

# Design and Performance Improvements for Fault Detection in Tightly-Coupled Multi-Robot Team Tasks

Xingyan Li and Lynne E. Parker

*Distributed Intelligence Laboratory, Department of Electrical Engineering and Computer Science*

*The University of Tennessee, Knoxville, TN 37996-3450*

*Email: {li, parker}@eecs.utk.edu*

## Abstract

*This paper presents our current work to improve the design and performance of our previous work: SAFDetection, a sensor analysis based fault detection approach that is used to monitor tightly-coupled multi-robot team tasks. We improve this prior approach in three aspects. First, we show how Principal Components Analysis (PCA) can be used to automatically generate a small number of sensor features that should be used during the learning of the model of normal operation. Second, we implement three different algorithms for clustering sensor data in SAFDetection and compare their fault detection rates on physical robot team tasks, to determine the best technique for clustering sensor data while learning the model of normal team task operation. A third improvement we present is to modify the state transition probability from constant to a time-variant variable to describe the operation of the robot system more accurately. Our results show that a PCA feature selection approach, combined with a soft classification technique and time-varying transition probabilities, yields the best fault detection results.*

## 1. Introduction

Due to the increasing needs for robotic applications in various challenging tasks, it is necessary for robots to monitor their performance so that the deviation from the expected behavior, which we call a *fault*, can be promptly detected. In our previous work [13], we presented the basic idea and structure of SAFDetection — a fault detection approach which is based only on the monitoring of robot sensor values. This paper presents three main improvements that we have addressed to SAFDetection: 1) Principal Components Analysis (PCA), 2) a probabilistic clustering algorithm, and 3) time-variant state transition probabilities. We believe that the results of our studies are general enough to be applied to many other applications in robotics.

The SAFDetection approach detects faults based on the sensor data from a robot or multi-robot system. Such a system usually consists of a large number of sensors that can be characterized with a huge number of possible features. The computational demands of a model learning process are intractable if the approach requires large numbers of features. Thus, the objective is to prune down the features to the smallest possible number that will achieve the desired

accuracy. In our previous work [13], feature selection was implemented manually by an experienced system designer. In this paper, we present how Principal Components Analysis (PCA) can be used to significantly reduce the dimension of the feature vector, and to automatically select the correct features for model learning. This ability enables SAFDetection to be used much more easily in new robot applications, as it requires little prior knowledge of the system.

Our SAFDetection approach treats the robot system as a black box and determines the current state of the robot system by clustering the sensor data. In our previous work on SAFDetection [13], we made use of the Fuzzy C-Means clustering algorithm. However, an open question was the degree to which the particular clustering algorithm used affected the ability of the system to accurately detect faults. In this paper, we compare the performance of crisp, fuzzy, and probabilistic clustering methods within the SAFDetection approach, determining the affect on the fault detection rate when using different clustering methods.

To learn the normal operation of a robot system, SAFDetection builds a probabilistic state transition diagram from sensor data generated during normal task execution by physical robots. In our previous work [13], the transition probabilities between states are constant average values learned from the training data. However, further experimentation shows that the transition probabilities between states usually vary over time due to the complex operation of the robot system. Therefore, the traditional Markov model based state transition diagram cannot describe the operation of the robot system accurately. This paper presents our current work that incorporates a time factor into the state transition diagram.

The remainder of the paper is organized as follows. We present related work, including a brief description of our previous SAFDetection work in Section 2, followed by details of the PCA, clustering techniques, and time-variant state transition diagram in Section 3. In Section 4, we analyze the results of using these design and performance improvement methods in a physical robot implementation of a box pushing experiment. We present our conclusions in Section 5.

## 2. Related work

Much prior work has been done in the area of robot fault detection. The most popular method for providing

fault detection in robot systems is based on motion control [7], [14], [11]. Other widely used computer fault detection methods include voting based modular redundancy [16], [3], analytical redundancy [12], [8], [5] and particle filter techniques [4], [18], [2].

The work presented in this paper is based on our SAFDection approach, which has been presented in [13]. The SAFDection approach is a training-classification based method for fault detection in tightly coupled multi-robot team tasks. In the training stage, a history of sensor data (i.e., training data) during the normal operation of physical robots is clustered into different states through the use of a clustering algorithm. A state transition diagram that represents the probabilistic transitions between these states is then learned from the training data. In the online fault detection classification stage, the on-line sensor data is clustered into states using the same clustering algorithm. The faults can be detected by comparing the current state, and state transitions, with the learned state transition model representing normal behavior.

Our current work offers several design and performance improvement methods into the SAFDection approach. These methods improve the fault detection rate of the approach, as well as simplify the application of this approach to new multi-robot team tasks. The comparisons and advantages will be detailed in later sections.

### 3. Design and performance improvements

In this section, we give a detailed description of three major design and performance improvements we incorporated into the SAFDection approach. These three improvements are: (1) making use of Principal Components Analysis (PCA) to automatically select features for learning; (2) the use of probabilistic clustering methods, which result in higher accuracy for fault detection; and (3) making use of a time-variant state transition diagram, which can describe the operation of the robot system more accurately.

#### 3.1. Automatic feature selection using PCA

In SAFDection, the only information used for building the model of normal operation is the robot sensor data. To reduce the high-dimensional sensor data, relevant features must be selected and monitored. An experienced human system designer who is highly familiar with the robot tasks could likely select the relevant sensor features manually; this is the approach we used to implement feature selection in our previous work. Unfortunately, in most robot tasks, it will be difficult to determine the relevant sensor features manually. Thus, we prefer techniques that allow us to automatically select the best sensor features for the robot team task.

Principal Component Analysis (PCA) [9] is a common method used to reduce the dimensionality of an input space without losing a significant amount of information (variation). Here we use it to automatically determine the best features for representing the state of a robot system during the performance of a repetitive task.

The transformation to principal component space can be thought of as projecting the  $m$ -dimension data set  $X$  onto the new lower  $k$ -dimension coordinate system  $P$ , resulting in scores  $T$  and residue  $E$ . The new  $k$  variables, called principal components or features, are globally uncorrelated, while most of the variation of the original data set  $X$  is preserved:

$$T = X * P + E \quad (1)$$

Various guidelines [15] have been developed to find the proper reduced space dimension,  $k$ . In PCA, the principal components are the eigenvectors of the covariance matrix of  $X$ ; the first eigenvector corresponds to the largest eigenvalue. Since the eigenvalues are proportional to the amount of variance (information) represented by the corresponding principal component, we limit the value  $k$  by selecting the first few principal components that represent most (for example, over 80%) of the information.

#### 3.2. Alternative data clustering techniques

In our prior SAFDection work, the Fuzzy C-Means (FCM) clustering algorithm was used to learn the states of the robot system. In continuing work, we have implemented two alternative methods for clustering the sensor data: K-means clustering and Soft K-means clustering.

K-means clustering [6] is one of the simplest unsupervised learning algorithms. It aims at minimizing the following objective function:

$$J = \sum_{j=1}^k \sum_{x_i \in S_j} (x_i - c_j)^2 \quad (2)$$

where  $k$  is the total number clusters,  $x_i$  is the  $i^{th}$  data point,  $S_j$  is the  $j^{th}$  cluster and  $c_j$  is the centroid of  $S_j$ . K-means is a fast, simple and efficient clustering algorithm. However, it is crisp, meaning that data points are assigned to exactly one cluster and all points assigned to a cluster are equal in that cluster. This clustering method may encounter problems when dealing with clusters that have some overlap. Additionally, when applied to robot applications, it is sensitive to noisy or partial data.

Soft K-means clustering [10] is a revised version of the K-means clustering method with a stiffness parameter,  $\beta$ . In soft K-means clustering, each data point  $x_i$  is given a soft degree of membership to each of the centroids. This characteristic of soft clustering helps deal with overlapping clusters, and is advantageous when dealing with noisy or partial data that is typical of robot applications. The soft degree of assignment of data  $x_i$  to cluster  $c_k$  is labeled  $r_i^k$ , and defined as:

$$r_i^k = \frac{\exp(-\beta * d(c_k, x_i))}{\sum_j \exp(-\beta * d(c_j, x_i))} \quad (3)$$

where  $d(c_k, x_i)$  is a distance measurement between data pattern  $x_i$  and cluster centroid  $c_k$ . The centroid of cluster  $c_k$  is then calculated as:

$$c_k = \frac{\sum_{i=1}^n r_i^k x_i}{\sum_{i=1}^n r_i^k} \quad (4)$$

A third type of clustering we studied is Fuzzy C-means clustering (FCM) [1], which has been used in many areas. This is the technique we previously implemented and reported in our SAFDetection approach [13]. In the FCM algorithm, the algorithm minimizes the objective function  $J$ :

$$J = \sum_{k=1}^n \sum_{i=1}^c u_{ik}^m d^2(x_k, v_i) \quad (5)$$

where  $x_k$  is the  $k^{\text{th}}$   $p$ -dimensional data vector,  $v_i$  is the prototype of the center of cluster  $i$ ,  $u_{ik}$  is the degree of membership of  $x_k$  in the  $i^{\text{th}}$  cluster,  $m$  is a weighting exponent on each fuzzy membership,  $d(x_k, v_i)$  is a distance measurement between data pattern  $x_k$  and cluster center  $v_i$ ,  $n$  is the number of data, and  $c$  is the number of clusters. The objective function  $J(U, V)$  is minimized iteratively as follows:

$$u_{ik} = \frac{1}{1 + \sum_{j=1}^c (d_{ik}/d_{ij})^{\frac{2}{m-1}}} \quad (6)$$

$$v_i = \frac{\sum_{k=1}^n u_{ik}^m x_k}{\sum_{k=1}^n u_{ik}^m} \quad (7)$$

where  $\forall i$   $u_{ik}$  satisfies:  $u_{ik} \in [0, 1]$ ,  $\forall k \sum_{i=1}^c u_{ik} = 1$  and  $0 < \sum_{k=1}^n u_{ik} < n$ .

The results of applying these clustering methods in SAFDetection to physical robot experiments are analyzed in the next section.

### 3.3. Time-variant state transition diagram

In SAFDetection, the state transition diagram learned from history data models the normal operation of the robot system. In our previous work, the learned state transition diagram recorded states and the transition probabilities between each pair of states. In addition, it also included the mean and standard deviation values of the time duration of the system in each state (i.e., before the system transits to another state). The state transition probabilities in our previous work are constant, and represent only the average values. However, in realistic applications, the transition probabilities between states usually vary with time. To describe transition changes as a function of time, a time based variable, instead of a constant, should be used to represent the state transition probabilities between states.

We have implemented a method to build a time-varying state transition diagram. In this method, for each state  $S$ , its transition probability to another state  $S'$  is a time-variant variable; in other words, the transition probability from state  $S$  to state  $S'$  changes with time  $t$ , which represents the time duration of the system in state  $S$ . In some cases, an equation representing a function of time  $t$  can be generated to describe this time-variant variable [17]. However, we have found it difficult to identify a general method for building such equations because of the complexity of robot operations during multi-robot tasks. Thus, in our current work, we record and calculate the average transition probability from  $S$  to  $S'$  for each time step  $t$ . Here,  $t$  is the relative time of duration of the system in state  $S$  before

it transfers to another state. In this approach, we represent the lack of a state change as a state transition back to the same state. This method for building a time-variant state transition diagram is outlined in the following steps:

- 1) For each training data, assign it to the learned states using a clustering method.
- 2) For each time step, record the time duration of the system in the state and any state transitions in this current time step, including a transition back to the same step.
- 3) For each state, calculate and record its average transition probabilities to all the states (including itself), for each possible time step (the maximum of the time steps is the longest time the system remained in that state).

With this time-variant state transition diagram, we can describe the state transition probabilities for each time step, which is much more accurate compared to constant probability transitions.

## 4. Experimental results

We have implemented the proposed design and performance improvements in our SAFDetection approach on a physical robot team performing a cooperative box pushing task. In this section, we present the results of using PCA, different clustering algorithms, and the time-variant state transition diagram.

### 4.1. Box pushing task

In the box pushing task, two robots are brought together to cooperatively push a box to a goal position, which in our case is indicated by a red object clearly visible to the robots. During the pushing, the robots need to adjust their speed and pushing direction to avoid losing the box; they also may need to either change the pushing direction fairly significantly, or to idle for a while to coordinate with their teammate's pushing activities. In our box pushing experiments, we use two Pioneer mobile robots equipped with laser, sonar ring, color camera, and motors with encoders. Each of these devices provides sensor data that can be monitored for our application. In our prior work [13], we showed how our SAFDetection approach works successfully to detect a number of faults during execution of this task on physical robots. The results in this section are presented in comparison to this prior SAFDetection implementation.

### 4.2. Feature selection

Principal Components Analysis is a data compression technique that puts redundant (correlated) information into separate variables. We can assume the features returned from one sensor are correlated and that PCA can be performed on those features. For example, for the camera, since we focus on the features of a red block (representing a goal position), we record the "left", "right", "top" and "bottom" edge positions of the red block in the camera

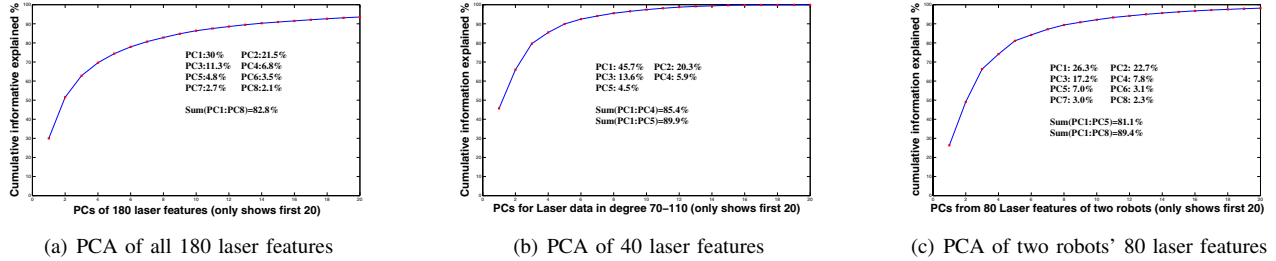


Fig. 1. PCA results for laser features.

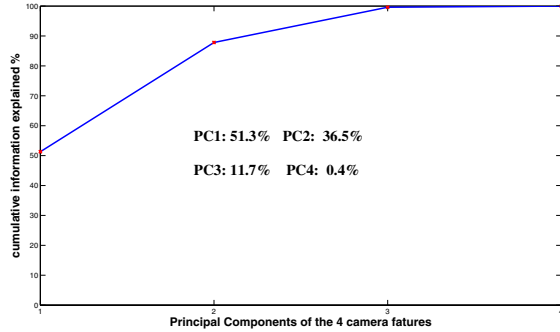


Fig. 2. The PCA results of camera sensor.

image as possible features. Table I shows the correlation among those four features.

TABLE I  
THE CORRELATION COEFFICIENT FOR CAMERA FEATURES

Correlation Coefficient	Left	Right	Top	Bottom
Left	1	0.925	-0.013	-0.170
Right	0.925	1	-0.225	-0.078
Top	-0.013	-0.225	1	0.586
Bottom	-0.170	-0.078	0.586	1

The PCA results on those four features are shown in Fig. 2. In this figure, the  $x$  axis refers to the number of the principal component, while the  $y$  axis represents the cumulative amount of information (variation) represented by the principal components from 1 through  $x$ . From this figure, we can see that the first two principal components contain most of the information (over 80%); therefore, we can use these two principal components as the features that represent the camera sensor instead of the original 4 features.

PCA is more important for high-dimensional sensor data such as 180 separate laser readings that are highly correlated with their neighbors. The PCA results on all the 180 laser features are shown in Fig. 1(a). We can see that the first 8 features include most of the information (over 80%) and can thus be selected as the features for representing the laser sensor. In this task, since we only focus on the objects in front of the robot, we are most interested in the original laser data in the 70-110 degree range. The PCA results of

these 40 features are shown in Fig. 1(b). These results show that we can represent the laser data using only 4 features.

PCA can also be applied to the robot team. In the box pushing task, we apply the PCA using all 80 range features from both robots (each one provides 70-110 degree range data). Fig. 1(c) shows the results, which indicate that we can represent the laser data for both robots using only 5 features.

These results show that PCA performs well in reducing feature dimensions for the camera and laser sensor. However, it also has limitations, in that it is only helpful when the original data are highly linearly correlated. Fig. 3 shows

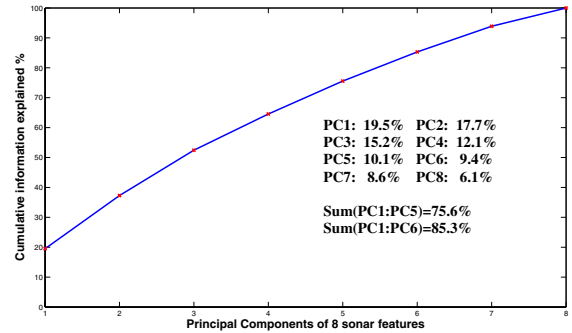


Fig. 3. PCA 8 sonar features.

the PCA result for the 8 sonar features, in which little reduction is achieved. This finding occurs because the 8 original sonar features are not highly correlated with each other, because their positions are not sufficiently close. To summarize, in the box pushing task, we can limit the sonar data to the 70-110 degree range and thus only two sonar features are selected. In addition, since both laser and sonar measure the distance of objects around the robot, these data are highly correlated and we can perform PCA on the combined laser and sonar data set.

In the box-pushing task, the possible features for selection are listed as follows:

- Laser range in 70-110 degrees (40 features)
- Sonar range (8 features)
- Robot speed
- Robot turn rate
- Robot current position (3 features)
- Robot current distance from original point

- Red block’s center in the camera image
- Red block’s left point in the camera image
- Red block’s right point in the camera image
- Red block’s top point in the camera image
- Red block’s bottom point in the camera image
- Battery charge

Performing clustering on these 60 features is barely meaningful because of the high dimension. We can manually monitor the feature values and filter out those that are irrelevant to reduce the dimension. However, the result of manual selection is dependent on human experience. Here, we show two manual selection results and compare their fault detection rate with PCA-selected features from 20 normal trials and 20 trials in which faults occur.

The first manual selection makes use of the following 16 features:

- Minimum laser range
- Laser index with minimum laser range
- Maximum laser range
- Laser index with maximum laser range
- Minimum sonar range
- Sonar index with minimum sonar range
- Maximum sonar range
- Sonar index with maximum sonar range
- Robot speed
- Robot turn rate
- Robot current position
- Red block’s center in the camera image
- Red block’s area in the camera image
- Red block’s width in the camera image
- Red block’s height in the camera image
- Battery charge

After studying the experimental results, we built the second manual selection with 8 features as follows:

- Minimum laser range
- Laser index with minimum laser range
- Minimum sonar range
- Sonar index with minimum sonar range
- Robot speed
- Robot turn rate
- Red block’s height in the camera image
- Battery charge

In comparison, the nine PCA selected features are listed as follows:

- Four principal components of laser and sonar data
- Two principal components of camera data
- Robot speed
- Robot turn rate
- Battery charge

Table II shows that PCA achieves an almost equivalent fault detection rate compared to a good manual selection of features; a bad manual selection of features causes a decrease in performance.

### 4.3. Data Clustering

We have implemented three clustering algorithms — K-means, Soft K-means and Fuzzy C-Means — on the sensor

TABLE II

FAULT DETECTION RATE FOR THREE FEATURE SELECTION METHODS.

	16 features	8 features	PCA selection
True positive	40%	90%	85%
True negative	55%	90%	95%
False positive	60%	10%	15%
False negative	45%	10%	5%

features obtained using PCA. Table III shows that Soft K-means and Fuzzy C-means are effectively equivalent for this application, with both giving good results. However, the crisp K-means clustering algorithm results in the most false positive errors, and in a lower accuracy for true positives. These errors are primarily caused by the mis-classification of noisy and partial data.

TABLE III

FAULT DETECTION RATES OF THREE CLUSTERING METHODS.

	K-means	Soft K-means	FCM
True positive	45%	80%	85%
True negative	90%	95%	95%
False positive	55%	20%	15%
False negative	10%	5%	5%

### 4.4. Building state transition diagram

The state transition diagram of one robot in the box pushing task was obtained from our previous work [13], and is shown in Fig. 4. To learn this model, the box pushing task was repeated 15 times, illustrating the “normal” operation of this task. During this task, the robot primarily switches among four states, which we label “push”, “wait” and two “alignment” states (alignment may happen in two different directions, left and right). The structure of the time-variant

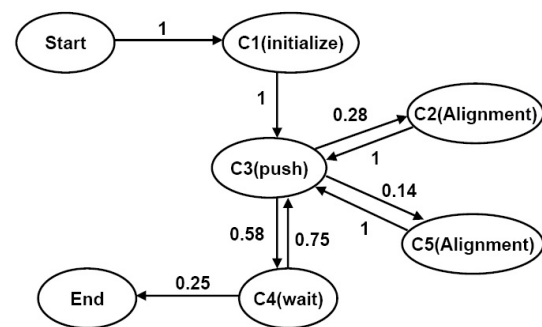


Fig. 4. State transition diagram learned for a single robot.

state transition diagram learned with our new approach is shown in Fig. 5. As can be seen, the new transition diagram is quite similar to Fig. 4 except for the transitions back to the same state. For example, Table IV shows the learned time-variant state transition probabilities of state C4. It shows that the probabilities of two possible transitions from state C4 (to C3 or remain in C4) vary as the time duration in C4 increases. The data illustrates that the longer the robot

TABLE IV  
THE STATE TRANSITION PROBABILITIES FROM STATE C4.

Time duration in C4 so far	1	2	3	4	5	6
Prob. of transition to C3	0.13	0.06	0	0.53	0.79	0.93
Prob. of remaining in C4	0.87	0.94	1	0.47	0.21	0.07

remains in state C4, the higher probability it has to change the state to C3. Compared to the average value shown in

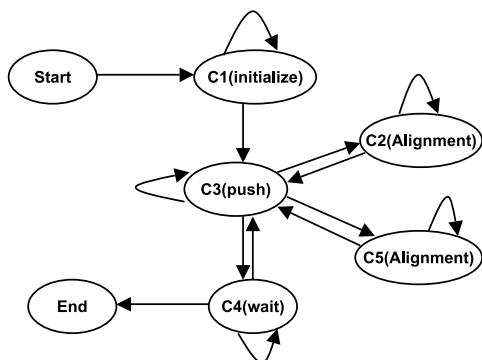


Fig. 5. Time-variant state transition diagram learned for a single robot.

Fig. 4, the time-variant transition probabilities can represent the operation of the robot system more accurately. Table V

TABLE V  
FAULT DETECTION RATE FOR TWO STATE TRANSITION DIAGRAMS.

	non time-variant	time-variant
True positive	75%	85%
True negative	65%	95%
False positive	25%	15%
False negative	35%	5%

records the fault detection rates for the two different state transition diagrams with PCA-selected features and Soft K-means clustering results, measured over 20 trials of normal operations and 20 trials of task executions that included a fault. These results indicate that incorporating time variant transition probabilities improves the performance of SAFDetection.

From the results shown above, we recommend the use of PCA feature selection, a soft classification technique (such as Soft K-means or Fuzzy C-Means), and time-varying transition probabilities with our SAFDetection approach.

## 5. Conclusions

In this paper, we have presented several design and performance improvement methods for our SAFDetection system. We addressed how to use Principal Components Analysis to select features automatically, and how to incorporate a time-varying factor into the state transition diagram to describe the system in a more accurate way. We also compared the fault detection rate of several clustering algorithms and showed the advantage of soft clustering techniques over

crisp ones. Our results show that a PCA feature selection approach, combined with a soft classification technique and time-varying transition probabilities, yields the best fault detection results. We believe that these methods are general enough to apply to a wide range of single- and multi-robot system applications. In continuing work, we are applying this extended SAFDetection approach for fault detection to more complex multi-robot applications.

## References

- [1] J. C. Bezdek, R. Ehrlich, and W. Full. FCM: The fuzzy c-means clustering algorithm. *Computers and Geosciences*, 10(2-3):191–203, 1984.
- [2] Z. Cai and Z. Duan. A multiple particle filters method for fault diagnosis of mobile robot dead-reckoning system. In *IEEE International Conference on Intelligent Robots and Systems*, pages 481–486, 2005.
- [3] W. Chen, R. Gong, and K. Dai. Two new space-time triple modular redundancy techniques for improving fault tolerance of computer systems. In *IEEE International Conference on Computer and Information Technology*, 2006.
- [4] Q. Cheng, P. K. Varshney, and J. Michels. Distributed fault detection via particle filtering and decision fusion. In *International Conference on Information Fusion*, pages 1239–1246, 2005.
- [5] F. J. Garca, L. J. Miguel, and J. R. Peran. Fault-diagnostic system using analytical fuzzy redundancy. *Engineering Applications of Artificial Intelligence*, 13:441–450, 2000.
- [6] J. A. Hartigan and M. A. Wong. A K-means clustering algorithm. *Applied Statistics*, 28:100–108, 1979.
- [7] M. Hashimoto, H. Kawashima, and F. Oba. A multi-model based fault detection and diagnosis of internal sensor for mobile robot. In *IEEE International Conference on Robotics and Automation*, pages 3787–3792, 2003.
- [8] B.P. Jeppesen and D. Cebon. Analytical redundancy techniques for fault detection in an active heavy vehicle suspension. *Vehicle System Dynamics*, 42:75–88, 2004.
- [9] I. T. Jolliffe. *Principal Component Analysis*. Springer-Verlag, 2002.
- [10] J. Kim, K. Shim, and S. Choi. Soft geodesic kernel k-means. In *IEEE International Conference on Acoustics, Speech and Signal*, volume 2, pages 429–432, 2007.
- [11] I. S. Lee, J. T. Kim, and J. W. Lee. Model-based fault detection and isolation method using ART2 neural network. *International Journal of Intelligent Systems*, 18:1087–1100, 2003.
- [12] M. L. Leuschen, J. R. Cavallaro, and I. D. Walker. Robotic fault detection using nonlinear analytical redundancy. In *IEEE International Conference on Robotics and Automation*, volume 1, pages 456–463, 2002.
- [13] X. Li and L. E. Parker. Sensor analysis for fault detection in tightly-coupled multi-robot team tasks. In *IEEE International Conference on Robotics and Automation*, pages 3269–3276, 2007.
- [14] M. Luo and D. Wang. Model-based fault diagnosis/prognosis for wheeled mobile robots: A review. In *Industrial Electronics Society, 2005. 32nd Annual Conference of IEEE*, pages 2267–2272, 2005.
- [15] B. Moore. Principal component analysis in linear systems: Controllability, observability, and model reduction. *IEEE Transactions on Automatic Control*, 26:17–32, 1981.
- [16] P. Sundvall and P. Jensfelt. Fault detection for mobile robots using redundant positioning systems. In *IEEE International Conference on Robotics and Automation*, pages 3781–3786, 2006.
- [17] F. Tian, H. Zhang, and Y. Lu. Research on modeling with dynamic bayesian networks. In *IEEE International Conference on Web Intelligence*, pages 606–609, 2003.
- [18] V. Verma and Simmons. Scalable robot fault detection and identification. *Robotics and Autonomous Systems*, 54:184–191, 2006.