Belief Space Metareasoning for Exception Recovery

Justin Svegliato[1], Kyle Hollins Wray[1,2], Stefan J. Witwicki[2], Joydeep Biswas[1], and Shlomo Zilberstein[1]

***Abstract*— Due to the complexity of the real world, autonomous systems use decision-making models that rely on simplifying assumptions to make them computationally tractable and feasible to design. However, since these limited representations cannot fully capture the domain of operation, an autonomous system may encounter unanticipated scenarios that cannot be resolved effectively. We first formally introduce an introspective autonomous system that uses belief space metareasoning to recover from exceptions by interleaving a main decision process with a set of exception handlers. We then apply introspective autonomy to autonomous driving. Finally, we demonstrate that an introspective autonomous vehicle is effective in simulation and on a fully operational prototype.**

I. Introduction

Autonomous systems have been deployed across many applications, such as autonomous vehicles [1], search and rescue robots [2], and space exploration rovers [3]. Simply put, these systems make decisions based on decision-making models that have inherent limitations. For example, a self-driving car may not be capable of driving on poorly marked roads or in heavy rain. Hence, in order to guarantee reliable operation, some restricting assumptions must be satisfied. This reduces the complexity of designing and verifying solutions for efficient planning and execution [4]. However, as a result of incomplete decision-making models, these systems may encounter a wide range of unanticipated scenarios that cannot be resolved optimally, feasibly, or even safely.

A simple approach to ensuring the necessary conditions of normal operation is to place the entire responsibility on the operator deploying the autonomous system. However, although relying on human judgment can improve performance [5], it is desirable to limit human involvement when the conditions of normal operation are violated. In fact, most of this responsibility should ideally be delegated to the autonomous system. We therefore offer a metareasoning framework that activates secondary decision-making models designed to restore normal operation with or without human involvement given any violation of its necessary conditions.

Despite tremendous progress in metareasoning centered on monitoring and controlling anytime algorithms [6], there have been few attempts to build autonomous systems that use metareasoning to recover from exceptional situations effectively. Such a system presents many challenges. First, because an unanticipated scenario is not captured by a decision-making model by definition, the model does not have the information needed to resolve that exception. Next, while a decision-making model can be extended to capture a set of unanticipated scenarios, a naïve approach will exponentially grow the complexity of the model with the number of exceptions. This is often infeasible for complex exceptions in real world applications. Finally, since a decision-making model cannot capture every unanticipated scenario, there will always be exceptions that cannot be resolved properly.

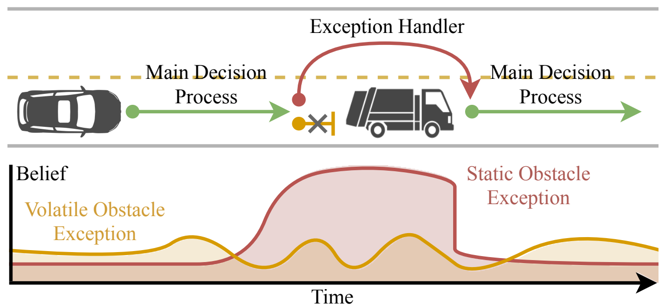

Fig. 1. An intuitive view of an introspective autonomous system.

Recent work in exception recovery has focused on fault diagnosis—detecting and identifying faults—during normal operation. For instance, many approaches diagnose faults by exploiting methods that use particle filters [7] or multiple model estimation with neural networks [8]. While these approaches *detect* and *identify* exceptions, they do not offer a comprehensive framework that can also *handle* exceptions without human assistance. Building on recent work in fault diagnosis, our goal is to provide an exception recovery framework that detects, identifies, and handles exceptions.

We offer an approach for building *introspective autonomous systems* that use belief space metareasoning for exception recovery. In Figure 1, this approach makes decisions by interleaving decision processes: a main decision process designed for normal operation and a set of exception handlers. As the system completes its task, it activates its decision processes based on a belief over potential exceptions. If its belief suggests normal operation, it executes its main decision process. Otherwise, if its belief indicates exceptional operation, it suspends its main decision process and executes an exception handler. It can also gather information or transfer control to an operator given uncertainty in its belief.

Our key contributions are: (1) a formal definition of an introspective autonomous system and its properties, (2) a framework for profiling decision processes, (3) an application of introspective autonomy to autonomous driving, and (4) a demonstration that an introspective autonomous vehicle is effective in simulation and on a fully operational prototype.

This work was supported in part by a National Science Foundation Graduate Research Fellowship DGE-1451512, the National Science Foundation grants IIS-1724101 and IIS-1813490, and Nissan Motor Company.

[1]College of Information and Computer Sciences, University of Massachusetts, Amherst, MA, USA. Emails: {jsvegliato, wray, joydeepb, shlomo}@cs.umass.edu

[2]Alliance Innovation Lab Silicon Valley, Santa Clara, CA, USA. Emails: {kyle.wray, stefan.witwicki}@nissan-usa.com

II. INTROSPECTIVE AUTONOMY

Given the complexity of the real world, autonomous systems have traditionally relied on decision-making models that depend on many simplifying assumptions to facilitate planning and execution [4]. These systems, however, can encounter unanticipated scenarios that cannot be resolved effectively. For instance, an autonomous vehicle can encounter different types of obstacles along its route. Achieving the complete potential of autonomous systems therefore requires the ability to recover from exceptional situations [9].

A. Background

A *partially observable Markov decision process* (POMDP) is a formal decision-making model for reasoning in partially observable, stochastic environments [10]. A POMDP can be described as a tuple $\langle S, A, T, R, \Omega, O \rangle$, where S is the set of states of the world, A is the set of actions of the agent, $T : S \times A \times S \rightarrow [0, 1]$ is the transition function that maps each state $s \in S$ and action $a \in A$ to the probability of ending up in state $s' \in S$, $R : S \times A \times S \rightarrow \mathbb{R}$ is the reward function that maps each state $s \in S$ and action $a \in A$ to the expected immediate reward gained in $s' \in S$, Ω is the set of observations of the agent, and $O : S \times A \times \Omega \rightarrow [0, 1]$ is the observation function that maps each state $s \in S$ and action $a \in A$ to the probability of observing observation $\omega \in \Omega$.

In a POMDP, the agent does not necessarily know the true state of the world at any given time. Instead, the agent makes noisy observations that reflect its state and action. To represent its uncertainty, the agent maintains a belief state $b \in B$, a probability distribution over all states, where B is the space of all belief states. Initially, the agent begins with an initial belief state $b_0 \in B$. After performing an action $a \in A$ and making an observation $\omega \in \Omega$, the agent updates its current belief state $b \in B$ to a new belief state $b' \in B$ using the belief state update equation $b'(s'|b, a, \omega) = \eta O(a, s', \omega) \sum_{s \in S} T(s, a, s') b(s)$, where η is the normalization constant $\eta = Pr(\omega|b, s)^{-1}$.

At each time step, the agent selects an action based on its current belief state using a policy $\pi : B \rightarrow A$ that maps a belief state $b \in B$ to an action $a \in A$. A policy π induces a value function $V^\pi : B \rightarrow \mathbb{R}$ that represents the expected cumulative reward of each belief state and an optimal policy π^* maximizes this reward. Note that there are many solution methods that calculate or approximate an optimal policy [11].

B. Introspective Autonomous Systems

In order to recover from exceptions, an *introspective autonomous system* maintains a *belief* over *potential exceptions*. The system uses this belief to reason about how to interleave *decision processes*. Naturally, the decision processes include the *regular process*, which makes decisions using a model designed for a particular task. If the system believes there is not an exception, suggesting *normal operation*, it executes the regular process. The decision processes also include a set of *exception handlers*, which make decisions based on a model designed for a specific exception. If the system believes there is an exception, indicating *exceptional operation*, it executes an exception handler. In a self-driving car, the regular process and exception handlers could be for navigation and obstacle handling respectively. In short, given its belief over potential exceptions, the system alternates between regular decision making and exception handling.

An introspective autonomous system uses its belief to encapsulate its uncertainty over whether or not the assumptions of normal operation have been violated. This belief belongs to a space of beliefs that is partitioned into distinct regions associated with different decision processes. Typically, the regular process will correspond to a large region that suggests normal operation while each exception handler will correspond to a small region that indicates exceptional operation. The system simply executes whichever decision process is associated with the region containing its current belief.

The execution of an introspective autonomous system can be viewed as a two-level hierarchy of decision processes. The high-level decision process is the introspective autonomous system while the low-level decision process is the regular process or an exception handler. When the introspective autonomous system activates the regular process or an exception handler, which is a form of an option [12], it transfers control to that decision process until a condition is met: the system executes the regular process *for a fixed duration* and an exception handler *until termination*. After the decision process meets this condition, the introspective autonomous system resumes execution once again.

All decision processes generate an *indicator* describing its status after execution. An introspective autonomous system uses each indicator to update its belief over potential exceptions. The information offered by the indicator, however, depends on the decision process. Because the objective of the regular process is to complete a specific task, it can generate a success or failure signal (e.g., the route has been or cannot be completed) or a signal that suggests whether or not an exception has been encountered (e.g., an obstacle has been encountered). However, since the goal of an exception handler is to resolve a particular exception, it can generate a success signal (e.g., the obstacle is no longer blocking) or a signal that suggests different modes of failure (e.g., the obstacle is still blocking). Note that it is possible for a decision process to generate other indicators as well.

An introspective autonomous system always has a default exception handler, called the *human assistance exception handler*, that is assumed to handle any exception not yet linked to an exception handler. In particular, if there is no exception handler designed for a specific exception, it will execute the human assistance exception handler as a general form of exception handling. For example, if a self-driving car is blocked by an unrecognized obstacle, it will transfer control to the driver rather than use an obstacle handler. Thus, as new exception handlers are added to the system, its reliance on human assistance will diminish appropriately.

An introspective autonomous system also has standard attributes in addition to exceptions, decision processes, and indicators. That is, the system has *standard states*, *standard actions*, and *standard observations*. As an example, along

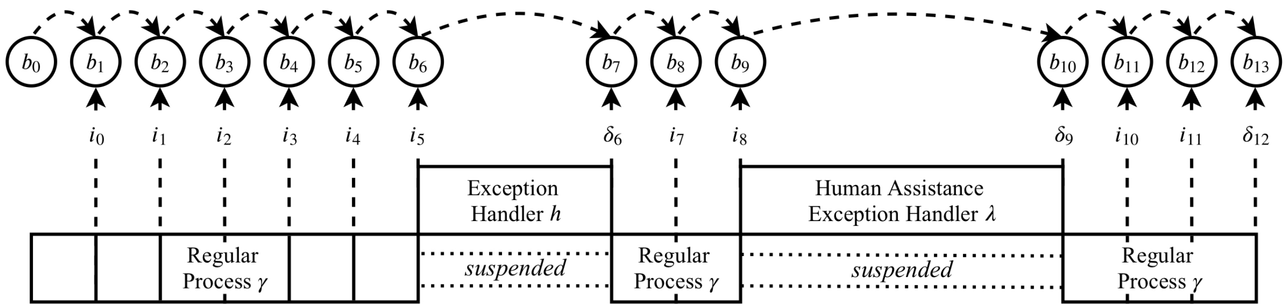

Fig. 2. An introspective autonomous system interleaving the regular process with exception handlers based on its belief.

with its corresponding standard observations, an autonomous vehicle could have standard states for wait time and standard actions for waiting and edging in order to facilitate information gathering about normal and exceptional operation. The system has a *standard transition function*, *standard reward function*, and *standard observation function* as well.

We now provide a formal description of an introspective autonomous system by extending a POMDP below.

Definition 1. *An* ***introspective autonomous system*** *can be described as a tuple* $\langle E, P, I, S, A, T, R, \Omega, O \rangle$*, where*

- E *is a set of exceptions (denoted as* e_i*),*
- P *is a set of decision processes (denoted as* p_j*),*
- I *is a set of indicators (denoted as* i_k*),*
- $S = S \times E$ *is a set of factored states: a standard state set* S *and an exception set* E*,*
- $A = A \cup P$ *is a set of actions: a standard action set* A *and a decision process set* P*,*
- $T : S \times A \times S \to [0, 1]$ *is a transition function composed of a standard transition function* $T : S \times A \times S \to [0, 1]$*, a transition profile* $_p : S \to \Delta^{|S|}$*, and an exception profile* $_p : S \to \Delta^{|E|}$*,*
- $R : S \times A \times S \to \mathbb{R}$ *is a reward function composed of a standard reward function* $R : S \times A \times S \to [0, 1]$ *and a cost profile* $_p : S \to \mathbb{R}$*,*
- $\Omega = \cup I$ *is a set of observations: a standard observation set and an indicator set* I*, and*
- $O : S \times A \times \Omega \to [0, 1]$ *is an observation function composed of a standard observation function* $O : S \times A \times \to [0, 1]$ *and an indicator profile* $_p : S \to \Delta^{|I|}$*.*

The exception set E requires normal operation . The decision process set P requires the regular process and the human assistance exception handler . The indicator set I requires a success signal and a failure signal . The automated exception handler set, without the regular process or the human assistance exception handler, is denoted as H.

There are several principles that should be followed when building an exception handler. First, in order to cover as many exceptions as possible, an exception handler should be general rather than narrowly specialized. For instance, a self-driving car should have exception handlers for broad classes of obstacles. Next, during the handling of an exception, an exception handler should meet the requirements of the regular process. Finally, by monitoring the conditions for which it has been activated, an exception handler should terminate itself if it deems its execution unnecessary.

Following recent work on metareasoning for anytime algorithms [13], [14], it is natural to view an introspective autonomous system as a meta-level controller that *monitors* and *controls* the regular process at fixed intervals. That is, the system *monitors* the regular process by maintaining a belief over whether or not the assumptions of normal operation have been violated and *controls* the regular process by executing it or suspending it to execute an exception handler. As a meta-level controller, the systems weighs the likelihood of normal and exceptional operation with the cost of executing the regular process or an exception handler.

Figure 2 offers a natural view of an introspective autonomous system. Generally, in order to complete a task, the system runs different decision processes that generate indicators used to update its belief: it either activates the regular process for a fixed duration or an exception handler until termination. In this example, the system executes the regular process (with signals suggesting whether or not there is an exception). In between the executions of the regular process, the system executes first an exception handler and then the human assistance exception handler that both succeed (with success signals). The system continues to run until the regular process is successful (with a success signal).

C. Decision Process Profiling

Although a decision process can make decisions using a sophisticated model, an introspective autonomous system does not rely on its internal mechanisms. A decision process is instead summarized by a set of *profiles*. Each profile forms an abstraction over a feature of the decision process within the system. Note that these profiles are used to describe the transition, reward, and observation function of the system.

The first profile expresses how a decision process transitions through the *standard state space* by mapping a *factored state* to a probability distribution over all *standard states*.

Definition 2. *A* ***transition profile****,* $_p : S \to \Delta^{|S|}$*, gives the probability of ending up in state* $s' \in S$ *after executing the decision process* $p \in P$ *in state* $s \in S$*.*

The second profile captures how a decision process transitions through the *exception space* by mapping a *factored state* to a probability distribution over all *exceptions*.

Definition 3. *An* ***exception profile****,* $_p : S \to \Delta^{|E|}$*, gives the probability of ending up with exception* $e' \in E$ *after executing the decision process* $p \in P$ *in state* $s \in S$*.*

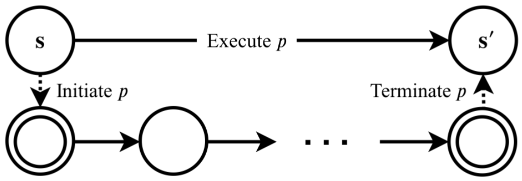

Fig. 3. The transition of an introspective autonomous system.

The third profile characterizes the *cost* of executing a decision process by mapping a *factored state* to an *expected immediate cost* of a decision process.

Definition 4. *A **cost profile**, ${}_p : S \to \mathbb{R}$, gives the expected cost of executing the decision process $p \in P$ in state $s \in S$.*

The fourth profile encapsulates how a decision process emits an *indicator* by mapping a *factored state* to a probability distribution over all *indicators*.

Definition 5. *An **indicator profile**, ${}_p : S \to \Delta^{|I|}$, gives the probability of observing an indicator $i \in I$ after executing the decision process $p \in P$ and ending up in state $s \in S$.*

Finally, putting all of these profiles together, we present the complete description of a decision process as follows.

Definition 6. *A **decision process**, $p \in P$, can be described as a tuple of profiles $\langle {}_p, {}_p, {}_p, {}_p \rangle$ that summarize its operation such that ${}_p$ is the transition profile, ${}_p$ is the exception profile, ${}_p$ is the cost profile, and ${}_p$ is the indicator profile.*

Note that each decision process can be viewed as using a policy derived from a decision-making model, such as a stochastic shortest path (SSP) problem, or a domain expert.

Figure 3 illustrates the transition of an introspective autonomous system during the execution of a decision process. Intuitively, while its internal mechanisms can be sophisticated, a decision process is simply an action that causes the system to transition through states in its state space. In this illustration, the system executes a decision process p starting in state s and ending in state s'. When the system activates the decision process in state s, it transfers control to that decision process. The decision process then transitions through states its own state space by performing actions in its own action space. After the decision process is done, it transfers control back to the system in state s'.

D. Dynamics

Now, by using the formal definition of a decision process, we can express the transition, reward, and observation functions of an introspective autonomous system. First, for the transition and reward functions, if the action is a decision process, the relevant profiles are used. Otherwise, the relevant standard function is used. Given a state $s = (s, e) \in S$, an action $a \in A$, and a successor state $s' = (s', e') \in S$, we describe the transition and reward functions below.

$$T(s, a, s') = \begin{cases} {}_a(s, s')\ {}_a(s, e') & \text{if } a \in P \\ T(s, a, s') & \text{otherwise} \end{cases}$$

$$R(s, a, s') = \begin{cases} {}_a(s) & \text{if } a \in P \\ R(s, a, s') & \text{otherwise} \end{cases}$$

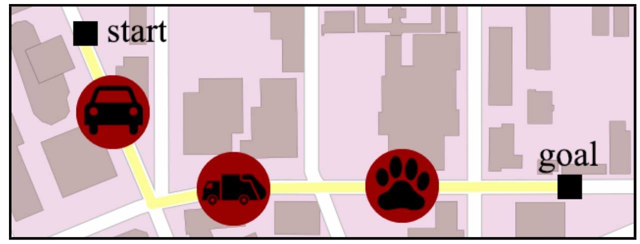

Fig. 4. An example route with several obstacles.

Moreover, for the observation function, if the action is a decision process and the observation is an indicator, the relevant profile is used. However, if the action is a standard action and the observation is a standard observation, the relevant standard function is used. Otherwise, the probability is nil. Given a successor state $s' = (s', e') \in S$, an action $a \in A$, and an observation $\omega \in \Omega$, we express the observation function as follows.

$$O(s', a, \omega) = \begin{cases} {}_a(s', \omega) & \text{if } a \in P \text{ and } \omega \in I \\ O(s', a, \omega) & \text{if } a \in A \text{ and } \omega \in \\ 0 & \text{otherwise} \end{cases}$$

E. Robustness

An introspective autonomous system enables the regular process to complete a task by handling exceptions with a set of exception handlers. These exception handlers are therefore essential to the effectiveness of the system. Accordingly, we define the main properties of an exception handler below.

Definition 7. *An exception handler, $h \in H$, is **strong** if it is guaranteed to handle a specific exception $e \in E$ for all states $s \in S$.*

Definition 8. *An exception handler, $h \in H$, is **conditionally strong** if it is guaranteed to handle a specific exception $e \in E$ for some states $s \in S$.*

Definition 9. *An exception handler, $h \in H$, is **weak** if it is not strong or conditionally strong.*

Finally, given all of these properties, we define the central property of an introspective autonomous system as follows.

Definition 10. *An introspective autonomous system is **robust** if there exists a strong or conditionally strong exception handler, $h \in H$, that is guaranteed to handle any exception $e \in E$ that may arise in any state $s \in S$ during operation.*

III. AUTONOMOUS DRIVING

We now provide an application of introspective autonomy to autonomous driving. In this domain, an autonomous vehicle must drive along a route from a start location to a goal location. However, as the autonomous vehicle progresses along this route, it can encounter different types of obstacles of increasing volatility: a *static obstacle* that remains stopped permanently, a *dynamic obstacle* that stops and goes repeatedly, and an *erratic obstacle* that behaves unpredictably. Figure 4 shows an example route with a static obstacle (the *car* icon), a dynamic obstacle (the *garbage truck* icon), and an erratic obstacle (the *paw* icon).

A. Navigation Problem

First, we consider the regular process. Its decision-making model is a navigation problem where the autonomous vehicle must drive along a route from a start location to a goal location with the assumption that there are no obstacles. More formally, for the regular process, $\in P$, the **navigation problem** can be represented by the tuple $\langle S, A, T, C, s_0, s_g \rangle$, where S is a set of states representing intersections, A is a set of actions representing road segments, $T : S \times A \times S \to [0, 1]$ is a transition function representing whether or not an intersection $s \in S$ is connected to an intersection $s' \in S$ by a road segment $a \in A$, $C : S \times A \times S \to \mathbb{R}^+$ is a cost function representing the length of a road segment $a \in A$ that connects an intersection $s \in S$ to an intersection $s' \in S$, s_0 is a start intersection, and s_g is a goal intersection.

B. Obstacle Handling Problem

Next, we consider each exception handler. This includes automated obstacle handlers designed for each obstacle and an obstacle handler that is assumed to handle any obstacle using human assistance with a high penalty. Each decision-making model is based on an obstacle handling problem. That is, for each automated obstacle handler, $h \in H$, the **obstacle handling problem** can be expressed as the tuple $\langle S^h, A^h, T^h, C^h, s^h_0, s^h_g \rangle$, where $S^h = S^h_p \times S^h_l \times S^h_r \times S^h_b$ is a set of factored states such that S^h_p describes the position of the autonomous vehicle (obstructed/passing/passing with caution/collision/unobstructed), S^h_l and S^h_r describe whether or not the left lane and right lane are available (open/closed), and S^h_b describes whether or not the obstacle is blocking (blocking/not blocking), $A^h = \{\mathit{Stop}, \mathit{Edge}, \mathit{Go}, \mathit{Pass}, \mathit{PassWithCaution}\}$ is a set of actions representing the maneuvers of the autonomous vehicle, $T^h : S^h \times A^h \times S^h \to [0, 1]$ is a transition function multiplying the probabilities of scenarios including the probabilities $\Pr(s'_l | s_l)$ and $\Pr(s'_r | s_r)$ that the availability of the left lane and right lane changes and the probability $\Pr(s'_b | s_b)$ of whether or not the obstacle is blocking changes, $C^h : S^h \times A^h \times S^h \to \mathbb{R}^+$ is a cost function with unit cost for every state other than a goal state, s^h_0 is a start state with an obstructed position, and s^h_g is a goal state with an unobstructed position. Note that any state with a collision or infinite waiting is an absorbing dead end state with unit cost.

All obstacles handlers use an instance of the obstacle handling problem where the transition function corresponds to the expected behavior of each type of obstacle. This involves adjusting the probability $\Pr(s'_b | s_b)$ of whether or not the obstacle is blocking changes. For the static, dynamic, and volatile obstacle handlers, their policies indicate to pass the obstacle immediately (*Pass*), pass the obstacle cautiously (*PassWithCaution*), and wait for the obstacle to move (*Stop*) due to a low, medium, and high probability respectively.

C. Introspective Autonomous Vehicle

Finally, given the regular process and each exception handler, we consider the introspective autonomous system. The **introspective autonomous vehicle**, v, can be described as an introspective autonomous system with $\langle E^v, P^v, I^v, S^v, A^v, T^v, R^v, \Omega^v, O^v \rangle$, where

- $E^v = \{\ , e_1, e_2, e_3\}$ is a set of exceptions such that is no obstacle, and e_1, e_2, and e_3 is the presence of a static, dynamic, and erratic obstacle respectively,
- $P^v = \{\ , \ , h_1, h_2, h_3\}$ is a set of decision processes such that is the regular process, is the human assistance obstacle handler, and h_1, h_2, and h_3 is the static, dynamic, and erratic obstacle handler respectively,
- $I^v = \{\ , \ , i_b, i_m\}$ is a set of indicators such that is the success signal, is the failure signal, and i_b and i_m are signals that indicate whether or not an obstacle is blocking and moving respectively,
- $S^v = S^v \times E^v$ is a set of factored states: a standard state set S^v and the exception set E^v such that S^v is the wait time (none/short/medium/long),
- $A^v = A^v \cup P^v$ is a set of actions: a standard action set $A^v = \{\mathit{Edge}, \mathit{Wait}\}$ and the decision process set P^v,
- $T^v : S^v \times A^v \times S^v \to [0, 1]$ is a transition function,
- $R^v : S^v \times A^v \times S^v \to \mathbb{R}$ is a reward function,
- $\Omega^v = \ ^v \cup I^v$ is a set of observations: a standard observation set $\ ^v$ and the indicator set I^v, and
- $O^v : S^v \times A^v \times \Omega^v \to [0, 1]$ is an observation function.

Note that a monolithic POMDP that instead combines the navigation problem and the obstacle handling problems together is intractable. Given a navigation problem with $|S|$ states and n obstacle handling problems with 40 states, there are $|S| \cdot 40^n$ states. A simple example with 30 intersections and 3 types of obstacles has over a million states, which cannot even be solved with state-of-the-art solvers [11].

D. Analysis

Our goal is to show that the introspective autonomous vehicle can complete its route by handling all obstacles that can be detected and identified during navigation. We first show that every obstacle handler is strong below.

Proposition 1. *An obstacle handler is strong.*

Proof (Sketch) 1. *For an obstacle handler to be strong, it must be guaranteed to handle its corresponding obstacle for all states. In particular, if the obstacle handling problem is described as an SSP, it must ensure goal reachability by satisfying both conditions of an SSP. First, the problem has a proper policy: the* Stop *action can be performed to always avoid dead end states and eventually reach the goal state once the obstacle is no longer blocking. Second, all improper policies of the problem incur infinite cost for any state that may not reach the goal state. Therefore, by satisfying both condition of an SSP, the obstacle handler is strong.*

Finally, by using this proposition, it is easy to show that the introspective autonomous vehicle is robust as follows.

Theorem 1. *An introspective autonomous vehicle is robust.*

Proof (Sketch) 2. *Since all obstacle handlers are strong by Proposition 1, the introspective autonomous vehicle is robust.*

TABLE I

THE PERFORMANCE OF ALL AUTONOMOUS VEHICLES.

Obstacle Handlers	Incidents	Autonomy (%)	Transfers	Time (s)
None	12	—	—	—
λ	0	51.4	12	750.2
λ, h_1	0	60.3	9	700.0
λ, h_1, h_2	0	72.0	6	649.8
$\lambda, h_1, h_2, 3_3$	0	84.3	3	599.5

IV. DEMONSTRATION

In this section, we demonstrate that the introspective autonomous vehicle is effective in simulation and on a fully operational prototype. In particular, we compare different versions of the introspective autonomous vehicle to a regular autonomous vehicle without introspective autonomy. For each introspective autonomous vehicle, v_i, the available obstacle handlers H^{v_i} are the following: $H^{v_1} = \{\ \}$, $H^{v_2} = \{\ , h_1\}$, $H^{v_3} = \{\ , h_1, h_2\}$, and $H^{v_4} = \{\ , h_1, h_2, h_3\}$.

Each experiment represents an instance of the navigation problem with different obstacle handling problems: the introspective autonomous vehicle has to complete a route by passing obstacles. To do this, we run an *introspective autonomous vehicle process* that uses a belief to interleave decision processes. If the belief suggests normal operation, the *navigation process* is executed. Otherwise, if the belief indicates exceptional operation, an *obstacle handling process* is executed. The experiment concludes once the introspective autonomous vehicle process has been terminated.

All autonomous vehicles traverse a route with 3 instances of each type of obstacle and 3 unrecognized obstacles that can only be handled by the human assistance obstacle handler. Other routes can be built using the observation that the expected time needed to handle each obstacle is consistent: the static, dynamic, erratic, and unknown obstacles require 12.3, 15.1, 12.5, and 14.7 seconds respectively. Due to implementation constraints of the vehicle, transferring control safely to and from the driver requires 8.0 seconds.

Table I shows the performance of the regular autonomous vehicle and each version of the introspective autonomous vehicle. *Obstacle Handlers* lists the obstacle handlers available to the vehicle. *Incidents* includes the number of exceptions that prevent the vehicle from completing its route due to an exception that leads to a collision or infinite waiting. *Autonomy* shows the percentage of time that the vehicle is driven autonomously. *Transfers* includes the number of activations of the human assistance exception handler by the vehicle. *Time* presents the duration of the route in seconds.

We also demonstrate that the complete introspective autonomous vehicle is effective on a road in the real world with a static obstacle (a parked car), a dynamic obstacle (a slow-moving car), and an erratic obstacle (an unpredictable pedestrian) on the fully operational prototype in Figure 5.

Fig. 5. A fully operational introspective autonomous vehicle prototype.

V. DISCUSSION

In Table I, when no obstacle handlers are available, the introspective autonomous vehicle cannot complete its route due to 12 potential incidents. Once the human assistance obstacle handler becomes available, the vehicle completes its route without any potential incidents. Relying on the driver, however, leads to the worst level of autonomy (51.4%) and route time (750.2 s). As each obstacle handler becomes available, the vehicle continues to improve until reaching its best level of autonomy (84.3%) and route time (599.5 s) where all unrecognized obstacles must be handled by the driver. In short, with each exception handler, the vehicle improves its independence and efficiency during operation.

VI. CONCLUSION

We introduce a new approach to exception recovery in autonomous systems that uses belief space metareasoning. By reasoning over the assumptions of normal operation, an introspective autonomous system interleaves the regular process with different exception handlers to identify, detect, and handle unanticipated scenarios. Finally, we offer an introspective autonomous vehicle and demonstrate its effectiveness in simulation and on a fully operational prototype. Future work will extend introspective autonomy to multiple concurrent exceptions that interact during operation.

REFERENCES

[1] K. H. Wray, S. J. Witwicki, and S. Zilberstein, "Online decision-making for scalable autonomous systems," in *26th IJCAI*, 2017.

[2] M. A. Goodrich, B. S. Morse, D. Gerhardt, J. L. Cooper, M. Quigley, J. A. Adams, and C. Humphrey, "Supporting wilderness search and rescue using a camera-equipped mini UAV," *J. of Field Robotics*, 2008.

[3] B. C. Williams, M. D. Ingham, S. H. Chung, and P. H. Elliott, "Model-based programming of intelligent embedded systems and robotic space explorers," *Proceedings of the IEEE*, 2003.

[4] A. Ferrando, L. A. Dennis, D. Ancona, M. Fisher, and V. Mascardi, "Recognising assumption violations in autonomous systems verification," in *17th AAMAS*, 2018.

[5] J. Zhang and E. Bareinboim, "Characterizing the limits of autonomous systems," in *17th AAMAS*, 2018.

[6] E. A. Hansen and S. Zilberstein, "Monitoring and control of anytime algorithms: A dynamic programming approach," *J. of Artificial Intelligence Research*, 2001.

[7] V. Verma, G. Gordon, R. Simmons, and S. Thrun, "Particle filters for fault diagnosis," *IEEE Robotics and Automation Magazine*, 2004.

[8] P. Goel, G. Dedeoglu, S. I. Roumeliotis, and G. S. Sukhatme, "Fault detection and identification in a mobile robot using multiple model estimation and neural network," in *ICRA*, 2000.

[9] R. Alterovitz, S. Koenig, and M. Likhachev, "Robot planning in the real world," *AI Magazine*, 2016.

[10] L. P. Kaelbling, M. L. Littman, and A. R. Cassandra, "Planning and acting in partially observable stochastic domains," *J. of Artificial Intelligence Research*, 1998.

[11] J. Pineau, G. Gordon, and S. Thrun, "Point-based Value Iteration: An anytime algorithm for POMDPs," in *18th IJCAI*, 2003.

[12] A. G. Barto and S. Mahadevan, "Recent advances in hierarchical reinforcement learning," *Discrete Event Dynamic Systems*, 2003.

[13] J. Svegliato and S. Zilberstein, "Adaptive metareasoning for bounded rational agents," in *IJCAI Workshop on AEGAP*, 2018.

[14] J. Svegliato, K. H. Wray, and S. Zilberstein, "Meta-level control of anytime algorithms with online performance prediction," in *27th IJCAI*, 2018.