Estimating Time Available for Sensor Fusion Exception Handling

Robin R. Murphy

Erika Rogers†

Center for Robotics and Intelligent Systems
Colorado School of Mines
Golden, CO 80404-1887
Office: (303) 273-3874, FAX: (303) 273-3875
rmurphy@mines.colorado.edu

†Clark Atlanta University
223 James P. Brawley Dr. S.W.
Atlanta, GA 30314
erika@pravda.gatech.edu

Abstract

In previous work, we have developed a generate, test, and debug methodology for detecting, classifying, and responding to sensing failures in autonomous and semi-autonomous mobile robots. An important issue has arisen from these efforts: *how much time is there available to classify the cause of the failure and determine an alternative sensing strategy before the robot mission must be terminated?*

In this paper, we consider the impact of time for teleoperation applications where a remote robot attempts to autonomously maintain sensing in the presence of failures yet has the option to contact the local for further assistance. Time limits are determined by using evidential reasoning with a novel generalization of Dempster-Shafer theory. Generalized Dempster-Shafer theory is used to estimate the time remaining until the robot behavior must be suspended because of uncertainty; this becomes the time limit on autonomous exception handling at the remote. If the remote cannot complete exception handling in this time or needs assistance, responsibility is passed to the local, while the remote assumes a “safe” state. An intelligent assistant then facilitates human intervention, either directing the remote without human assistance or coordinating data collection and presentation to the operator within time limits imposed by the mission. The impact of time on exception handling activities is demonstrated using video camera sensor data.

Keywords: sensor fusion, mobile robots, teleoperations, evidential reasoning, Dempster-Shafer theory fault tolerance

1 Introduction

Previous work has described the VIA-SFX architecture [12], a teleoperation robot architecture for limited autonomous perceptual and motor control with a knowledge-based operator assistant providing strategic selection and enhancement of relevant data. The assistant is expected to reduce cognitive fatigue by managing the presentation of sensor data and by guiding planning and problem solving activities at the local. The assistant maintains a low communications bandwidth by requesting only the data from the remote which is believed pertinent to the current cognitive task. The overall work efficiency of the operator is likely to increase as the cooperative assistant assumes more management responsibilities, freeing the operator to supervise multiple remotes. VIA-SFX supports the incremental addition of artificial intelligence as more progress is made in learning and planning.

This paper reports on current efforts on extending and improving the VIA-SFX architecture to accommodate cooperative assistance in recovering from sensing failures in teleoperated mobile robots. A *sensing failure*, or *exception*, is declared when the perceptual processing needed to support a motor behavior is not able to return a percept with

a high degree of certainty. Sensing may fail for one or more of the following three reasons: a sensing malfunction has occurred (e.g., broken camera lens), the environment has changed with deteriorative effects on sensing (e.g., the lights are turned off), or the remote has errant expectations (e.g., is told to look for something that isn't there). For the purposes of this paper, *exception handling* is the process of detecting a sensing failure, classifying the cause(s), and recovering by instantiating a new sensing plan.

The paper concentrates on the impact of time on exception handling activities at both the remote and local. One objective of VIA-SFX is to allow the remote to be as self-sufficient as possible and to demand operator interaction only when there is no other safe option. The scheme described and simulated in this paper specifies when and for how long the remote can maintain autonomous operations while attempting to identify and recover from a sensing failure. It also specifies when the remote must seek help from the operator, even though it has not necessarily exhausted its own autonomous problem solving resources. Projected measurement error and the decay in the belief for an object are used to estimate the real-time deadlines for the various exception handling activities.

The contributions of this paper are threefold. First, it puts forward a strategy for incorporating the role of time in constraining the exception handling activities at both the remote and local. This is expected to allow robots to operate more effectively and reliably in domains with hard mission deadlines without increasing cognitive overloading of the operator. Accordingly, it is expected to allow operators to supervise multiple remotes. Second, it uses a generalization of Dempster-Shafer theory to estimate the time limits on decision making. Third, it integrates the results of research in fully autonomous mobile robots into a teleoperations application, illustrating the evolution of human cooperative assistance architectures from heavily reliance on the human to increasing self-sufficiency.

The remainder of this paper is divided into five sections. Section 2 reviews techniques for exception handling for sensing failures, noting their shortcomings. Exception handling in cooperative assistance architectures is also summarized. Section 3 presents the modifications to the VIA-SFX architecture and how they address the shortcomings found in current exception handling methods and teleoperation control schemes. The generalization of Dempster-Shafer theory is introduced in Section 4. Work in progress testing and implementing these modifications is presented and discussed next in Section 5. Simulations of the remote using actual sensor data show the estimation of deadlines and detection, classification, and recovery activities at the remote. Interim conclusions are drawn in Section 6.

2 Previous Work

Methods for autonomously handling sensing failures have been successful for applications where the operational environment and possible failure modes are well-known and readily modeled. It is unlikely that most of these techniques will be transferable to mobile robots in open worlds. As summarized below, cooperative assistance architectures have generally attempted to facilitate human intervention from the local, but have neglected problem solving activities at the remote.

2.1 Exception Handling for Autonomous Mobile Robots

The bulk of exception handling for sensing has been applied to domains other than robotics, such as sensor monitoring in manufacturing plants [7,9,10]. Both [9] and [10] use pattern recognition methods to detect errors in sensors observing manufacturing plants. Fernandez and Durrant-Whyte [9] uses a pattern detector to pull up the correct model of how the sensors should be operating and identify the causes. Naidu, Zafriou, and McAvoy [10] use a neural network to classify errors with good results. Weller, Groen, and Hertzberger [16] perform context sensitive tests to detect sensing errors after each processing step for generic sensor applications. These efforts each heavily rely on task- and platform-specific models of sensor interaction, limiting their transferrability to new domains. Also none considered the impact of time on the diagnostic and recovery process.

Chavez and Murphy in [1] specifically targeted handling sensing failures in autonomous mobile robots in the SFX-EH architecture. They addressed failures in a perceptual process supporting a reactive motor behavior. Failures were detected via evidential reasoning then classified using a Generate, Test, and Debug [14] methodology. In keeping with their use of reactive behaviors, their perceptual process attempts to classify and recovery for the sensing exception

without any additional information or assistance from other computational agents, resulting in several limitations. The most significant is an inability to handle *contention* for resources external to the perceptual process. This is problematic for both classification and recovery. Consider determining between the possibilities of whether a video camera has failed or if lights have been turned off. If there is a second video camera, it can be queried to see if its light readings are normal; this type of test is easily generated by the GTD method. But if the second video camera has been allocated to another perceptual process, there will be contention, and some type of negotiation scheme must be implemented. Contention during recovery can arise if the perceptual process attempts to replace a defective sensor with another sensor currently allocated. Second, *compound failures* exacerbate the classification process. Whereas it was fairly easy to automate tests to confirm hypotheses of single source failure, it was very difficult to generate conclusive tests when two problems happened simultaneously. A third problem with the Chavez and Murphy implementation was that it did not limit the amount of *time* used to classify and recovery from the exception; it was assumed that the robot could default to a safe behavior and wait until the sensing problem was resolved. This may not be true of robots in hostile environments where the prolonged exposure is undesirable.

Farrell [8] also has addressed fault tolerant sensing for a hexapod robot. Her approach was to write component-specific detection and recovery schemes. It considered the impact of time on exception handling only in retesting a sensor at periodic intervals in case it may have spontaneously recovered.

2.2 Cooperative Assistance

While the Chavez and Murphy work shows that a GTD methodology is suitable for classifying mobile robot types of sensing failures, the difficulties that they encountered suggests that human assistance may be necessary to resolve contention and determine the source of compound failures. Therefore, an logical alternative is to merge the portions of exception handling which are successful into a teleoperation architecture which supports cooperative assistance. Coiffet and Gravez [2], Edwards, et. al [3], and Rogers and Murphy [12] each have proposed such architectures.

Of the three systems, Murphy and Rogers' architecture, VIA-SFX, is the most concerned with problem-solving assistance for sensing related issues. VIA-SFX is a *semi-autonomous* control architecture which allows the remote robot to proceed autonomously until it completes a task or encounters a difficulty it cannot resolve. It consists of three computational agents: the remote robot, the human operator, and an *intelligent assistant*. The intelligent assistant acts as an intermediary between the human and the robot. It uses a blackboard architecture to observe and manage information posted independently by the remote and human intelligences. As technology advances and the remote acquires more intelligent capabilities, the assistant can be modified accordingly.

Previously reported work on VIA-SFX concentrates on the use of the cooperative assistance agent aiding human in completing the exception handling activities at the local. The major novel technique for cooperative assistance is *visual interaction* [11]. Under this paradigm the operator communicates with the intelligent assistant and the remote via a graphical interface. The graphical interface uses cognitive models of visual problem solving in conjunction with task-dependent models to determine what information, sensor data, and associated levels of image enhancement and/or presentation style to display. The intelligent assistant manages the hypotheses, reminds the operator of appropriate diagnostic procedures, requests sensor data from the remote, and then enhances it to highlight attributes needed to confirm the current hypotheses. The assistant also displays relevant contextual information such as terrain or cartographic data.

As a result of the focus on cooperative assistance at the local, the previous version of VIA-SFX did not precisely define the exception handling activities at the remote. Likewise it has largely ignored the issue of time constraints.

3 Exception Handling Extensions in VIA-SFX

In order to address its shortcomings for exception handling, the VIA-SFX architecture has been refined to account for the impact of time on exception handling activities and operator attention. Both the remote and local systems must now consider deadlines as they either autonomously, or with human assistance, generate and test hypotheses about the nature of the sensing failure. A method of computing the real-time deadlines using evidential reasoning has been

developed and is presented in the next section.

This paper posits that there are two natural deadlines in exception handling. First, there is the time that the system can afford the remote to autonomously classify and recover from the sensing failure. Second, is the time the system can devote as a whole to exception handling, either at the remote or the local, before it must abort the behavior and do something else.

3.1 Types of Deadlines

In computing these deadlines, it should be noted that when a sensing failure occurs, a remote may be able to continue executing the behavior for a period of time in a “dead reckoning” mode. The period of time from the detection of a sensing failure by a remote to when it cannot safely continue executing the behavior will be designated as t_s , *the time remaining until the execution of the behavior must be suspended*. During this time the operator does not need to be involved while the remote is attempting to autonomously recover from a failure; this allows the operator to continue with current tasks without needless interruption. If the remote is successful, a message can be logged with the operator that a failure was detected and resolved. But it does not require the operator to immediately read or acknowledge it since the problem has been handled with almost no time delay.

If the remote does not resolve the sensing failure before t_s , then execution of the behavior is suspended. Ideally, suspension would mean that the robot would assume a fall-back or “defensive” state, allowing it to remain stationary and continue autonomous or cooperative exception handling. Unfortunately, the robot may not be able to maintain this fall-back state indefinitely; other behaviors or overarching mission parameters which are not affected may need to move the robot away from the sensing region where the failure occurred, disrupting its ability to analyze the cause of the failure. Consider the operation of a mobile robot in a highly radioactive environment. If the robot has CCD cameras, it will want to reduce unnecessary exposure to hard radiation. If the robot is not making progress on its task, it may be part of its mission to return to a shielded area.

The upper bound on how long the system can tolerate the suspension of the behavior before it has to effectively abort it is designated as t_a . If the remote is able to continue its exception handling in the interval between t_s and t_a , the operator must still be informed that the remote has entered the fall-back state. If the operator is busy and the time remaining until a behavior abort is long, the operator may choose to let the system continue to exhaust its autonomous capabilities before requiring human interaction.

This paper proposes that t_s be the deadline for the remote exception handling activity and t_a be a factor influencing the intelligent assistant data collection and presentation activities. This arrangement is practical and produces a reasonable overall system response, as can be seen by the canonical cases below.

- $t_s = 0$. In this instance, the behavior at the remote cannot operate for any length of time in a dead-reckoning mode. The remote exception handler immediately transfers control to the local without attempting to solve it autonomously. This has the advantage of notifying the operator that a behavior has been suspended. It does not interfere with autonomous exception handling, since the remote handler is a subset of the local and the local can instruct the remote to continue classification and recovery under the intelligent assistant’s supervision.

This case exemplifies what would happen if the remote perceptual process encounters a problem immediately upon instantiation and so has no belief in the percept (and thereby no basis for dead-reckoning) or if the remote’s survival depended on the behavior (for example, it might be deemed dangerous for the remote to attempt to navigate, not matter for how short a time, without the sensing for obstacle avoidance).

- $0 < t_s < t_a$. This is the nominal case, where the remote has some time available for exception handling without local supervision. One of three events might transpire during the time until t_s : the failure may be successfully classified and the remote recovers autonomously; the classification process may reach a point where it can go no further without human assistance and voluntarily transfers control to the local; or the classification process may still be active but the deadline is reached and control is by necessity passed to the local. In the last case, the local intelligent assistant can instruct the remote to continue its autonomous exception handling activities, but the operator is aware that the behavior has been suspended.

- $t_a = 0$. This condition could arise when the remote is operating under safety-critical constraints and any change in the situation requires human cognizant. Control would be passed immediately to the local and the all exception handling done under the direct supervision of the operator.
- $t_s \geq t_a$. In this situation, the remote is prevented from operating as long as might be theoretically possibly (t_s) due to some other consideration which set $t_a < t_s$. Control must be immediately passed to the local, even though the remote could operate in dead reckoning mode for $|t_s - t_a|$, in order to insure that the local will some time to gather and store any relevant data prior to aborting the behavior.

The interval $|t_a - t_s|$ is of particular importance to the local when the remote exception handler cannot recover. A large interval indicates that the remote can safely sit and wait for further directions from the local. A small interval serves as a warning that the remote may have to move away, that after that time, the local may not be able to request real time sensor data for help in isolating the failure(s). As a result, $|t_a - t_s|$ determines if the local requests all possible sensor data from the remote, regardless of communication bandwidth cost, in order to be sure and have it if the operator needs it. Furthermore, it can influence choice of strategy employed by the intelligent assistant; for example, displaying sensor data at a lower resolution in order to see if the operator can immediately identify the problem. On the other hand, if $|t_a - t_s|$ is large, the intelligent assistant is under no pressure to violate it's goal of minimizing communication between the systems. It can instruct the remote to continue its autonomous exception handling capabilities or request data on demand from the operator.

3.2 Involvement of Operator

The operator has three levels of supervisory involvement in exception handling in the VIA-SFX architecture. First, the operator does not need to participate in exception handling activities if (1) the remote is performing autonomous exception handling prior to t_s or (2) the intelligent assistant is continuing the remote's autonomous exception handling in the interval $|t_a - t_s|$. The operator is informed that exception handling has commenced but does not require the operator's attention. If the failure is resolved autonomously, the success will be posted. Again, the operator does not necessarily have to attend to that posting, and can continue to focus on other supervisory activities. Second, the operator may have cooperative supervisory duties. These would occur when (1) neither the intelligent assistant nor the remote was able to recover from the failure autonomously or (2) a rapidly changing situation requires the operator to be aware of what is happening. In these cases, the VIA-SFX interface would assist the operator. Third, the operator may assume total supervisory control of the remote at any time.

3.3 Computation of t_s

t_a is supplied by either the task manager software or the human operator. t_s is a function of the last "good" state of sensing and the projected activity of the robot so must be computed by the remote at the time of failure.

A behavior can continue as long as its certainty in task performance is acceptable. These measures of system uncertainty are generally viewed as decaying with time: the longer the behavior operates in a "dead reckoning" mode, the less certain the system is. For example, the more a remote moves without some external localization, the more uncertain its position is. The rate of increase of that uncertainty depend on the particular hardware configuration, the characteristics of the operating environment, and what the remote does between observations. A remote sitting still in a static world may have no reason to discount its previous observations, no matter how long an absolute time period has passed.

Uncertainty shows up in 2 forms: belief in the recognition of the correct object and measurement error. Consider a remote attempting to locate itself with respect to a specific landmark. It must believe that it is perceiving that landmark in order to proceed with extracting relative position. The position measurement will have some error associated with it. The estimation of t_s must take into account how long the remote can both track on the object of interest and operate within measurement error. These two components are treated as independent sources of uncertainty; algorithms for relative position can precisely measure the distance to the wrong landmark.

The time remaining to suspend the behavior t_s due to lack of certainty is the minimum of the time that the system can go without recognizing the object, t_{rec} , and the time that the system can operate within measurement error, t_{error} :

$$t_s = \min(t_{rec}, t_{error})$$

For example, consider the remote attempting to place itself 1 foot from a specific landmark. Given the measured distance remaining to the landmark, the estimated error, and the projected velocity, the remote can compute how long it can operate safely before it must update its position. But there will also be a constraint on how far the robot can move without running the risk of mistaking the landmark for another nearby landmark; if the remote is fixated on a pair of lines, how far can it move without resensing and still believe that it is tracking the same landmark. In situations where the landmark or object is unique (e.g., the only blue object in the area), recognition is easy to maintain and measurement error will dominate the computation of t_s . In cluttered areas such as a toxic waste dump where the percept might be a particular steel drum in a jumble, recognition may be less certain and t_{rec} will be t_s .

4 A Generalization of Dempster-Shafer Theory for Temporal Updating

Estimating measurement error is well understood, while projecting recognition uncertainty is not. Recognition uncertainty is typically treated as an evidential reasoning problem. Belief over time is the combination of prior belief with current belief. The two prevalent theories of evidence, Bayesian updating and Dempster-Shafer theory, weight the past equally with the present. In the case where a sensing failure has occurred and the current belief is missing or “don’t know”, this weighting effectively prevents any loss of belief over time; this is counterintuitive and undesirable. The most common solution is to introduce an ad hoc decay factor within a probability model to discount the previous belief during updating [5,6]. A high decay rate says that the state is changing rapidly and therefore the remote cannot safely operate for long without data.

Evidential reasoning for estimating belief over time in VIA-SFX is done using a variation on the Dempster-Shafer (DS) theory of evidence [13]. As noted by [17], Dempster’s rule of combination is only one possible rule of combination. Any rule of combination which takes two combinable belief functions and always yields a third is permissible. Variations of Dempster’s rule have been proposed, most notably by Smets [15] and Zadeh [18]. We can rewrite Dempster’s rule for combining belief into what we call a generalized form, where $m(a_i)$ represents the belief for a_i :

$$m_{Bel_1 \oplus Bel_2}(a_k) = \frac{\sum_{a_i \cap a_j = a_k} f(m(a_i), m(a_j))}{\sum_{a_i \cap a_j \neq \phi} f(m(a_i), m(a_j))}, \quad a_k \neq \phi, \quad (1)$$

If $f(m(a_i), m(a_j)) = m(a_i) \times m(a_j)$, then the above is Dempster’s rule. By changing f , new rules can be created. Wilson [17] maintains that different rules of combination reflect the operationalization of different assumptions. Rather than apply a decay factor, our approach has been to construct a new rule of combination for the situation of a remote taking readings over time. There are two assumptions implicit in the new rule that are different from Dempster’s rule. First, observations over time from the same sensor about the same target can not be considered independent. If Dempster’s rule of combination is applied to a series of belief functions extracted from the same sensor over time, counter-intuitive combined belief functions occur. Second, we note that the order of the presentation of evidence is non-commutative for the EVAHR application. Consider a series of belief functions where the first offers strong evidence that the target is object p , the second strong evidence, the third weak, the fourth strong, the fifth strong, the sixth weak, and the seventh strong. Compare this with a different sequence of belief in p : strong, strong, strong, strong, strong, strong, weak, weak. The first sequence may indicate noise in the observations, while the second may be a sign of a sudden change (e.g., sensor failure).

The case where:

$$f(m(a_i), m(a_j)) = [m(a_i)m(a_j)]^n$$

and $n > 1$ produces an evidential decay that follows the intuition. As with [5], the choice of values for the decay parameter is made by a knowledge engineer.

The time remaining until the belief decays below the minimum acceptable level can be estimated in a number of ways. The simplest is:

$$t_{rec} = n_{obs} * \frac{time}{observation}$$

where n_{obs} is the number of missed observations before the total belief $belief_{current}$ drops below $belief_{min}$. If it is assumed that missing belief will be represented as the vacuous belief function (all belief assigned to “don’t know”) and that all belief is either belief in the percept or “don’t know”, then the last observed belief $belief_{total}$ will decay according to:

$$belief_{current} = (belief_{last})^{n_{obs}-2}$$

If the $belief_{min}$ is substituted for $belief_{current}$, the number of observations, n_{obs} , of “don’t know” that it will take until the current belief decays to $belief_{min}$ can be computed by:

$$n_{obs} = 2 + \log\left(\frac{\log(belief_{min})}{\log(belief_{last})}\right) / \log(n)$$

In contrast to t_{rec} , the estimation of t_{error} depends on the behavior itself and what is known about the measurement error.

5 Demonstrations

Work is currently in progress on implementing exception handling and the use of the generalized Dempster-Shafer theory to compute when to swap control from the remote to the local. This section presents three preliminary exercises of the exception handler operating independently of a mobile robot but using actual sensor data.

Each exercise simulates a mobile robot navigating with a reactive **move-to-goal** behavior to a landmark selected by the operator, in this case the blue square, shown in Figure 1. The perceptual process consists of 2 redundant sensing plans; one uses video camera C_1 and the other uses C_2 to recognize the landmark region and extract relative distance. Recognition is based on how well the seven normalized moments extracted from the blue segmented region match the expected shape. All processing is done on a Sun Sparcstation 2 with a Videopix framegrabber. The minimum belief, $belief_{min}$, was chosen to be 0.2 to allow sufficient time for the GTD method to be demonstrated. A constant velocity and translational error rate was assumed for estimating the time until a collision with the goal.

5.1 Sensor Malfunction

In the first demonstration, the camera C_1 was successfully observing the blue triangle with a belief of 0.80 when a sensing hardware malfunction was introduced by disconnecting the input to the frame grabber. As shown in the program execution trace in Figure 2, this results in a missing observation and triggers a failure condition. Execution of the perceptual process is suspended and the time remaining until the entire behavior must be suspended, t_s , is computed. Note that t_s is the minimum of the estimates for evidential decay and amount of time that the agent can safely dead reckon. Control is now passed to the exception handler, which notes that a hardware failure has occurred and recovers by employing the back up video camera C_2 . Essentially the exception handler’s first hypothesis is that a hardware failure has occurred and checks for any explicit errors.

5.2 Environmental Change

The second demonstration invoked the remote exception handler with a failure caused by an environmental change: turning off the lights. The camera was successfully observing the blue triangle with a belief of 0.95 when lens caps were placed over both cameras (simulating the lights going out). As shown in the program execution trace in Figure 2,

this results in a missing observation and triggers a failure condition. Execution of the perceptual process is suspended and the time remaining until the entire behavior must be suspended, t_s , is computed. Note that in this demonstration the last good belief of 0.95 is higher than in the previous case (0.80) and is reflected by the different values of t_s , 17.8 and 27.4 seconds respectively.

Control is now passed to the exception handler, which generates a set of failure hypotheses and tests that can be completed in the remaining time. The exception handler hypothesizes that either a sensor is malfunctioning or an environmental change has occurred. It tests for a malfunctioning sensor by comparing its average image intensity with that of the redundant sensor. At the same time, it tests for a change in lighting conditions. It notes that both cameras report an average intensity below the threshold for the expected lighting, thereby signalling a change in the environment. Because there are no alternative sensors for the task, the system cannot recover autonomously and the operator must be notified.

5.3 Errant Expectation

In the third demonstration, the operator selected a blue triangle as the landmark but only a blue square was present. This represents the case where the remote has errant expectations and so sensing will fail immediately. As shown in Figure 4, the perceptual process cannot generate any belief in presence of the blue triangle, because it had no prior belief. Since the belief is already below the minimum that it can decay to, the time t_s is 0. Therefore, the behavior must be suspended automatically and control passed to the local for resolution. The local elects to allow the remote to employ its autonomous exception handling and it classifies the cause of the failure as an errant exception because the cameras appear to be operating correctly.

5.4 Discussion

The above demonstrations are limited but do suggest that the impact of time can be effectively incorporated into an intelligent assistance architecture. The demonstrations also indicate that using a special rule of combination for estimating the time available is promising. However, as currently formulated, it uses a heuristically chosen value for the decay parameter, n . Until a formal method of selecting n is found, the advantages of using the special rule is limited. To this end, we are investigating the use of Kalman filtering to relate the change in measurement certainty with change in belief. The paper has focused on the definition of reasonable deadlines for exception handling and how to estimate those time limits. It has ignored the issue of whether it is possible, or even desirable, to classify and recover from sensing failures using an *anytime* algorithm [4]; more work is needed in this area.

6 Conclusions

In conclusion, this paper has reported on work in progress in extending the ability of the VIA-SFX cooperative assistance architecture. These modifications are expected to improve its exception handling capability and ability to function effectively in real-time. Exception handling for sensing failures is performed both at the remote and at the local. The remote detects sensing failures while executing reaction plans for sensing. Once a failure is detected, the remote estimates the time remaining until the behavior must be suspended; this becomes the time limit on exception handling at the remote. If the remote cannot complete exception handling in this time or needs assistance, exception handling responsibility is passed to the local. An intelligent assistant then facilitates human intervention, either directing the remote without human assistance or coordinating data collection and presentation to the operator within time limits imposed by the mission.

The strategy of using real-time deadlines on exception handling is expected to lead to more effective use of remotes for several reasons. First, the overall actions of the remote should be smoother in the face of sensing failures. The remote can continue operating in a dead reckoning mode giving the exception handler a chance to "catch up" and recover. However, the dead reckoning mode may not be possible or even desirable for all behaviors. It can be eliminated by specifying a low time limit for aborting the behavior, essentially forcing the remote to immediately transfer control to the local. Since the remote exception handling capabilities are a subset of the local's, the local can perform the

same tests. Second, since the remote attempts to resolve the problem first, then the local, the system does not bother the operator with easily classified failures, preventing cognitive fatigue. If the remote may a fall-back state that it can assume, the operator can manage his/her attention between multiple remotes as it is unlikely that all remotes will fail and require immediate attention at the same time.

The time remaining until the behavior is aborted and the remote might leave the region where the sensing error occurred has an important impact on the data collection strategy and communication demands employed by the intelligent assistant at the local system. The intelligent assistant in VIA-SFX attempts to reduce communication demands where ever possible by filtering data and requesting updates of only what is needed to confirm the current hypotheses. This is a desirable strategy in situations where communication bandwidth is limited and the remote can remain for an indefinite period of time in the same position; the local can gather data from the remote on demand. However, if the robot is likely to move, the human and the intelligent assistant may not be able to identify the problem immediately. Therefore, the intelligent assistant may demand all data as quickly as possible and store it for later. This should also result in the remote remaining as near operational as possible while not interfering with exception handling.

The modifications to VIA-SFX illustrate that it is possible for a teleoperation architecture to support an evolution to increasing levels of autonomy. VIA-SFX makes use of results from research in exception handling for autonomous mobile robots by adding the portions which work reliably to the semi-autonomous remote and supplementing those that are not reliable with human assistance at the local. This is truly a "best of both worlds" scenario, where new levels of autonomy are added and cooperative assistance overcomes the hard problems of resolving contention and identifying compound failures. Although this paper concentrates on exception handling at the remote, it should be noted that the local system can further support the evolution of intelligence by learning classification tests and new recovery strategies from observing the human operator.

Current work is concentrating on implementing and testing the basic modifications to VIA-SFX. Additional research is needed in the planning and scheduling of exception handling activities within real-time constraints. Another area that is currently being explored is adapting the speed of the remote to increasing t_s so that it will not reach the deadline before exception handling is expected to be completed. This would allow the remote to "limp along" and completely exercise all its exception handling capabilities before necessitating operator assistance.

Acknowledgments

This work was supported in part by NSF/ARPA grant IRI-9320318 and NASA SBIR NAS.9-19040 in conjunction with Robotics Research Harvesting. The authors would like to thank Jim Hoffman for his initial investigation of generalized Dempster-Shafer theory, Elizabeth Nitz for data collection, and Dr. Marcel Schoppers, Jim MacMillan, and John Blitch for their helpful discussions.

References

- [1] Chavez, G. T., and Murphy, R.R., 1993. "Exception Handling for Sensor Fusion" SPIE Sensor Fusion VI, Boston, MA, Sept. 7-10, 1993, pp. 142-153.
- [2] Coiffet, P. and Gravez, P., 1991. "Man-Robot Cooperation: Towards an Advanced Teleoperation Mode". In S. G. Tzafestas (Ed.), *Intelligent Robotic Systems*. Marcel Dekker: New York, pp. 593-636.
- [3] Edwards, G.R., Burnard, R.H., Bewley, W.L., and Bullock, B.L., "The Ground Vehicle Manager's Associate", *Tech Report AIAA-94-1249-CP*, 1994, pp. 520-526.
- [4] Dean, T., and Boddy, M., "An analysis of time-dependent planning," in *Proceedings AAAI-88*, 1988, pp. 49-54.
- [5] Dean, T. and Kanazawa, K., "Persistence and Probabilistic Projection", *IEEE Transactions on Systems, Man, and Cybernetics*, Vol. 19, No. 3, May/June 1989, pp. 574-585.

- [6] Driankov, D. and Lang, J., "Possibilistic decreasing persistence", *Uncertainty in Artificial Intelligence*, Proceedings of the Ninth conference, Heckerman and Mandani eds., Morgan Kaufman, July 1993, pp. 469-476.
- [7] Gao, Y. and Durrant-Whyte, H. F., *Multi-Sensor Fault Detection and Diagnosis Using Combined Qualitative and Quantitative Techniques*, 1994 IEEE Conference on Multisensor Fusion and Integration for Intelligent Systems, Las Vegas, NV.
- [8] Ferrell, C., "Robust Agent Control of an Autonomous Robot with Many Sensors and Actuators," *MS thesis*, Department of Electrical Engineering and Computer Science, Massachusetts Institute of Technology, May 1993.
- [9] Fernandez, M. and Durrant-Whyte, H. F., *A Failure Detection and Isolation Algorithm for a Decentralized Multi-sensor System*, 1994 IEEE Conference on Multisensor Fusion and Integration for Intelligent Systems, Las Vegas, NV.
- [10] Naidu, S., Zafriou, E., and McAvoy, T., *Use of Neural Networks for Sensor Failure Detection in a Control System*, 1989 American Control Conference, Pittsburgh, PA.
- [11] Rogers, E. 1992. *Visual Interaction: A Link Between Perception and Problem-Solving*. Technical Report No. GIT-CC-92/59, College of Computing, Georgia Institute of Technology, Atlanta, GA 30332.
- [12] E. Rogers and Murphy, R.R., 1994. "Tele-assistance for Semi-Autonomous Robots", *AIAA/NASA Conference on Intelligent Robots for Field, Factory, Service, and Space*, Houston, March 21-24, 1994, pp. 500-508.
- [13] Shafer, S. A., Stentz, A., Thorpe, C. E., "An Architecture for Sensor Fusion in a Mobile Robot." Proceedings of *IEEE International Conference on Robotics and Automation*, San Francisco, CA, 1986, pp. 2002-2011.
- [14] Simmons, R., Davis, R., "Generate Test and Debug: Combining Associational Rules and Causal Models." *Proceedings of 10th International Joint Conference on Artificial Intelligence*, August 1987.
- [15] Smets, P., "The Combination of Evidence in the Transferable Belief Model", *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 12, no. 5, May 1990, pp. 447-458.
- [16] Weller, G. A., Groen, F. C. A., Hertzberger, L. O., "A Sensor Processing model Incorporating Error Detection and Recovery." In T. C. Henderson, editor, *Traditional and Non-traditional Robotic Sensors*, Maratea, Italy, Aug. 28- Sep. 2 1989, Springer-Verlag.
- [17] Wilson, N., "The Assumptions Behind Dempster's Rule," *Proceedings of the Ninth Conference on Uncertainty in Artificial Intelligence*, Washington, DC, July 9-11, 1993, pp. 527-534.
- [18] Zadeh, L.A., "A Simple View of the Dempster-Shafer Theory of Evidence and its Implication for the Rule of Combination", *AI Magazine*, Summer, 1986, pp. 85-90.

Figure 1: Blue triangle landmark as seen from video camera 1.

```

<successful data collection and fusion>
Fusion step returned SUCCESS
Total Belief: Support: 0.80 Against: 0.00 Don't Know: 0.20 Conflict: 0.00

<sensing failure has now occurred on data collection>

<immediate classification due to hardware error message>
Could not open camera device.
*****
*** TIME CALCULATIONS ***
Last good belief in the percept = 0.800000
Given n = 1.200000, average seconds per observation = 1.200000,
    time remaining = 17.761688 seconds.
Given distance = 5.000000, current velocity = 0.200000, and translational error rate = 0.250000
    time remaining = 6.250000 seconds.
TOTAL TIME REMAINING = 6.250000 seconds.
*****
Welcome (bienvenidos) to error recovery!!!

Description/sensor nodes in original sensing plan
Desc/sensor node: 0 color camera

Color video hardware error recovery
<replace bad camera with other camera if available>

Description/sensor nodes in repaired sensing plan
ERROR RECOVERY COMPLETE
  
```

Figure 2: Annotated program trace for sensor malfunction.

```

<successful data collection and fusion>
Total Belief: Support: 0.95 Against: 0.00 Don't Know: 0.05 Conflict: 0.00

<failure is now detected>
Belief: Support: 0.00 Against: 0.00 Don't Know: 1.00 Conflict: 0.00
  PREPROCESSING STEP FAILURE
*****
*** TIME CALCULATIONS ***
Last good belief in the percept = 0.950000
Given n = 1.200000, average seconds per observation = 1.200000,
  time remaining = 27.438582 seconds.
Given distance = 5.000000, current velocity = 0.200000, and translational error rate = 0.250000
  time remaining = 6.250000 seconds.
TOTAL TIME REMAINING = 6.250000 seconds.
*****
Welcome to error classification!!!
<2 hypotheses generated: sensor malfunction and environmental change>
Color camera diagnostic
Check intensity
<carries out tests>
Test 1: sensor diagnostic function
      Found a good color sensor, number 0. Ok to run other tests.
Test 2: testing for environmental change
      *****VIOLATION: COLOR MINIMUM INTENSITY THRESHOLD*****
Intensity change confirmed
ERROR CLASSIFICATION COMPLETE!!!
*****
Welcome (bienvenidos) to error recovery!!!
Intensity change recovery

```

Figure 3: Annotated program trace for environmental change.

```

NO BLUE REGION FOUND
Total belief: Support: 0.00 Against: 0.00 Don't Know: 1.00 Conflict: 0.00
  PREPROCESSING STEP FAILURE
*****
*** TIME CALCULATIONS ***
Last good belief in the percept = 0.000000
Given n = 1.200000, average seconds per observation = 1.200000,
  time remaining = 0.000000 seconds.

Given distance = 5.000000, current velocity = 0.200000, and translational error
rate = 0.250000
  time remaining = 6.250000 seconds.
TOTAL TIME REMAINING = 0.000000 seconds.
*****
<transfer control to local, which continues autonomous exception handling>
<2 hypotheses generated: sensor malfunction and environmental change>
Color camera diagnostic:
Check intensity:
<carries out tests>
Test 1: sensor diagnostic function
      Found a good color sensor, number 0. Ok to run other tests.
Test 2: testing for environmental change
      Environmental intensity is ok, detected with sensor 0.
No errors classified -- Robot is in a bad position.
***** ERROR CLASSIFICATION COMPLETE *****
<recovery requires replanning>

```

Figure 4: Annotated trace for errant expectation.