Modeling Cross-Sensory and Sensorimotor Correlations to Detect and Localize Faults in Mobile Robots

Zsolt Kira
College of Computing,
Georgia Institute of Technology
Atlanta, GA, 30332, U.S.A.
Zsolt.Kira@cc.gatech.edu

***Abstract*— We present a novel framework for learning cross-sensory and sensorimotor correlations in order to detect and localize faults in mobile robots. Unlike traditional fault detection and identification schemes, we do not use *a priori* models of fault states or system dynamics. Instead, we utilize additional information and possible source of redundancy that mobile robots have available to them, namely a hierarchical graph representing stages of sensory processing at multiple levels of abstractions and their outputs. We learn statistical models of correlations between elements in the hierarchy, in addition to the control signals, and use this to detect and identify changes in the capabilities of the robot. The framework is instantiated using Self-Organizing Maps, a simple unsupervised learning algorithm. Results indicate that the system can detect sensory and motor faults in a mobile robot and identify their cause, without using *a priori* models of the robot or its fault states.**

I. Introduction

As mobile robots begin to be used for multiple complex tasks in different environments, adaptation to changes in the perceptual and motor capabilities of the robot becomes an increasingly important issue. Applications where such adaptation is crucial include planetary exploration, search and rescue, and in general applications where robots are expected to perform tasks for long periods of time with limited assistance. There exists a large body of literature on the topic of fault detection and identification for industrial processes [1] and manipulator robots [7-8], and a smaller but still substantial body of work for mobile robots [9]-[17]. Although successful, these approaches have several limitations that have not been addressed fully, especially for problems particular to mobile robots in unknown environments and autonomous adaptation [10]. Some limitations include the requirement for faults to be encountered and fault modes modeled *a priori*, limited or no handling of multiple faults or faults that degrade slowly through time, and limited control system adaptation.

We claim that some of these limitations can be ameliorated on a mobile robot by utilizing additional information available to the robot. Specifically, robots usually have available to them a hierarchy of perceptual processing that solve a particular task. Here we call them perceptual schemas; the definition of a schema used by researchers varies widely [18], but the aspect we are interested in is that perceptual schemas process sensor data (either raw or the result of lower perceptual schema) to gather knowledge relevant to motor schemas. Unlike in industrial systems, this hierarchy can be made large with the abundant variety of sensors and sensor processing algorithms that have been devised in fields such as computer vision, sound and speech analysis, laser and sonar-based processing, etc. In other words, these perceptual schemas encode *processed* sensor data that derive knowledge at multiple levels of abstraction useful for a particular task. Figure 1 depicts a simple example of perceptual schemas for one particular task, moving to a goal location. An important aspect of this is that mobile robots typically have several *heterogeneous* sensors that measure different, but similar, aspects of the world. The perceptual processing performed for a task already fuses the data from several sensors into similar representations.

The main insight of our work is that this hierarchy can be utilized to detect, identify, and adapt to changes in the capabilities of the robot by creating models or mappings between different elements in the hierarchy. Here, similarly to other works, we define a fault to be any abnormal behavior by the robot that results in a change in the pattern of sensing and acting data it receives. For example, faults that are not borne out in the perceptual schema used by the robot for the current task will not be detected. A particular task coupled with the perceptual processing performed for that task produces task-specific cross-sensory and sensorimotor correlations that can be modeled. Such mappings will change whenever the capabilities of the robot changes. For example, looking at the schemas in Figure 1, when a change occurs in its visual abilities the optical flow and visual obstacle detection will change. The outputs of these schemas will differ from the other schemas measuring similar aspects of the world, namely obstacle detection and relative motion estimation. The changes can be further identified because each schema is tied to a particular set of sensors or actuators. Mappings between schemas that do not utilize the faulty sensor or actuator will not be changed, hence ruling out particular sensors or actuators. In other words, sensor information from non-faulty sensors that measure similar aspects of the world can be used to localize the faulty sensor. In the visual fault example, the schemas

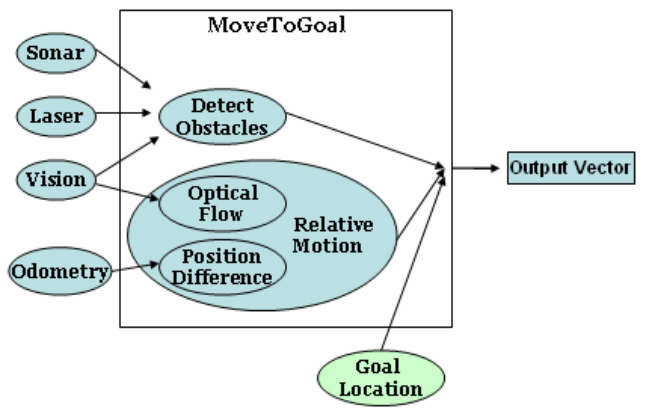

Figure 1 – Example of a motor schema and its associated hierarchy of perceptual schemas.

common to vision will change, whereas others will not. Note that the information measured does not have to be exactly of the same type, but only has to be statistically correlated. Finally, adaptation can potentially be performed by restoring the mappings between schemas that have become changed. For example, it may be possible to detect and correct for the optical flow vectors upon a camera rotation if the changes between the current mappings and previous mappings are established.

There are several advantages to this approach. A large majority of other approaches use *all* of the raw data from the sensors in order to diagnose particular sensors. Such systems do not take advantage of the fact that certain tasks use sensors differently or not at all, or that other sensors that sense similar physical properties or are processed to obtain the same type of knowledge may provide useful cross-checking information in diagnosis. Our approach creates models between perceptual schemas in a hierarchy for a specific task; although the algorithms are not task-specific, it is assumed that the mappings are learned on a per-task basis. By using the notion of task-specificity, only the sensors that are used and the output of those sensors are considered. Furthermore, sensor values that are ignored in a particular perceptual schema will likewise be ignored when learning the models. This limits computation time, but more importantly allows the learning algorithms to use more *relevant* data; *i.e.* data relevant to the task at hand. Also, most of the approaches use *a priori* models of a limited set of fault states, hence making them brittle to unknown environments and faults as well as multiple interacting faults (again, see [10] for agreement on these limitations).

By correlating perceptual schemas, the models can be learned in an unsupervised manner. Furthermore, after one fault the mappings can be restored and represent normal functioning thereafter; hence, if there is another change in the capabilities afterwards the same process can be used. In addition to brittleness, maintaining a discrete set of fault states makes more difficult the detection of gradual faults until those faults cause sufficient deviation from the models. Using correlations instead of modeled fault states presents the opportunity for non-discrete, gradual adaptation. Finally, the reconfiguration of control systems in order to adapt to the faults has not been addressed in most of the work [10]. We believe that modeling the interactions of perceptual schemas can form the basis of restoring normal functionality.

In summary, the overarching research question is whether task-specific cross-sensory and sensorimotor correlations experienced by a mobile robot can be used to detect, localize, and adapt to multiple, *unanticipated*, time-varying sensing and motor capability changes. This first paper seeks to address one small part of this question, namely whether faults can be detected and localized in an unsupervised manner by modeling cross-sensory and sensorimotor patterns. We further address the question of whether mappings that are inherently inconsistent (*e.g.* between different sensors measuring independent attributes of the world) can be ruled out during training. In future papers, we will attempt to answer the rest of the questions posed above, including whether faults can be identified in an automated fashion, whether our approach can detect classes of faults that have traditionally been unexplored (*e.g.* time-varying faults or multiple interacting faults), and whether our approach can add new adaptation capabilities (*e.g.* automatic control system reconfiguration).

II. THEORETICAL FRAMEWORK

We will now describe the general framework for our system. Figure 2 shows the steps involved during training and testing of the system, and the following subsections expand upon each step.

A. *Input*

It is assumed that, as an input, a hierarchical structure of perceptual schemas is available. Such information is already explicitly available in, for example, behavior-based systems but can also be easily available in robotic systems where modular programming practices have been followed. Specifically, a directed acyclic graph (DAG) P is provided, where each node represents the output of specific perceptual processing and the links represent inputs to the processing. The root nodes of the DAG are the raw sensors available to the robot (*e.g.* vision, laser, etc). The specific type and amount of processing that warrants a new node is up to the programmer; however, presumably in general the deeper a node of the graph is, the more abstract the knowledge it represents becomes since it is processed more and utilizes more sensors than lower levels. The goal of this research is to eventually provide guidelines for the type of information that would be useful, in terms of the amount of independence between the schemas and other properties, as well as desirable properties of the graph. For now, we arbitrarily pick the processing points at which nodes lie.

B. *Construction and Learning of Correlation Graph*

The input graph P is subsequently transformed into a fully connected undirected graph G, with the perceptual schemas

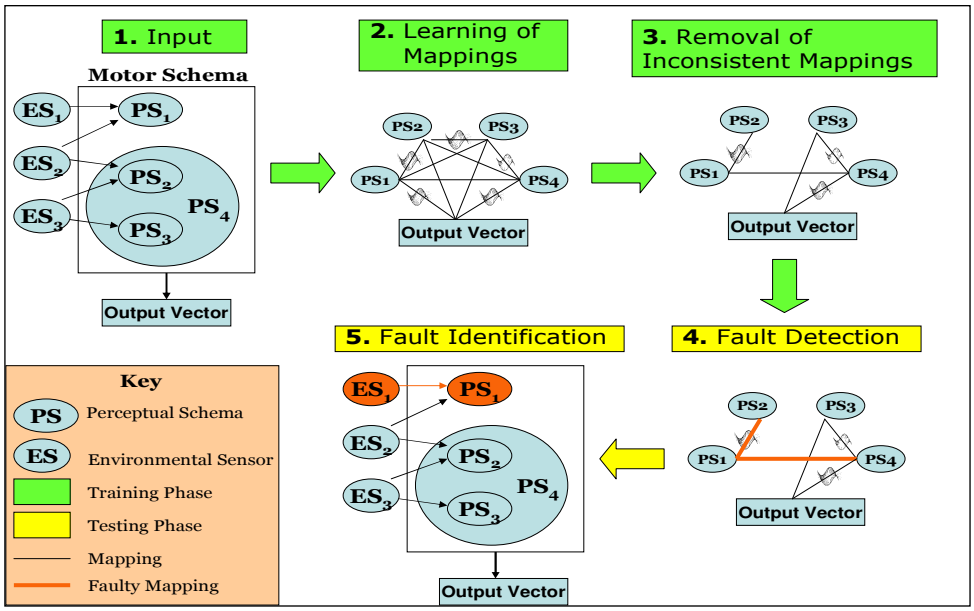

Figure 2 – Steps involved in the detection and identification of changes in the perceptual and motor capabilities of a robot in our system.

and the output of the motor schema (*i.e.* the control signal) as nodes. The links in this graph represent functional mappings between the two perceptual schemas, *i.e.* the predicted value of the output of one schema given the output of the other. In general it can represent any functional mapping, and we call this graph the *correlation graph*.

There are several alternatives to initializing the graph as fully connected. For example, connections between two perceptual schemas that feed into one another will probably not lead to useful correlations, as they come from the same sources. In general, there may be algorithms analyzing the transformation from the hierarchy P to the graph *G*.

Once the graph *G* has been created, the functions represented by the edges must be learned. Through training data obtained during normal functioning of the robot, a model is created between each pair of cross-sensory (two perceptual schema) or sensorimotor (a perceptual schema and the motor output) connections. Models can be simple or complicated statistical models, or even sensor-specific models (*e.g.* if a motion model is known).

Although many representations are possible, there are several properties that are desirable for functionality and practicality. Specifically, that:

1. The representation can be learned for a given task and environment in an unsupervised manner, without *a priori* fault models.

2. The representation can detect which sensorimotor and cross-sensory mappings are valid; that is, useful for detecting and adapting to faults. Mappings that are inconsistent during *normal* operation can be ruled out.

3. In order to detect unanticipated faults, only data during normal operation is assumed to be available. The representation can be used to detect and identify faulty sensing or motion.

4. The representation can be used to adapt the control system after the fault.

5. The representation can be used to detect and adapt to multiple faults occurring after each other.

6. The representation can detect and adapt to faults occurring at different time-scales (*e.g.* slow-varying faults such as tires losing air).

C. Removal of Inconsistent Mappings

Once the models are learned, a further training step involves removing inherently inconsistent links. Some perceptual schemas during a task, for example odometry and sound-based sensors, reflect inherently independent aspects of the world. Hence, modeling their interaction will not be useful. In order to do this, a metric for consistency is required for a given representation in order to rule out inconsistent mappings *a priori*.

D. Detection of Changes

Once consistent models have been found and learned, the stability of the mappings can be monitored. Again, a metric or set of metrics must be provided to determine whether a mapping is stable or not. These metrics can be the same as, or different from, the metric used to remove inconsistent mappings. It is hypothesized that changes or faults in the perceptual or motor capabilities of the robot will lead to instability in one or more mappings. Of course, smoothing and other techniques must be performed in order to deal with noise and other factors.

E. Identification of Source of Change

Finally, given a graph with connected perceptual and motor schemas, along with a set of mappings that are inconsistent, the source of the fault can be traced back.

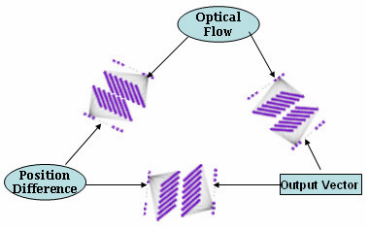

Figure 3 – Diagram depicting connection between three schemas.

Assuming that there are several perceptual schema measuring similar aspects of the world, some of the links will remain consistent allowing the robot to rule out certain sensors. Hence, reasoning upon this graph can be used to trace back to the fault. Note that if there is not enough redundancy in the type of knowledge the robot has from several sensors, there may be ambiguity as to the cause of the fault. In general, the more sensors and sensor processing there is, the better the fault identification will be.

III. FRAMEWORK IMPLEMENTATION

The preceding section described a general framework for cross-sensory and sensorimotor integration in order to detect changes in capabilities. We have implemented a simple instantiation of the framework in order to test the proposed methods and see whether unsupervised learning can be used to detect and identify faults using training data, with perceptual processing at different abstraction levels, obtained during normal operation. In order to instantiate the framework, the following have to be defined: 1) Preprocessing algorithm for transforming *P* into *G* (*i.e.* whether *G* is fully connected or heuristics and other knowledge is used to remove some links *a priori*), 2) Representation *R* used to define the links in graph *G*, 3) A consistency metric for *R* to rule out inconsistent mappings, 4) A metric for determining when a mapping has changed during operation (can be the same as the consistency metric), and 5) An algorithm to identify the source of the change through analysis of the two graphs *P* and *G*, and the locations of the changed mappings.

In our case, we do not preprocess the transformation from *G* to *P*, *i.e.* we do not rule out any connections and leave *G* fully connected. The model or representation used to link the schemas is a simple unsupervised method, namely Self-Organizing Maps (SOM) [20]. A SOM is an artificial neural network that has been used for classification or clustering of input spaces in an unsupervised manner. The network consists of a lattice of nodes or cells, usually configured in a rectangular or hexagonal shape. Each node has associated with it a weight vector, whose dimensionality equals the dimensionality of the input. For each input to the network, the output is the node or weight associated with it inside the actual network (alternatively, arbitrary vectors can be associated with each node and that can be the output). Initially, these weight vectors are initialized randomly. Training occurs in the following manner. For each input vector, the winner node is obtained. The winner node is defined as the node whose weight vector is most similar to the input vector (similarity can be arbitrarily defined, but is usually just Euclidean distance). The weight vector of the winning node is then modified. In addition, the weight vector of nodes that are within a certain neighborhood of the winning node is also modified. The update rule can be expressed as follows:

$$w_i(t+1) = \begin{cases} w_i(t) + \alpha(t)(x(t) - w_i(t)) & i \in N_c(t) \\ w_i(t) & \text{otherwise.} \end{cases} \quad (1)$$

where $w_i(t)$ is the weight vector of node i at time t, $a(t)$ is the learning rate, $x(t)$ is the input at time t, and $N_c(t)$ is the neighborhood function.

Figure 3 depicts the implementation with two perceptual schemas and a motor schema. We aim to show that this representation fulfills the first three requirements listed previously, namely it is unsupervised, can provide a consistency measure, and can be used to detect and identify faults in motor and perceptual capabilities using only data gathered from normal operation. For each pair of perceptual schemas or motor schemas, a Self-Organizing Map with a vector corresponding to both percepts or motor output was created. During the training phase, these SOMs were trained on data from a normally functioning robot. This reduced the dimensionality of the data and utilized the property of SOMs that enables them to represent parts of the sensory space that occur more frequently with more nodes. In other words, the SOM will represent pairs of percept values that occur more frequently using more nodes than pairs of percept values that occur less frequently.

The consistency metric used to identify stable mappings is quantization error, that is the average error (*i.e.* distance) between the winner node and the data for a separate training set that is gathered during normal functioning. If the quantization error differences between the training data (measuring baseline noise in the data itself) and the normal testing data is statistically significant, then the mapping is determined to be unstable.

In order to use the system in testing mode to detect and identify faults, perceptual and motor data coming from the robot are processed through the SOM, and the average quantization error is calculated. This is the metric used to identify mappings that have changed. Quantization errors that are significantly different that those obtained during testing in normal operation are assumed to indicate a fault. In order to identify the source of the problem, we use a simple algorithm that identifies the node for which all of the links have changed. Note that this is a simple algorithm used to show that the framework is feasible; in order to detect different types of faults more complicated analysis must be instantiated.

IV. EXPERIMENTAL DESIGN

In order to test whether it is feasible to detect and identify faults, the robot was teleoperated along similar paths for the following conditions: 1) Normal operation for training (10

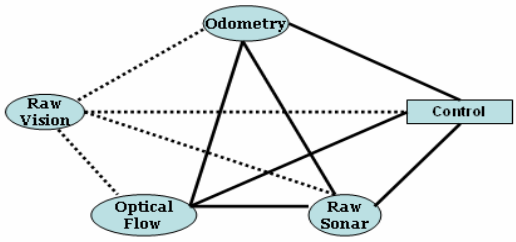

Figure 4 – Undirected graph for the host of sensors and motor output used for the experiments. The solid lines represent stable mappings as determined during training, while the dotted lines represent mappings that have been ruled out during training.

runs) 2) Normal operation for elimination of mappings (10 runs) 3) Operation with a camera that has been hit and rotated approximately 30 degrees (10 runs). 4) Operation with faulty wheels that are loose (6 runs). The perceptual data available included odometry, proprioception (the velocity commands sent to the robot), raw sonar data, raw visual data (reduced to 5x5 images), and optical flow processed on the vision input. Training was done using data from normal operation. Testing consisted of running the trained system through the data obtained during faulty operation.

During testing, it is hypothesized that for the first fault (rotated camera), the mappings between odometry and optical flow and proprioception and optical flow would produce the most errors. This is because the rotated camera would change the correlation between the flow and actual motion, as well as the correlation between the flow and odometry. The correlation between odometry and optical flow should remain unchanged. This pattern of errors allows one to identify the fault as being in the visual system.

It is hypothesized that for the second fault (loose wheel), the mappings between odometry and proprioception and proprioception and optical flow would produce the most errors. This is because despite the wheel fault, the odometry and the optical flow percepts measure similar aspects of the world (relative motion), and hence the fact that the robot did not move as commanded would not affect this mapping. This pattern of errors allows one to identify the fault as being in the motion of the robot.

The robot used to gather the data was a Pioneer 2DX with an onboard Dell Latitude 100L laptop, with a 2.66GHZ Mobile Intel Pentium 4 CPU and 1GB of memory. Sampling was done asynchronously when gathering, and was later synchronized through software. SOMPAK, a Matlab implementation of Self-Organizing Maps, was used with the following parameters: map size of 10x10, initial radius of 5, an initial alpha of 0.5, a Gaussian neighborhood function, and 10 epochs during training.

V. RESULTS AND DISCUSSION

The first interesting thing to look at in the results is to see which mappings of the graph *G* of perceptual schemas have been determined to be stable during training. Figure 4 shows these mappings in solid lines, while mapping that have been ruled out are dotted. In this case, all of the mappings have been determined to be stable except for the raw reduced vision data. Unlike the raw visual data, raw sonar data did produce stable mappings. Although it may seem raw sonar data would not provide correlations with control and other perception, in this case they may have because the data was sparse and the teleoperation was very repetitive. For other tasks that are more varied, this will likely not be the case. This shows that for different tasks, different data or mappings will be deemed useful and the system only utilizes those which are useful for the current task.

Figure 5 displays the broken linkages, as determined by significantly different quantization errors from the normal testing data, in graphical form for each of the two faults described previously. Figure 6 displays the results in terms of average quantization errors for all mappings during normal and faulty operation. The quantization errors during operation with a faulty camera validates the first hypothesis: the quantization errors for the mappings involving optical flow are significantly higher, whereas they are the same for other mappings. Unexpectedly, the mapping between optical flow and odometry differed as well. This could be due to the simple learning method used, as a change in the distribution of optical flow would change the model learned. Note that, as intended, this can be used to *identify* where the fault has occurred. One can detect which mappings contained errors, and localize the fault to the common node: In this case, between optical flow and the other sensors.

Figure 5 and 6 also display the results that confirm most of the second hypothesis. Again, the mappings can be used to identify where the fault has occurred by tracing the common node into which mappings are faulty, namely the proprioception or the relation between motor commands and the perceptual result. Note that unexpectedly, the mapping between optical flow and proprioception stayed the same, although with a larger confidence interval. Hence, it could be that the sensor (optical flow) is too noisy to measure changes in the correlations in this case. In addition, the mapping between odometry and raw sonar differed as well, possibly due to the fact that due to the mechanical failure, the robot was not teleoperated along the same route resulting in different sonar readings for different odometry values.

In order to evaluate the system with two co-occurring faults, an additional artificial fault was created by zeroing out one column of the sonar sensor, with and without the faulty camera. Figure 7 shows a diagram representation of the mappings and Figure 8 shows the raw quantization errors. As can be seen, with a sonar fault two mappings with sonar are significantly different, with a camera fault three mappings with optical flow are significantly different, and with both faults the union of those mappings are significantly different. However, when training on data obtained with broken sonar (T2), it is as if only the camera sensor is broken. This shows that once a fault is detected, training can be performed in order to assign it as “normal”. Although this feature is conferred by other data-driven methods, in this

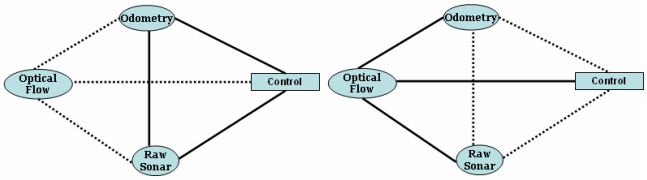

Figure 5 – Mappings during camera (left) and wheel (right) faults. Solid lines indicate mappings that have remained stable despite the faults, and dotted lines indicate mappings that have not.

system only a small subset of mappings or data have to be retrained, resulting in time savings. If a model-based method is used, then models for the system with both faults must be derived *a priori*, which is difficult to obtain.

VI. RELATED WORK

The literature for fault detection and identification in the context of industrial processes and industrial robotics is extremely large in scale. Surveys and books include [1], [3], [4], [5], and [6]. Besides numerous approaches that use physically redundant sensing , by far the dominant approach involves *analytic redundancy* whereby models of the dynamics and interactions of the system are created and differences between the predicted values and sensed values are used to detect and identify faults [2]. Often *residuals*, functions of the data and model differences that are sensitive to specified faults, are created [4]. Most of the approaches using this concept utilize *a priori* models of the system itself, be it the chemical processes being monitored or the kinematics of the manipulator robot [8]. They also rely on pre-determined fault states for which residuals can be derived. Some approaches use *virtual sensors* that are learned models of properties related to the faults [11]. Again, this requires knowledge of the faults beforehand. There are also knowledge-based methods that employ more qualitative models [1]. If robots are to perform tasks and survive for long periods of time unassisted, such anticipation of faults would be an undue burden.

Data-driven methods include multivariate statistical techniques such as principal component analysis (PCA) or fisher discriminant analysis (FDA) [1]. Such methods transform the data into subspaces that minimize variance (PCA) or are optimal for discriminating known classes (FDA). Again, some of these methods require classes of faults that are determined *a priori*. Other methods minimize metrics such as variance, but it is not clear that such a subspace will be good for detecting faults. Furthermore, they do not provide models of the interaction between the variables explicitly, and hence cannot be used to *restore* the correlations in order to adapt, nor do they discard irrelevant data before processing.

A key difference between fault detection and identification in these domains and in mobile robotics is that in the industrial environments data from sensors such as temperatures, pressures, etc. represent physical states that

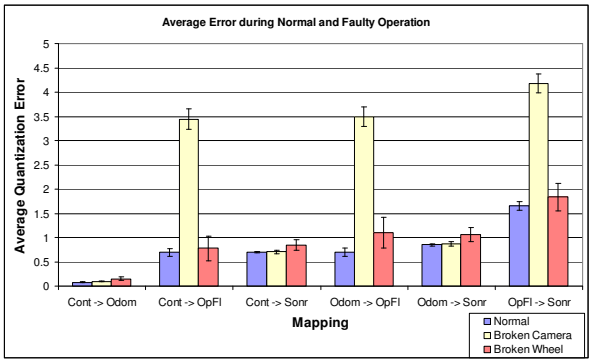

Figure 6 - Average quantization error with wheel and camera faults, for each mapping.

cannot be processed further in multiple ways. In contrast, mobile robotics utilizes rich sensors such as vision, laser, and sound that can be *processed* in multiple ways to obtain different types of information. There are huge bodies of literature in the fields of vision, acoustics, and perception that have devised many algorithms for processing the sensor data. The hierarchy that describes the flow of this processing (*i.e.* which sensors are used, etc.) can be utilized to provide a further form of redundancy. It makes sense to create methods that automatically correlate such data and detect faults through changes in these correlations.

Although there are an increasingly large number of papers on fault detection and identification in the mobile robotics community, most of the approaches follow their predecessors in industrial applications in using concepts such as state estimation and analytic redundancy. For example, there have been specialized particle filtering algorithms designed to be risk-sensitive or with variable resolution [13].

In other words, there have been few systems that utilize the richness of the sensors and the redundancy they can provide. One notable exception is [14], whereby redundant position information obtained at multiple levels of abstraction were used. However, in that case the difference between the two position estimates were fed to a Kalman filter state estimator. As a result, *a priori* models were used in that instances as well, and the information obtained from the two sensors were identical. In our system, we model the correlations between different information obtained from sensors, allowing for richer relationships. Furthermore, there has not been a general framework created for exploiting the redundancy provided by multiple levels of representation, to the best of our knowledge.

In this work, the framework creates models of various levels of perceptual processing independent of what they are used for. Modeling the statistics of raw sensory data is not unheard of; for example, pattern theory [19] attempts to describe statistical patterns in, for example, natural scene images. Finally, sensorimotor correlations have been explored in areas such as developmental robotics [21]; however, there is no demonstration or focus on fault detection and adaptation. The main contribution of this work is to create a general framework for exploiting the

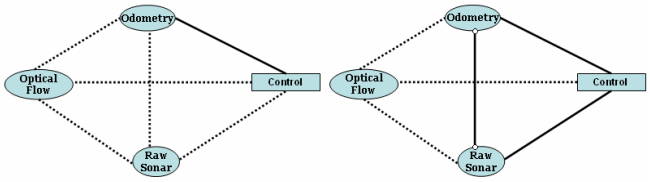

Figure 7 - Mappings with multiple co-occurring faults (both camera and sonar faults). Left: Mappings after training on normal data. Right: Mappings after training after only the sonar fault, hence isolating the camera fault.

redundancy provided by the rich hierarchical sensory processing available on mobile robots today.

V. CONCLUSIONS

In summary, we have proposed a novel task-based approach that utilizes cross-sensory and sensorimotor correlations to detect, localize, and adapt to faults. In this paper we have demonstrated that given an input hierarchy of the perceptual processing performed on a robot, correlations between elements of the hierarchy can be learned in an unsupervised manner. We showed that unstable mappings can be identified during training and ignored, and that the remaining stable correlations can be used to detect and identify both motor and perceptual changes in the capabilities of the robot without *a priori* models of system dynamics or pre-specified faults.

The results shown here are preliminary and have only answered a few of the questions we have posed. There is a great deal of future work in implementing the framework completely, replacing some of the simple blocks we have substituted with more grounded and robust algorithms. After this, the system will be tested with a larger and more realistic perceptual processing graph, with several vision, laser, and sound processing algorithms.

There is then future work in order to demonstrate the efficacy, scalability, and robustness of the architecture. We will attempt to answer the rest of the questions posed, including whether our approach can detect classes of faults that have been relatively unexplored (*e.g.* time-varying faults). An important area for future research is to see whether the mappings learned here can form a basis for adaptation capabilities, by restoring old mappings that are no longer stable.

ACKNOWLEDGMENT

I wish to thank SAIC for their generous research support.

REFERENCES

[1] L.H. Chiang, E.L. Russell, and R.D. Braatz, *Fault Detection and Diagnosis in Industrial Systems."*, Heidelberg:Springer, 2001.

[2] E. Chow and A. Willsky, "Analytical redundancy and the design of robust failure detection systems," IEEE Transactions on Automatic Control, 29, 603-614, 1984.

[3] P.M. Frank, "Fault Diagnosis in Dynamic Systems Using Analytical and Knowledge-based Redundancy-A Survey and Some New Results," *Automatica*, Vol. 26, No. 3, pp. 459-474, 1990.

[4] Frank, P. and X. Ding, "Survey of robust residual generation and evaluation methods in observer-based fault detection systems", Journal of Process Control, 7, 403-424, 1997.

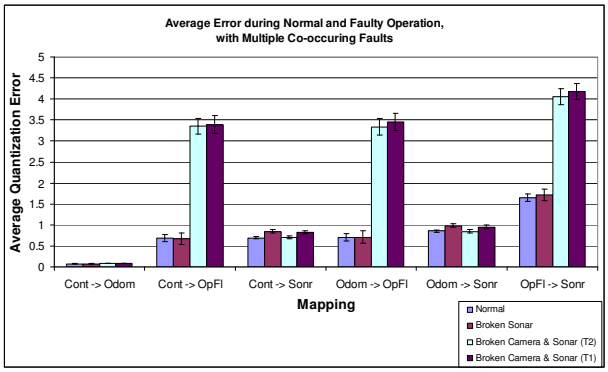

Figure 8 - Average quantization error with multiple co-occurring faults, for each mapping. T1 indicates training with normal data, and T2 indicates training with only faulty data.

[5] A. Willsky, "*A survey of design methods for failure detection in dynamic systems*," Automatica, no. 12, pp. 601--611, 1976.

[6] Gertler, J. *Fault detection and diagnosis in engineering systems.* New York: Marcel Dekker, 1998.

[7] M. Mufti, V. Ramani, George J. Vachtsevanos: ""Fuzzy Wavelets for Feature Extraction and Failure Classification,"" in Fuzzy Hardware, Architectures, and Applications, edited by A. Kandel and G. Langholz, Kluwer Academic Publishers, pp. 311-344, 1998.

[8] Fault diagnosis and fault tolerance for mechatronic systems : recent advances / Fabrizio Caccavale, Luigi Villani, (eds.), 2003.

[9] M. Visinsky, Fault Detection and Fault Tolerance Methods for Robotics, *Masters Thesis*, (December 1991).

[10] D. Zhuo-hua, C. Zi-xing, and Y. Jin-xia, "Fault Diagnosis and Fault Tolerant Control for Wheeled Mobile Robots under Unknown Environments: A Survey", Proceedings of the IEEE International Conference on Robotics and Automation, April 2005.

[11] C. Ferrell. Failure recognition and fault tolerance of an autonomous robot. Adaptive Behavior , 2(4):375-398, 1994.

[12] S. I. Roumeliotis, G. S. Sukhatme, and G. A. Bekey. "Fault detection and identification in a mobile robot using multiple-model estimation", In Proc. 1998 IEEE International Conference on Robotics and Automation, pages 2223--2228, May 1998.

[13] Vandi Verma, Geoff Gordon, Reid Simmons, and Sebastian Thrun, "Particle Filters for Rover Fault Diagnosis", IEEE Robotics & Automation Magazine special issue on Human Centered Robotics and Dependability, June 2004.

[14] P. Sundvall and P. Jensfelt, "Fault detection for mobile robots using redundant positioning systems", Proceedings of the IEEE International Conference on Robotics and Automation, April 2005.

[15] Dixon, W.E.; Walker, I.D.; Dawson, D.M.; "Fault detection for wheeled mobile robots with parametric uncertainty," in *Proc. IEEE/ASME Int'l Conf. Advanced Intelligent Mechatronics*, 2001, pp. 1245 - 1250.

[16] V. Verma, J. Langford, and R. Simmons, "Non-Parametric Fault Identification for Space Rovers," International Symposium on Artificial Intelligence and Robotics in Space (iSAIRAS), 2001.

[17] R. R. Murphy, and D. Hershberger, "Handling sensing failures in autonomous mobile robots," International Journal of Robotics Research, vol. 18, no. 4, pp. 382-400, 1999.

[18] R.C. Arkin, *Behavior-Based Robotics.* MIT Press, 1998.

[19] D. Mumford, Pattern Theory: The Mathematics of Perception, in *Proceedings of ICM 2002, Beijing*, vol. 1, 2002.

[20] Kohonen, T. 1990. "The self-organizing map", Proceedings of IEEE, 78:1464–1480.

[21] Y. Yoshikawa, H. Kawanish, M. Asada, and K. Hosoda, "Body scheme acquisition by cross map learning among tactile, image, and proprioceptive spaces," in *Proceedings of the 2nd International Workshop on Epigenetic Robotics*, pp. 181-184, 2002.