation

Keywords
Fault detection, Model-Based Diagnosis, Robotics, UAV.

1. INTRODUCTION
Autonomous physical agents such as Unmanned Vehicles (UVs) or robots are susceptible to a variety of hardware and software faults. These faults might lead to mission failure or even endanger the safety of the expensive agent or its environment. For example, a pitot-static system failure in an Unmanned Aerial Vehicle (UAV) might lead to a stall and then a crash. To continue operate autonomously, the agent must have an accurate fault detection mechanism. Upon fault detection a diagnosis process can be triggered and a decision on how to continue can be made.

Given the nature of autonomous physical agents, i.e. physical systems which operate autonomously and interact with a physical environment, an accurate fault detection mechanism faces several challenges: (1) since the agent is autonomous i.e., there is an absence of human operators, there is no other perception which can be compared to the agent's own perception. (2) The physical environment is both dynamic and nondeterministic and therefore the environment and the agent's effects are both very hard to model. (3) Physical faults have many expressions, such as a stuck value, a drifting value or abrupt intermittent offsets. Furthermore, some faults might have unknown expressions. (4) A fault expression can span over time. (5) Since the agent does not stop its operation, a fault must be detected as quickly as possible, online, and with high accuracy. (6) Since the agent is already engaged in heavy computations, e.g. vision processing, a fault detection mechanism should be kept computationally light. (7) Due to the environmental and behavioral contexts, this domain is also characterized for having unclassified data, which means the data is not recognized by the user as healthy or faulty.

Appears in: *Alessio Lomuscio, Paul Scerri, Ana Bazzan, and Michael Huhns (eds.), Proceedings of the 13th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2014), May 5-9, 2014, Paris, France.*

In a recent work [1], we presented an unsupervised approach (hereinafter SFDD) for fault detection in the domain of autonomous systems. This approach shows a high rate of fault detection and a low rate of false positives (false alarms). In this paper we aim to extend the state of the art by describing a hybrid approach which is based on the SFDD approach and is significantly more accurate. In addition, the presented approach is more suitable to meet the challenges for a fault detection mechanism in the domain of autonomous physical agents.

The SFDD approach is used offline to classify a training set. Then, even though the SFDD has a certain degree of false positives, a standard supervised learning algorithm is applied and a fault detection model is created. The created model is used online to detect faults.

We empirically evaluate the presented approach on simulated and real world domains: a high fidelity flight simulator, a commercial UAV and a laboratory robot. We show that the learnt fault detection model is more accurate than the original unsupervised SFDD approach.

The significances of this paper are by (1) introducing a hybrid approach to fault detection of autonomous physical agents. We show that this approach is general and can be applied with other unsupervised algorithms too. (2) In addition, we theoretically analyze our approach and provide an explanation why it is only slightly affected by the unsupervised SFDD false positives. (3) Finally, we empirically evaluate the hybrid approach and show its accuracy.

The paper is structured as follows. In the next section we discuss the related work. In Section 3 we describe the hybrid approach: the problem description, the outline of the approach, and how we use the SFDD for offline classification of a training set. Also, we describe the learning process and why it is only slightly affected by the false positives of the SFDD. In Section 4 we describe the experimental setup and in Section 5 we show the results. Finally, Section 6 discusses the different aspects of the hybrid approach.

2. RELATED WORK
Steinbauer conducted a survey on the nature of faults of autonomous robots [2]. The survey participants are developers competing in different leagues of the RoboCup competition [3]. The reported faults were categorized as hardware, software, algorithmic and interaction related faults. The survey concludes that hardware faults have a high negative impact on mission success. In this paper we focus on detecting such faults.

Three general approaches are usually used for fault detection: Knowledge-Based systems, Model-Based, and Data Driven approaches. Knowledge based systems [4] typically associates recognized behaviors with predefined known faults and hence, are less likely to detect an unknown fault.

Model-based approaches [5] on the other hand, are very equipped to detect unknown faults. The expected behavior of each component is modeled analytically. The system output is compared to the modeled output and a high residual indicates a fault. However, in the domain of autonomous physical agents, the task of modeling the behavior of components is very challenging due to the context of the physical environment.

Steinbauer et al. [6] use a model based approach for detecting failures in the control software of a robot. The software architecture was utilized for the model creation. Faults are detected with observers that observe different aspects of software components. They do not use any model of the environment. In recent work Steinbauer et al. [7] emphasize the importance of the robot's belief management and fault detection with respect to the real-world dynamic environment.

Struss and Dobi [8] use a model based approach for automating the functional safety analysis of vehicles. They use a qualitative model of the vehicle and a model of the environment. The environment model includes a spatial representation of positions of the vehicle and other objects relative to the road and their interference under different scenarios. Yet, the environment model is simplistic and cannot account for every possible scenario.

Data driven approaches, supervised or unsupervised, are model free and have a natural appeal for detecting unknown faults. Unsupervised data driven approaches are usually slower than model-based or knowledge-based approaches. Online outlier detection is applied and a decision is made whether or not an outlier is the expression of a fault [9]. This usually involves heavy statistical computations and thus challenging to be done quickly as the domain of autonomous physical agents requires.

For example, in previous work [10] we created an online unsupervised data driven approach for the domain of unmanned vehicles. The approach utilizes the *Mahalanobis distance* calculation [11] to return a residual between normal and current observations. A residual above a dynamic threshold is considered to be the result of a fault. In order to keep the approach computationally light and feasible to be executed online they reduce the dimensions of the sampled data by observing only correlated attributes.

Correlated or redundant data is very useful for fault detection in the domain of autonomous agents. Since no external perception is available for comparison with the agent's own perception, correlation breaks within the agent's perception may suggest an unexpected behavior or a fault. In later work [1], we used this notion as a heuristic that detects sensor faults upon correlation break – the SFDD. In our proposed approach we utilize the SFDD.

Supervised methods are model-free, able to detect unknown faults, and potentially very quick as well. These methods produce offline static fault detection models which are very accurate and also computationally light when applied online. However, supervised methods rely on classified training data and in the domain of physical agents classified training data is usually not available.

For example, Leeke et al. [12] present a methodology for generating efficient error detection mechanisms. Their approach relies on injecting faults into data that is used as a training set for a learning algorithm. The learnt error detection model is of high accuracy and efficiency. A similar approach is presented in [13]. Our proposed approach is similar in concept but instead of relying on a supervised fault injection we suggest the use of an unsupervised method to classify faults that already exist in the unclassified training data. We suggest the use of an unsupervised approach even though it might have some false positives.

For this aim, we suggest the use of the SFDD. The SFDD is an unsupervised approach that very accurately detects faults expressions such as "drift" and "stuck". These types of generic faults expressions appear in a variety of related physical domains. For example, the Advanced Diagnostics and Prognostics Testbed [14] depicts these faults expressions to sensors on an electrical circuit. This testbed is used for the DX competition [15]. Another example is the work of Hashimoto et al. [16] that uses Kalman filters along with kinematical models to detect sensor faults expressions such as "stuck", "abrupt" and "scale" on a mobile robot. In our proposed approach continuous sampled data is transformed into categorical data. The categories are such fault expressions i.e. abrupt, drift, stuck, etc. or a non-fault expression i.e. "ok".

The hybrid approach, proposed in this paper, aims to benefit from all the advantages of the different approaches. Due to the properties of the SFDD the hybrid approach is unsupervised, able to detect unknown faults and does not rely on models that are difficult to construct. Due to the supervised learning process the produced fault detection model is very accurate and computationally light when applied online. Furthermore, the produced model is lighter and more accurate than the original SFDD as we show in the results section.

3. THE HYBRID APPROACH

In this section we describe the problem of fault detection in the domain of autonomous physical agents. Then, we describe the outline of the proposed hybrid approach. We continue with demonstrating how the unsupervised classification of a training set is done. Finally, we describe the learning process.

3.1 Problem Description

Let $A = \{a_1 \dots a_n\}$ be a set of attributes that are monitored in real time e.g. air-speed, heading, pitch, altimeter, etc. and let $V_t = \{v_1 \dots v_n\}$ be the set of values for attributes $\{a_1 \dots a_n\}$ at time t where $v_i \in \mathbb{R}$ is the value assigned to a_i at time t. Past data of m time units of these values $H_m(t) = (V_{t-m}, V_{t-m+1}, V_{t-m+2}, \dots, V_t)$ is also available. $H_m(t)$ is a sliding window (see Figure 1) containing at time t the latest m values of the monitored attributes. In addition, unclassified past online recordings of the physical agent's operations are also available. These recordings can be used as a training set for a machine learning algorithm if classified. We denote this training set as $\mathcal{H} = \{H_{l_1}(e_1), H_{l_2}(e_2), \dots, H_{l_k}(e_k)\}$ where l_i is the length of operation i and e_i is the end time of operation i, thus

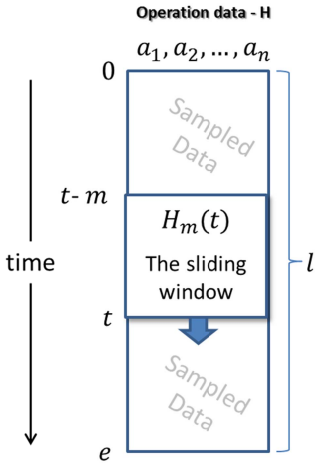

Figure 1: Operation data H & a sliding window

$H_{l_i}(e_i)$ denotes all the values recorded for the monitored attributes in A during operation i.

Given $A, V_t, H_{m(t)}$ and $\mathcal{H}$, the goal is to online recognize whether a fault has occurred to any of the attributes in A. This decision should be made as quickly as possible after a fault has occurred, and should be as accurate as possible. By 'accurate' we mean that a fault detector should have a high detection rate and a low false positive rate.

3.2 The outline of the approach

We introduce a hybrid approach which consists of an offline preprocess and an online fault detection process. The offline preprocess conducts an unsupervised algorithm to classify an unclassified training set. Then, a supervised learning algorithm is used to construct a fault detection model (FDM). This process is described in Section 3.2.1. The FDM is used online to detect faults with greater accuracy. The online process is described in Section 3.2.2. Figure 2 depicts the outline of our approach.

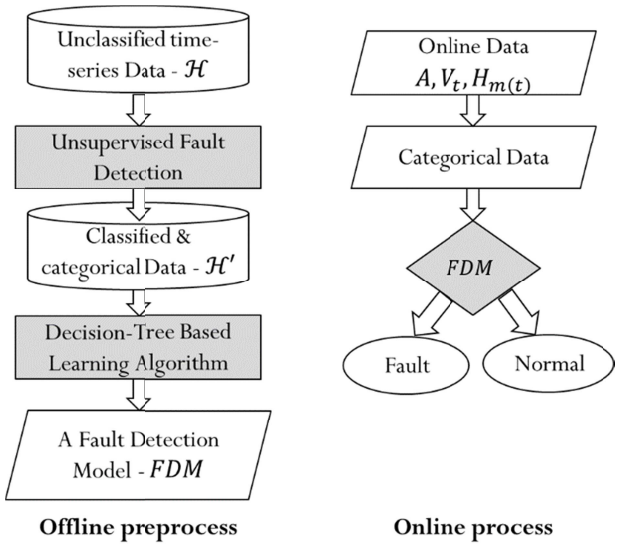

Figure 2: The outline of the hybrid approach

3.2.1 The Offline Preprocess

The unclassified past operations recordings $\mathcal{H}$ are delivered as an input to an unsupervised fault detection algorithm. For this matter we use the SFDD algorithm which is summarized in section 3.3. The SFDD goes throw each operation in a sliding window fashion. For every instance of the sliding window, for each attribute, the SFDD associates the behavior of an attribute (expressed in its time-series data) with a descriptive categorical value. The categories can be "*ok*" for an unsuspected behavior or one of pre-defined suspicious behaviors e.g. "*stuck*", "*drift*".

In addition, each instance of the sliding window is classified according to the SFDD's decision i.e. "*Normal*", or "*Fault*". However, a small portion of normal instances may be misclassified as faults; these are the false positives of the SFDD.

The resulted categorical and classified data is fed into a decision-tree based learning algorithm. For this matter we use the random tree algorithm. The result of this learning process is a fault detection model – FDM. The FDM can be used online, and is more accurate than the original unsupervised fault detection approach as shown in the results section.

3.2.2 The Online Fault Detection

The online input is consumed in a sliding window fashion - $H_{m(t)}$. With each time step t the time-series data in $H_{m(t)}$ is transformed into a categorical data which can be fed into the fault detection model – FDM (resulted by the offline preprocess). The fault detection model classifies the data presented in $H_{m(t)}$ as "*Normal*" or "*Fault*". In section 3.5 we describe the fault detection online process in detail.

3.3 Using the SFDD for offline classification

Each past operation of the autonomous system $H_{l_i}(e_i) \in \mathcal{H}$ is fed into an unsupervised fault detection engine – the SFDD. The SFDD uses a heuristic decision which is based on a prior knowledge of a structural model as well as the online consumed data $H_{m(t)}$ (sliding window) to determine an occurrence of a fault. We chose this approach due to its very low rate of false positives and the high fault detection rate which usually is 1 or very close to 1. Note that any successful unsupervised fault detection approach can be used to classify the unclassified training set.

Since an autonomous agent has no external perception but its own, the SFDD approach relies on correlated attributes to provide the necessary comparison between expected and unexpected behavior of attributes. The approach assumes that strongly correlated attributes behave as redundant to one another and thus can testify of fault occurrence upon correlation break. The domain of autonomous physical agents is characterized by having a rich array of components with interdependencies and redundancy. Therefore, the occurrence of correlated attributes is very likely.

Each attribute is subjected to tests by generic suspicious pattern recognizers. The latest m values of an attribute extracted from $H_{m(t)}$ are tested. For example, we use a *drift* test and a *stuck* test. However, when an attribute shows a suspicious pattern it does not necessarily suggest a fault; it could be a reaction to a normal action of the agent. For example, maintaining altitude may appear as *stuck*, and altitude climbing may appear as a *drift*.

To differentiate between a normal reaction and a fault, the approach uses a heuristic decision (see Figure 4) based on a structural model (see Figure 3). A structural model depicts components dependency e.g. the altimeter is dependent on the static system and the GPS is dependent on the electrical system. Note that a structural model is much easier to construct than an analytical behavioral model.

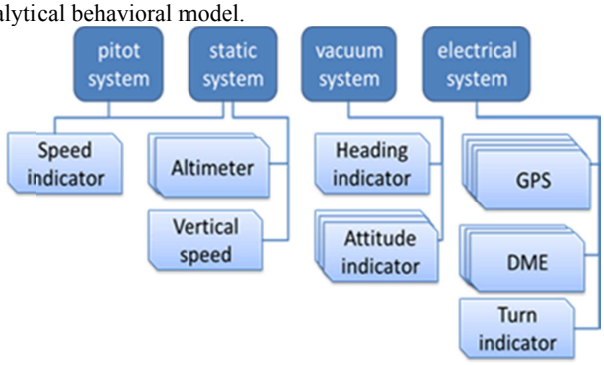

Figure 3: a pratial structural model of a UAV

The heuristic decision compares an attribute that shows a suspicious pattern with an attribute that used to be correlated to it in the previous sliding window. If they do not share component dependency and show different patterns then this is due to a fault. Otherwise, it might be a reaction to a normal action of the agent.

For example, assume the altimeter is suspected for a *drift*, and the GPS indicated altitude was found to be correlated to the

altimeter in the previous instance of the sliding window. These two attributes are dependent on different subsystems i.e. static system and the electrical system respectively (see Figure 3). This fact makes these attributes less likely to be affected by the same fault. If the GPS indicated altitude is also drifting then this is probably due to the altitude climbing action of the UAV and less probable due to a fault that hit both systems. However, if the GPS indicated altitude is not drifting, then this contradiction is probably due to a fault.

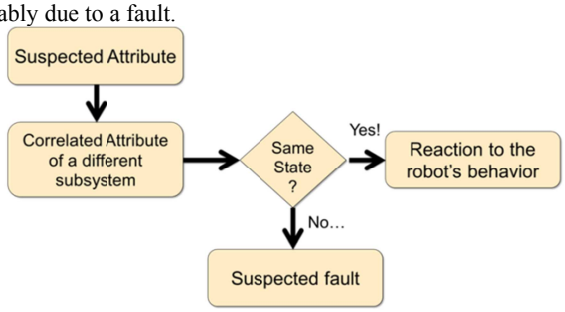

Figure 4: The huristic decision making of the SFDD

The data of each sliding window, $\forall t \in (m \dots e_i)$, $H_{m(t)} \subseteq H_{l_i}(e_i) \in \mathcal{H}$, is transformed into a single output comprised of categorical values that describe the behavior of each attribute i.e. "*ok*", "*drift*", "*stuck*". Each of these outputs is classified according to decision made by the SFDD algorithm i.e. "*Normal*" or "*Fault*".

For example, assume a sliding window of size $m = 4$ containing the values of 3 attributes at a time step t:

Time step	a_1	a_2	a_3
t-4	0.9	1	3
t-3	0.1	2.5	3
t-2	0.05	3	3
t-1	0.15	3.1	3
t	0.05	3.9	3

Assume that the SFDD reported a fault for this time step. The data of the sliding window derives an output of one training sample, where a_1 is categorized as "*ok*", a_2 is categorized as "*drift*", and a_3 is categorized as "*stuck*". In addition, this sample is classified as a fault. Thus, the training sample output for this time step is: *ok, drift, stuck, Fault.* Note that the fact that this sample contains *drift* and *stuck* does not necessarily entail a fault. In some cases it may indicate a normal behavior.

The "*Fault*" classification may be correct or not; it depends on the accuracy level of the unsupervised approach. The next section discusses how this inaccuracy affects the constructed decision tree.

3.4 Learning a fault detection model

We use the SFDD offline to classify a training set. Then, a supervised decision-tree based learning algorithm is applied and a fault detection model is produced– FDM. The FDM is later used online to detect faults. We simply could have used the SFDD online as originally intended, but the learnt FDM has two important advantages. (1) The FDM is computationally lighter than the SFDD since this static model, opposed to the SFDD, has no correlations to calculate, heuristics to apply, or structural models to inquire. Therefore, the FDM is more suitable for the domain of autonomous physical agents. (2) More importantly, the resulted FDM is significantly more accurate than the original SFDD.

The SFDD, though highly accurate, is not perfect. A small degree of false positives (usually less than 3% on tested domains) yields misclassified normal instances in the training data. However, decision trees, as other supervised learning algorithms, have some tolerance towards misclassified instances in the training set.

During the model construction the algorithm grows a decision tree. In each step, the algorithm chooses an attribute that best classifies the remaining data by selecting the attribute with the highest information gain ($IG(a_i)$). The information gain is determined by the entropy of the attribute which is affected by the ratio between the normal and faulty instances. The lower the faulty instances are, the higher the information gain is. If a portion of instances are misclassified as "*Fault*" then this might reduce the information gain of an attribute. However, if the degree of reduction is small enough then the construction of the tree may be less affected.

The degree of information gain reduction due to falsely classified training instances is dependent on several factors.
Let

- S be the training set.
- α_{ij} be $|S_{a_i=v_j \wedge Class=fault}|$ the number of instances in which attribute a_i has the value v_j and the classification is "*Fault*" when all instances are classified correctly by an oracle.
- β_{ij} be $|S_{a_i=v_j \wedge Class=normal}|$ the number of instances in which attribute a_i has the value v_j and the classification is "*Normal*" when all instances are classified correctly by an oracle.
- x_{ij} be the number of instances from $\left|S_{a_i=v_j}\right|$, that were falsely classified as "*Fault*" due to false positives of the unsupervised approach (note that $\left|S_{a_i=v_j}\right| = \alpha_{ij} + \beta_{ij}$).

The effect of the misclassified x_{ij} instances on the information gain of attribute a_i is the new (affected) information gain $IG_{x_{ij}}(a_i)$ minus the original information gain $IG(a_i)$:

$$f(x_{ij}) = IG_{x_{ij}}(a_i) - IG(a_i)$$

$$f(x_{ij}) = H(S) - \sum_{v_j} \frac{\left|S_{a_i=v_j}\right|}{|S|} H_{x_{ij}}\left(S_{a_i=v_j}\right) - H(S) + \sum_{v_j} \frac{\left|S_{a_i=v_j}\right|}{|S|} H\left(S_{a_i=v_j}\right)$$

Where H is the entropy function and $H_{x_{ij}}$ is the entropy affected by falsely classified instances. For given x_{ij} falsely classified instances in $S_{a_i=v_j}$ the affected entropy is:

$H_{x_{ij}}\left(S_{a_i=v_j}\right) = -\frac{\alpha_{ij}+x_{ij}}{\alpha_{ij}+\beta_{ij}} \log \frac{\alpha_{ij}+x_{ij}}{\alpha_{ij}+\beta_{ij}} - \frac{\beta_{ij}-x_{ij}}{\alpha_{ij}+\beta_{ij}} \log \frac{\beta_{ij}-x_{ij}}{\alpha_{ij}+\beta_{ij}}$. Note that x_{ij} instances that are falsely added to α_{ij} are in the expense of β_{ij}. After some algebra $f(x_{ij})$ can be presented for given x_{ij} falsely classified instances in $S_{a_i=v_j}$ as:

$$f(x_{ij}) = \frac{1}{|S|}\left(\alpha_{ij} \log \frac{\alpha_{ij}+x_{ij}}{\alpha_{ij}} + \beta_{ij} \log \frac{\beta_{ij}-x_{ij}}{\beta_{ij}} + x_{ij} \log \frac{\alpha_{ij}+x_{ij}}{\beta_{ij}-x_{ij}}\right)$$

We can see that when $x_{ij} = 0$ then $f(x_{ij}) = 0$. When x_{ij} grows, $f(x_{ij})$ decreases. This affect is depicted in Figure 5. Figure 5 illustrates an example for this effect on the information gain where $|S| = 300, \alpha_{ij} = 20, \beta_{ij} = 100$.

As x_{ij} grows, the information gain decreases. When we reach an equal number of "*fault*" and "*normal*" instances ($x_{ij} = 40$) the effect is at its deepest point. As x_{ij} continues to grow a mirror effect occurs as the instances are heading back to the original partition - only with the opposite classification ($\alpha_{ij} + x_{ij} = 100$, $\beta_{ij} - x_{ij} = 20$).

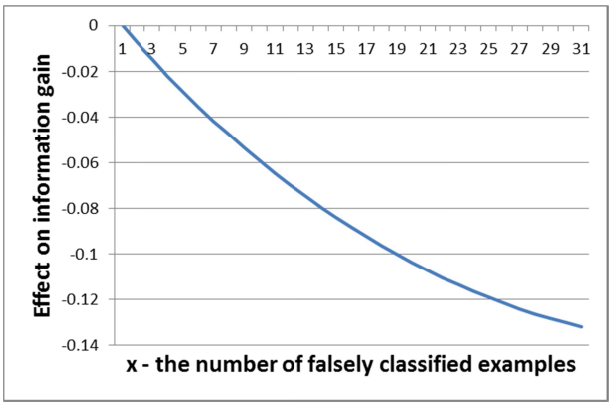

Figure 5: the negative effect on the information gain

In the case of the SFDD classification of the training set, the false positive rate is usually less than 3%. A low rate of false positives in the training data has very little effect on the decision tree construction. As a result the supervised learnt FDM improves the accuracy of the SFDD when compared to the intended online unsupervised original use of the SFDD. The 3% of false positives affect differently attributes with high information gain and attributes with low information gain.

Case 1: the effect on attributes with high information gain

In the particular example above, a_i has a high information gain to begin with (0.74), and when $x_{ij} = 3$ (i.e. 3% of $\beta_{ij} = 100$) then $f(x_{ij}) = -0.021$. If $IG(a_i)$ is originally greater than the information gain of another attribute $IG(a_j)$ it is still possible that $IG_{x_{ij}}(a_i) = IG(a_i) + f(x_{ij}) > IG(a_j)$. In this case a_i is still going to be selected and the tree construction is **unaffected** by the amount of falsely classified instances. In other cases, where attributes with close information gain exist the following can be true: $IG(a_i) > IG(a_j) \wedge IG_{x_{ij}}(a_i) < IG(a_j)$. In this case the tree construction is changed and attribute a_j is selected before a_i. However, this only affects the order of which the data is sliced, and not the decision of the tree.

When no other attributes can be selected the decision tree decides on a classification according to the majority of instances. In this particular example the 3 falsely classified instances did not change the majority class of "*Normal*". Therefore, the decision tree makes the same decision as a tree that is constructed from a correctly classified training data. This explains the greater accuracy of the supervised approach.

Case 2: the effect on attributes with low information gain

Since attributes with high information gain are selected first (to the top of the tree), it is quite possible that attributes with low information gain are selected last and thus influence the decision tree. Consider a case where $\alpha_{ij} = 59, \beta_{ij} = 61$ and $x_{ij} = 2$. The two falsely classified instances affect the majority and the tree decision is going to be "*Fault*" rather than "*Normal*" as the tree that is constructed from a correct training set would have decided. This explains why there are still some false positives to the supervised FDM.

To summarize, the FDM is only slightly affected by the false positives of the unsupervised approach that is used to classify the training data. In rare cases a small number of misclassified instances change the majority class of an attribute with low information gain that determines the decision of the tree. In other cases the tree decision in unaffected. Therefore, the resulted FDM is more accurate than the SFDD. Moreover, the learnt FDM can potentially have a false positive rate that tends to 0 when a highly accurate unsupervised approach is used to classify the training set.

3.5 Using the FDM online

When given a new online input, we consume it in a sliding window fashion $H_m(t)$. We transform the time-series data in $H_m(t)$ to an output of categorical data in the same manner of the SFDD. Only when the new output is different than the previous one i.e. some attribute changed its state, it is fed as an input into the offline-learnt fault detection model - FDM. The FDM decides (online) whether or not it is a fault.

For example, consider a static-system failure. One of the expressions of this failure is the frozen value of the altimeter. In $H_m(t)$ the values of the altimeter attribute are all equal, while the values of the GPS indicated altitude attribute diverse. The equal values of the altimeter are recognized as a suspicious pattern by the stuck-pattern detector. Therefore, the corresponding categorical output to $H_m(t)$ has the value "*stuck*" for the altimeter attribute and the value "*ok*" for the GPS indicated altitude attribute (other attributes also get their own categorical values) . This output is fed into the FDM that decides whether or not these values express a fault. It is possible that the next categorical output that corresponds to $H_m(t+1)$ will not be different than its predecessor. In this case, the FDM is not triggered again; only if at least one of the attributes changed its state e.g. from "*ok*" to "*stuck*" as the altimeter, then the FDM is triggered.

To summarize our hybrid approach: in the first stage we use unsupervised fault detection (SFDD) to classify unclassified training set. The suspicious pattern detectors of the SFDD approach are used to transform the time-series data in each sliding window into categorical data. Each sample is classified according to the unsupervised decision i.e. "*Normal*" or "*Fault*". Then, we apply a decision tree based learning algorithm on the training data and produce a fault detection model - FDM that can be applied online. The online process uses the same suspicious pattern detectors to produce categorical data for each sliding window. The categorical data is fed into the FDM which was learnt offline. The FDM makes a choice whether or not the online input is an expression of a fault.

4. EXPERIMENTAL SETUP

To examine our approach we present tests which examine the accuracy of our hybrid fault detection approach. We use three domains to test the fault detection accuracy. The first (see Figure 6) is a high fidelity flight simulator [17] the second is a commercial UAV, and the third is a laboratory robot *Robtican1* [18] (see Figure 8). We expect our proposed hybrid approach to be more accurate in fault detection than the original unsupervised SFDD approach.

Figure 6: FlightGear screenshot

FlightGear domain: *FlightGear* (see Figure 6) is an open source flight simulator designed for research purpose and is used for a variety of research topics. FlightGear has built-in realistically simulated instrumental and system faults. For example, if the vacuum system fails, the HSI gyros spin down

slowly with a corresponding degradation in response as well as a slowly increasing bias/error.

We recorded 32 flights. Each flight had duration of 5 minutes, and included a take-off and left and right turns. 23 attributes were sampled in 4Hz. Each flight was injected with a different type of fault. Each fault had duration of 35 seconds and was injected twice to the same flight at random times. In total, we tested 11 different types of instrumental and system failures. In total, the test set contains 25,977 instances out of which 5,880 are expressions of faults.

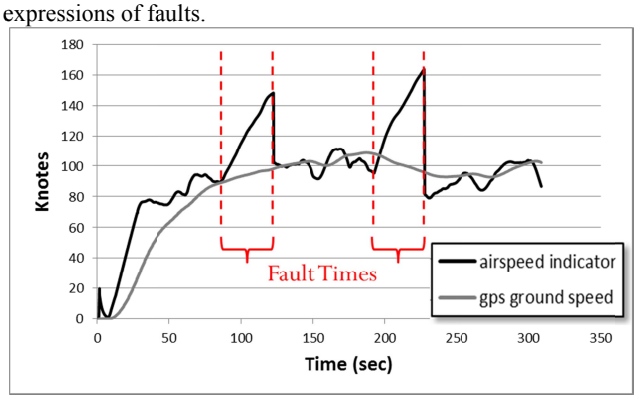

Figure 7: pitot system failure

Figure 7 illustrates a flight with an injected *pitot system* failure. A pitot system failure is expressed by the drift of the dependent airspeed indicator as shown in the black curve. However, this could only be considered as an outlier with respect to something else. The gray line illustrates the ground speed measured by the GPS. If these two attributes were deemed to be correlated then the fault would be detected by the SFDD approach since they are dependent on different subsystems and show different behaviors.

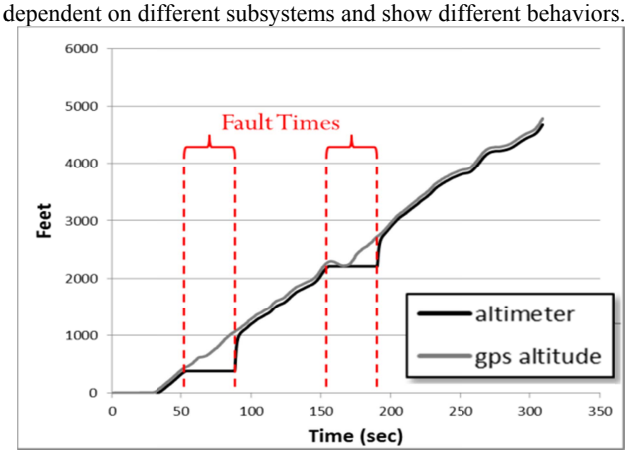

Figure 9: static system failure

Another example of an injected system fault is the *static system* failure. This fault causes the airspeed indicator to drift down to 0 while the GPS ground speed remains "*ok*". In addition, this fault causes the altimeter to be stuck on the same value while the GPS indicated altitude is "drift" (increasing) as illustrated in Figure 9. The altimeter and the GPS indicated altitude are redundant to each other and hence are usually correlated. Since these two attributes are dependent on different subsystems then the SFDD can detect such a fault.

The SFDD applied on the training set achieved a detection rate of 1 (all faults were detected). The resulted categorical and classified data was used as a training set.

Commercial UAV domain: The real UAV domain consists of 6 recorded real flights of a commercial UAV. 53 attributes were sampled in 10Hz. The attributes consists of telemetry, inertial, engine and servos data. Flights duration varies from 37 to 71 minutes. The UAV manufacture injected a synthetic fault to two of the flights. The first scenario is a value that drifts down to zero. The second scenario is a value that remains frozen (stuck). The detection of these two faults were challenging for the manufacture since in both scenarios the values are in normal range. These two flights were used as a test set. The remaining four flights were used as a training set where into two flights we injected similar synthetic faults. In total, the test set contains 65,741 instances out of which 1,593 are expression of faults.

Laboratory robot domain: *Robotican1* is a laboratory robot that has 2 wheels, 3 sonar range detectors in the front, and 3 infrared range detectors which are located right above the sonars, making the sonars and infrareds redundant systems to one another. This redundancy reflects real world domains such as unmanned vehicles. In addition, the Robotican has 5 degrees of freedom arm. Each joint is held by two electrical engines. These engines provide a sensed reading of the voltage applied by their action.

We devised 10 different scenarios that included different injected faults while the robot performed different tasks. Faults were injected to each type of sensor (motor voltage, infrared and sonar). The injected faults to the sensors were of type *stuck* or *drift*. These faults were injected to one or more sensors in different time intervals. 15 attributes were sampled in 8Hz. Scenarios duration lasted 10 seconds where the last 5 seconds expressed a fault. 4 scenarios were used as an unclassified training set and the other 6 were used as a test set. Note that in this domain, the training set did not cover all the faults included in the test set. In total, the test set contains 480 instances out of which 240 are expression of faults.

Figure 8: Robotican 1

For the supervised learning we have experimented with several decision tree algorithms: ID3, J48, and a Random Tree [19]. As expected the Random Tree performed better and its results on the three domains are shown in the next section.

5. RESULTS

Figure 10 illustrates the average false positive rate of the hybrid vs. the unsupervised approach (SFDD), taken over the 21 test flights of the FlightGear domain, using a sliding window size of 250 time steps. The hybrid approach significantly improved the false positive of the unsupervised algorithm.

To demonstrate the degree of reduction of the false positive rate by the suggested hybrid approach we used different sizes of sliding windows during the offline training phase. Smaller sizes create more opportunities for reports and thus more opportunities for false positives.

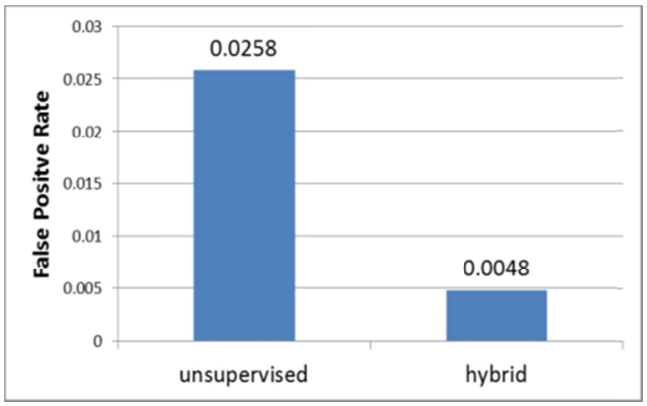

Figure 10: FP rate, unsupervised vs. hybrid, flightgear

Figure 11 illustrates the degree of reduction in the average false positive rate over the 21 test flights in the FlightGear domain. Note that the false positive rate is in logarithmic scale. We can see that with each size of sliding window the false positive rate of the hybrid approach is significantly lower than the SFDD unsupervised approach.

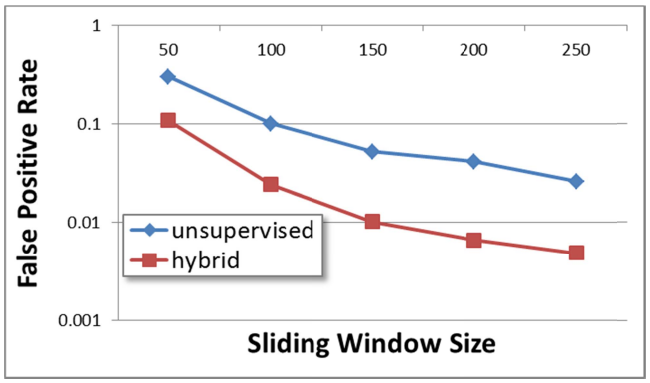

Figure 11: FP rate vs. sliding window size

The different parameters used by the unsupervised approach during the offline phase can be viewed as different unsupervised approaches; each with its own rate of false positives. The hybrid approach contributes to the reduction of false positive rate for each of these unsupervised approaches. Moreover, as the false positive rate of the unsupervised approach is getting lower, thus the false positive rate of the hybrid approach tends to 0.

Satisfied by the very low rate of false positives, we decreased the sliding window size used **online**. It is suggestible to use a smaller m for $H_m(t)$ when classifying an online input than the m used during the offline training. This increases the number of reports, and since the false alarm rate is very low, we can tolerate an increase of false positives in return for a higher true positives rate.

Figure 12 illustrates the ROC of false alarm rates and the detection rates of the unsupervised approach verses the hybrid approach under the influence of a changing size of the online sliding window (62sec – 47sec). Note that scale of Figure 12 zooms in on high detection rate (close to 1) and low false alarm rate (close to 0). The added of false positives to the hybrid approach is of little significance while the effect on the unsupervised approach is apparent.

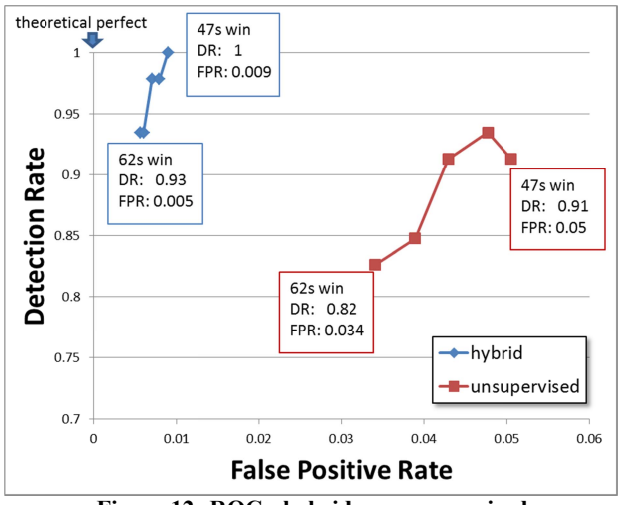

Figure 12: ROC - hybrid vs. unsupervised

In addition, the detection rate of the hybrid approach is getting higher as the size of the sliding window decreases. This is explained by the fact that a smaller size of a sliding window increases the frequency of state changes and hence the total amount of reports. Therefore, there is a greater chance for detection as well as some false positives. The hybrid approach gets a lower rate of false alarms and a higher rate of fault detection than the original unsupervised approach.

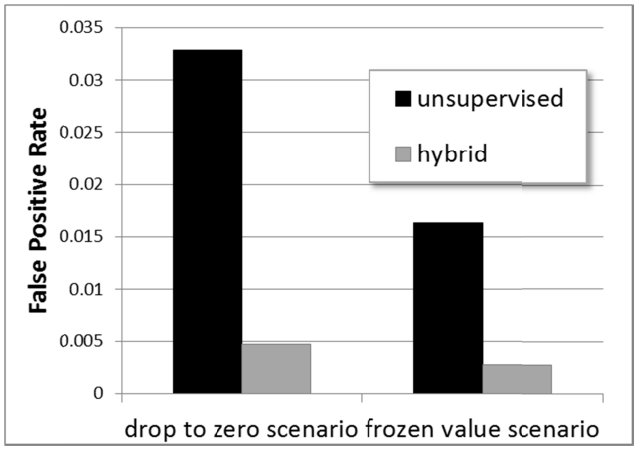

Figure 13: UAV, unsupervised vs. hybrid

In the UAV domain the hybrid approach keeps a similar trend. In the two examined scenarios, both the hybrid and unsupervised approaches had a detection rate of 1. However, the hybrid approach had a significantly lower false positive rate than the unsupervised approach as Figure 13 shows.

In the *Robotican1* domain, even though the training set did not include all possible faults that were included in the test set, the detection rate of the learnt fault detection model was 1. Being online and unsupervised, it is not surprising that the unsupervised approach also scored a detection rate of 1 on the test set. However, it is interesting to note that the offline learnt *FDM* of the hybrid was able to generalize the heuristic decision of the unsupervised approach such that unseen faults were detected.

The average false alarm rate of the unsupervised approach on the 6 tested scenarios was 0.067 while the hybrid approach scored 0.041. Again, the hybrid approach reduced the false positive rate.

6. CONCLUSIONS and DISCUSSION

In this paper we described a hybrid approach for fault detection in the domain of autonomous physical agents that is unsupervised, able to detect unknown faults, use an easy to construct model, and also computationally very light and thus can detect faults very quickly online.

The approach uses an accurate unsupervised method to offline classify an unclassified training set, applies a supervised learning algorithm, and produces a fault detection model that is computationally lighter and is more accurate than the original unsupervised method when applied online. We have explained the causes for the improvement in accuracy and showed satisfying results in three different domains – both real-world and simulated.

The offline step classifies the data with an unsupervised approach. An alternative approach for classifying the data is a general clustering algorithm e.g. K-means where k=2. However, an unsupervised fault detection approach is more specific to the fault detection problem and thus expected to be more accurate than the general clustering algorithm. We showed that the higher the accuracy of the unsupervised approach, the closer the false positive rate of the hybrid approach tends to 0.

We chose to demonstrate how a hybrid approach extends the state of the art with the use of, and a comparison to the SFDD approach since it showed a high detection rate and a very low false positive rate. Any other highly accurate unsupervised approach could have been used for classifying the unclassified training set. The high detection rate is very important since all faults should be classified as such.

The proposed approach is unsupervised since the starting point is with unclassified training set. The SFDD uses a structural model (dependency associations) and thus is model based. However, constructing such a model is easier than the typically suggested analytical behavioral models. Both the SFDD and the online phase of the hybrid approach use suspicious pattern recognizers to categorize the behavior of attributes. These pattern recognizers are very generic and are suitable for a large variety of physical systems; the same pattern recognizers were used in all the tested domains.

The learnt FDM generalized the original heuristic decision of the unsupervised approach. The model is independent of heavy online computations such as correlation calculations and thus is computationally lighter. In addition, the FDM is less susceptible to false positives than the original unsupervised approach.

We showed that the use of a static model i.e. learnt offline, is very accurate. It is quite possible that models that are learnt online in a supervised manner would be even more accurate when supplied with enough training instances. However, these online computations might be heavy for the agent and not feasible to detect faults quickly enough.

For future work we plan to extend the hybrid approach to multiclass supervised learning. The classification options will be the different diagnoses for the fault. We hope that the learnt fault detection and diagnosis model will provide an accurate and minimal diagnosis as well.

7. REFERENCES

[1] E. Khalastchi, M. Kalech and L. Rokach, "Sensor Fault Detection and Diagnosis for Autonomous Systems," in *the 12th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2013)*, Saint Paul, Minnesota, USA, 2013.

[2] G. Steinbauer, "A Survey about Faults of Robots used in RoboCup," in *RoboCup 2012: Robot Soccer World Cup XVI*, pringer Berlin Heidelberg, 2013, pp. 344-355.

[3] http://www.robocup.org/, "RoboCup competition".

[4] R. Akerkar and P. Sajja, Knowledge-based systems, Jones & Bartlett Publishers, 2010.

[5] R. Isermann, "Model-based fault-detection and diagnosis--status and applications," *Annual Reviews in control,* pp. 71-85, 2005.

[6] G. Steinbauer and F. Wotawa, "Detecting and locating faults in the control software of autonomous mobile robots," in *the 19th International Joint Conference on Artificial Intelligence (IJCAI-05)*, 2005.

[7] G. Steinbauer and F. Wotawa, "On the Way to Automated Belief Repair for Autonomous Robots," in *the 21st International Workshop on Principles of Diagnosis (DX-10)*, 2010.

[8] P. Struss and S. Dobi, "Automated Functional Safety Analysis of Vehicles Based on Qualitative Behavior Models and Spatial Representations," in *the 24th International Workshop on Principles of Diagnosis (DX-13)*, Jerusalem, 2013.

[9] V. Chandola, A. Banerjee and V. Kumar, "Anomaly detection: A survey," *ACM Computing Surveys (CSUR),* vol. 41, pp. 1-58, 2009.

[10] E. Khalastchi, G. A. Kaminka, M. Kalech and R. Lin, "Online anomaly detection in unmanned vehicles," in *the 10th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2011)*, Taipei, Taiwan, 2011.

[11] P. C. Mahalanobis, "On the generalized distance in statistics," in *the National Institute of Sciences*, 1936.

[12] M. Leeke, S. Arif, A. Jhumka and S. S. Anand, "A methodology for the generation of efficient error detection mechanisms," in *the 41st International Conference on Dependable Systems & Networks (DSN)*, 2011.

[13] A. L. Christensen, R. O'Grady, M. Birattari and M. Dorigo, "Fault detection in autonomous robots based on fault injection and learning," *Autonomous Robots,* vol. 24, pp. 49-67, 2008.

[14] S. Poll, A. Patterson-Hine, J. Camisa, D. Garcia, D. Hall, C. Lee, O. J. Mengshoel, C. Neukom, D. Nishikawa and J. Ossenfort, "Advanced diagnostics and prognostics testbed," in *Proceedings of the 18th International Workshop on Principles of Diagnosis (DX-07)*, 2007.

[15] S. Poll, J. de Kleer, R. Abreu, M. Daigle, A. Feldman, er, D. Garcia, A. Gonzalez-Sanchez, T. Kurtoglu, S. Narasimhan and A. Sweet, "Third International Diagnostic Competition--DXC'11," in *the 22nd International Workshop on Principles of Diagnosis (DX-11)*, 2011.

[16] M. Hashimoto, H. Kawashima and F. Oba, "A multi-model based fault detection and diagnosis of internal sensors for mobile robot," in *International Conference on Intelligent Robots and Systems (IROS 2003)*, 2003.

[17] A. R. Perry, "The flightgear flight simulator," in *USENIX Annual Technical Conference*, Boston, MA, 2004.

[18] Robotican1, "http://www.robotican.net/".

[19] L. Reiman, "Random forests," *Machine learning,* vol. 45, pp. 5-32, 2001.