Scalable Model-Based Discrete Mode Estimation for a Lunar Rover Power System

Annabel R. Gomez[1], Zaki Hasnain [2], Michel D. Ingham[2], Seung H. Chung[2], and Brian C. Williams [1]

[1] *Massachusetts Institute of Technology, Cambridge, Massachusetts, 02139, USA*
annabelg@mit.edu

[2] *Jet Propulsion Laboratory, California Institute of Technology, 4800 Oak Grove Drive, Pasadena, CA, 91109, USA*
michel.d.ingham@jpl.nasa.gov, seung.h.chung@jpl.nasa.gov

ABSTRACT

Reliable autonomous operation of planetary rovers requires robust onboard monitoring and diagnosis of power systems under uncertain and dynamic conditions. This work presents a model-based mode estimation framework for a simplified rover power system that combines physics-based modeling with probabilistic reasoning to estimate system and environmental state, including anomalous behavior. The approach adapts the Miniature Mode Estimation (Mini-ME) architecture to perform real-time belief tracking over discrete component modes based on noisy measurement signals including voltage, current, temperature, and state of charge, and command inputs into the system. Mini-ME is a system health management tool that monitors components, diagnoses faults in real-time, and supports downstream system recovery. We simulate the physics of the simplified power system under varying loads using a lumped parameter equivalent circuit model that captures the electrical and thermal behavior of the battery. A compiled automaton is used to encode admissible probabilistic state transitions and observation likelihoods derived from sensor models. By compiling out the combinatorial search and integrating these representations within a Bayesian update process, the system efficiently detects, isolates, and diagnoses anomalies arising from faults or unexpected operating regimes. The simulation results demonstrate accurate recovery of hidden system modes during charge-discharge cycles, illustrating how model-based reasoning supports autonomous fault diagnosis and system health management in autonomous missions. This of the paper provides a thorough set of computational experiments to evaluate the performance of compiled mode estimation, versus uncompiled mode estimation, on space relevant benchmarks.

1. INTRODUCTION

1.1. Motivation

Autonomous systems for space exploration are rapidly evolving in complexity and capability, driven by the need for sustained operations in uncertain and communication-limited environments (Ingham et al., 2024). Future missions must feature autonomous operation for long durations while managing power, thermal, and mobility subsystems under uncertainty. They must infer internal health states, diagnose off-nominal behavior, and reconfigure themselves to maintain safe operation with minimal ground intervention, all while meeting severe computational, power, environmental and latency constraints.

Traditional rule-based systems for onboard system health management (SHM) lack the flexibility to capture novel or coupled failures, and purely data-driven learning methods lack the structure and verifiability required for safety-critical missions. Bridging these limitations requires an autonomy framework that is model-based, adaptive, and scalable.

An example of the limitations of traditional embedded software and rule-based systems is the Mars Polar Lander. When the lander was 40 meters above the surface of Mars and began polling its monitors to detect touchdown, the force of deploying the landing legs generated a transient noise spike on the Hall effect sensors. The software latched onto this and the lander read this stored spike as a real landing signal. This resulted in the lander prematurely cutting the engines, causing the spacecraft to plummet to the ground. Model-based programming approaches, on the other hand, are designed to prevent such "commonsense mistakes" by enabling a spacecraft to recognize that a touchdown signal is physically inconsistent with other sensor data, such as an altimeter showing it is still high above the terrain (Albee et al., 2000; Williams, Ingham, Chung, & H.Elliott, 2003).

Since then, a number of promising mode estimation tech-

niques have been demonstrated in flight. The next step, towards wide technology adoption is the systematic evaluation of these alternative mode estimation technologies, in terms of accuracy and computational demand, on space relevant benchmarks.

This work presents a model-based mode estimation framework, Mini-ME (Miniature Mode Estimation), applied to a simplified rover power system that combines physics-based modeling with probabilistic reasoning to estimate system and environmental state, including anomalous behavior.

NASA's proposed Endurance lunar rover mission provides a compelling context for model-based discrete mode estimation. The lunar South Pole-Aitken (SPA) is the largest and oldest impact basin on the Moon. Exploring and returning samples from it could answer many of our questions about the Moon and the formation of our solar system, including understanding the impact history in the Earth-Moon system and providing a window into the early thermochemical evolution of an archetypal rocky world (National Aeronautics and Space Administration, 2025). However, traversing across the SPA basin means that the rover would have to travel at least 2500 km, collecting various samples. Hence, the incorporation of advanced autonomy is necessary to increase the distance that can be covered during the mission.

Operating in the heavily shadowed regions of the SPA, under extreme environmental conditions, the Endurance rover must autonomously manage its power, thermal, and mobility systems over extended durations (Keane et al., 2023). In addition to collecting samples, the plan is to also have Endurance conduct continuous science observations as it traverses (Keane et al., 2023). The advancement of independent, self aware spacecraft provides a foundation for exploration of the solar system in the presence of highly-constrained communications, as would be experienced by the Endurance rover over the course of its long mission.

1.2. Mode Estimation

While operating in inhospitable environments like the Moon, the ability to infer system health, respond to faults, and adapt to evolving conditions becomes critical. The core capability that enables this is mode estimation: the inference of the internal health mode (nominal, degraded, or faulted) of a system from noisy and incomplete sensor data.

At the system level, a mode is a joint state configuration of all constituent components. For example, in the case of a rover power subsystem, the battery may be nominal or degraded, the heater may be on or off, and the drive motor may be active or idle. The combination of these discrete component conditions defines the overall system mode. Mode estimation therefore seeks to infer how these component-level modes evolve over time in response to commands and observed behavior. This is done by probabilistic reasoning over discrete operating regimes to infer symbolic health conditions that represent abstractions of system behavior. By encoding structural and behavioral knowledge of the system, model-based approaches provide interpretability and generalization, crucial for downstream recovery.

1.3. Relevant Work in Autonomy

This work is grounded in a lineage of model-based autonomy from MIT's Model-Based Embedded and Robotics Systems (MERS) Group (Williams et al., 2003; Ayton, Reeves, Timmons, Williams, & Ingham, 2020) and the NASA Ames Research Center (Aaseng, Do, Frank, Fry, & Sweet, 2023).

An earlier pivotal system is NASA's Autonomous Remote Agent (RA), which was used in the Deep Space 1 (DS1) mission (Bernard et al., 1999; Muscettola, Nayak, Pell, & Williams, 1998). This spacecraft autonomy system utilized Livingstone, a model-based mode estimation and reconfiguration engine (Williams & Nayak, 1996), to diagnose faults in the spacecraft's systems and autonomously take appropriate corrective actions. RA and Livingstone reduced reliance on ground control, allowing the spacecraft to recover from malfunctions independently. However, its algorithm implementation was resource-intensive and required considerable computational overhead that grew with system complexity (Bernard et al., 1999).

More recent work has explored model-based fault diagnosis within autonomous robotics systems, where fault estimation is integrated with higher-level autonomy and decision making. For example, Hasan et al. developed diagnosis algorithms capable of detecting, localizing, and estimating actuator faults in robotic systems to be used for monitoring and control of operational decisions (Hasan, Tahavori, & Midtiby, 2023). Such examples show that, unlike model-free data-driven approaches that require large datasets for proper operation, model-based approaches that use physical understanding of the systems dynamics perform more efficient diagnosis (Hasan et al., 2023). This reflects an efficient method in autonomous systems that reduces data dependency by coupling model-based approaches with system-level health management and reducing data dependency.

These approaches highlight the growing need for autonomous systems to reason over health state information and use diagnostic outputs to guide subsequent decisions in a computationally efficient manner (Hasan et al., 2023; Chung, Van Eepoel, & Williams, 2001). Mini-ME was developed to address this. It's approach aims to move the expensive NP-complete satisfiability portion of model-based diagnosis offline and compile the model to retain only information needed for diagnosis. Its modular design efficiently performs the most computationally intensive parts of mode estimation and fault diagnosis offline, making it more reactive and robust for space mis-

sions (Chung et al., 2001). However, after in-flight demonstrations, to carry mode estimation methods further towards future implementation, a systematic evaluation is required. This paper's systematic evaluation demonstrates compiled mode estimation as a promising option for computationally-constrained spacecraft systems.

The following sections of this paper describe the approach taken in this research to evolve Mini-ME and apply it to the Endurance lunar rover power subsystem, the results from testing Mini-ME in various scenarios and benchmarking it against the earlier (Livingstone-like) implementation, and the conclusions and future work.

2. APPROACH

2.1. Mini-ME

Mini-ME addresses the challenge of computational complexity in the estimation and diagnosis problem by compiling the discrete model offline into an efficient finite-state representation, enabling predictable and efficient runtime inference without combinatorial search (Chung et al., 2001). Rather than constructing diagnoses online through search, Mini-ME performs belief updates over precomputed admissible joint states, significantly reducing onboard computational burdens.

The approach discussed in this paper modernizes compiled reasoning and rebuilds Mini-ME into a contemporary software environment with integrated probabilistic inference techniques to better handle uncertainty. Here, Mini-ME is implemented on a lunar rover power system and extended to incorporate coupled electrical-thermal dynamics via a simplified circuit battery model. The goal is to conduct belief tracking and probabilistic reasoning over the discrete component modes given noisy measurement signals, including voltage, current, and state of charge.

Mini-ME, as detailed in Figure 1, is made up of an offline compiler and an online runtime engine.

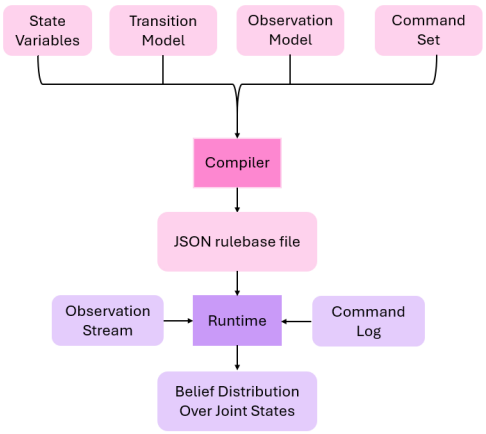


Figure 1. Mini-ME Structure

The inputs to the compiler include state variables that map component names to their possible modes, a set of all possible commands that affect mode transitions, a transition model that defines the probability of transitioning between system states given a command, and an observation model that defines the likelihood of receiving particular discrete measurement values given the current system mode.

Each component maintains a discrete set of operating states that abstract underlying system behavior. In this work, we focus on variables that capture how energy flows through the system, specifically the rate of energy input and the level of energy consumption. For example, *Charge Rate* $\in \{nominal, not\_charging, abnormally\_low, higher\_than\_nominal\}$, representing the level of resource input relative to expected operation. Similarly, *Activity Rate* $\in \{background, lower\_than\_expected\_load, nominal, higher\_than\_expected\_load\}$ representing the level of system load relative to baseline conditions. At a given time, the system mode is defined as the joint assignment of all the component states.

The transition model defines the probability of transitioning between system modes, given a command, while the observation model defines the likelihood of observing a particular symbolic measurement given the current system mode. The Mini-ME model assumes first-order Markov dynamics, where the next system mode depends only on the previous system mode and the current system commands.

An example of some transitions are shown in Figure 2, specifying the four possible *Activity Rate* states and the possible transitions between them. Nominal transitions (solid lines) reflect expected changes in load, such as increasing from background to nominal when an activity, e.g. driving, is commanded, or returning to background when activity stops. Anomalous transitions (dotted lines) capture unexpected changes in load that are not explained by commands. Each transition is associated with a probability that reflects the likelihood of that change in state occurring.

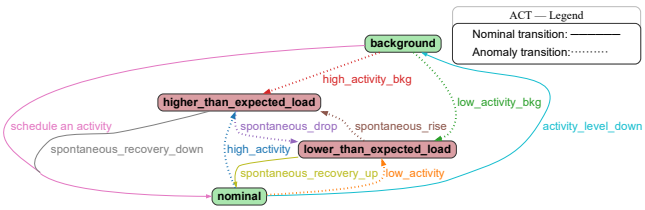


Figure 2. Activity Rate Transition Automata

During compilation, all admissible system modes are enumerated and indexed, and the full transition and observation probability structures are precomputed and stored in a rulebase file. Logical consistency constraints are applied to eliminate physically impossible combinations of component states.

This process transforms a structured component-level description into an executable finite-state stochastic model.

At runtime, Mini-ME performs recursive Bayesian belief updates of a distribution over system modes, using these precompiled rules. At each timestep, the prior belief distribution is first propagated forward using the transition model (prediction step), and then corrected using the observation likelihood (measurement update), yielding a posterior distribution over system modes, as shown in Equation 1.

$$\mathrm{B}(S') = \eta \underbrace{P(o \mid S')}_{\text{observation}} \sum_{S} \underbrace{P(S' \mid S, u)}_{\text{transition}} \underbrace{\mathrm{B}(S)}_{prior} \tag{1}$$

where S and S' denote system modes at consecutive timesteps, u is the issued command, o is the symbolic observation, and η is a normalization constant. The summation is performed over the finite set of admissible system modes produced during compilation.

By compiling all admissible joint states and their associated probabilities offline, Mini-ME avoids combinatorial search during execution, enabling efficient, real-time onboard inference.

2.2. Simplified Rover Power Subsystem Simulator

To test Mini-ME without a hardware testbed for the rover power subsystem and provide physically realistic data for Mini-ME to process, a first-order equivalent circuit model was created for a Li-ion cell, as shown in Figure 3. This battery is modeled using a steady-state resistive equivalent circuit model consisting of an ideal voltage source in series with an internal resistance (Jagdale, 2023; Murnane & Ghazel, 2023). Dynamic electrochemical effects, including capacitive transients and diffusion processes, are neglected. The system is evaluated at discrete operating points, assuming steady-state conditions at each timestep. This abstraction is sufficient to capture power balance behavior and fault-induced current deviations relevant to discrete mode estimation.

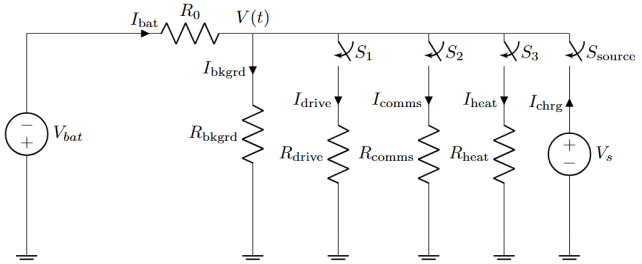


Figure 3. Li-ion Battery Simulator Circuit Model

The model includes multiple electrical loads in parallel, corresponding to distinct system activities that the proposed Endurance Rover may need to perform such as driving, communications, and heating. These are modeled as switchable, resistive branches that draw current when active. Additionally, as is expected in a rover, there is a resistor representing the constant background activity needed to maintain general system function such as keeping components at their required temperature and powering an always-on onboard computer. A switch was also added to represent a possible incoming power source to the battery.

The simulator is executed using realistic resistances, voltages, and current values for each activity, along with a schedule of charging and activity durations to generate representative current profiles. This information was gleaned from power subject matter experts at JPL. From these simulations, expected current ranges are used to define thresholds that map continuous telemetry into qualitative symbolic observations. The observation thresholds and transition and observation probabilities were manually specified using current profiles generated from nominal and off-nominal simulation schedules. The probabilistic parameters were therefore calibrated offline from representative simulated operating regimes rather than estimated online during inference.

In particular, current measurements are converted into variables such as *Charge Rate* and *Activity Rate*, each representing nominal and off-nominal operating regimes. These symbolic observations are provided as inputs to Mini-ME, along with a compiled automaton that encodes admissible state transitions and observation likelihoods derived from the system model. Through a Bayesian update process, Mini-ME maintains a belief over system modes to detect, isolate, and explain anomalies arising from faults or unexpected operating conditions.

3. MINI-ME RESULTS: NOISELESS TELEMETRY

3.1. Case 1: Nominal Run

To evaluate the behavior of the compiled Mini-ME estimator under nominal operating conditions, we first consider a representative simulation episode. The resulting activity schedule, battery response, and inferred system modes are shown in Figure 4.

In this plot, the Mini-ME estimation results are shown on the third subplot. The y-axis labels the joint diagnosis of the *Activity Rate* and *Charge Rate* hidden variables where the top line of the label, "C:", refers to the *Charge Rate*, and the bottom line of the label, "A:", refers to the *Activity Rate*.

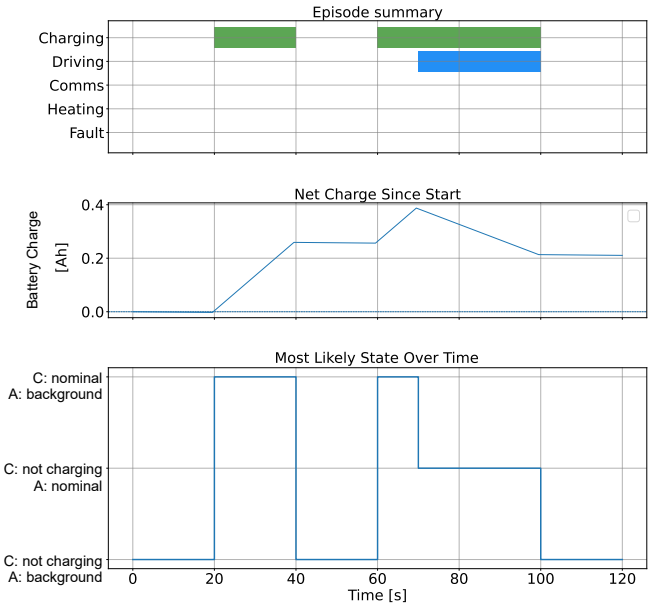


Figure 4. Integrating Mini-ME with a Driving Activity

In this nominal Mini-ME test run with a driving activity, the episode begins in a background, non-charging state. At 20 seconds, charging is initiated, resulting in an increase in net charge. During this period, Mini-ME correctly infers the joint mode transition to *Charge Rate = nominal, Activity Rate = background*.

At 70 seconds, a drive begins, as indicated by the activation of a driving load. Of the possible activities, driving results in the largest power load. The increase in current draw causes a decrease in net charge, even though the voltage source switch is on (i.e., the charging activity is active). Mini-ME then transitions its state estimate to reflect the higher activity level to *Charge Rate = not charging, Activity Rate = nominal.* Once the driving activity ends, the system returns to a background load state and the estimator correspondingly updates the joint mode.

Throughout the episode, Mini-ME tracks the discrete component modes consistently with the commanded activities and the resulting electrical behavior, demonstrating correct inference under nominal operating conditions.

3.2. Case 2: Injected Fault

Next, injected faults were incorporated into the simulator's schedule to test how well Mini-ME could handle anomalous behavior. This is demonstrated in Figure 5. In this particular scenario, an individual charging activity and simultaneous charging and driving activities were scheduled. Up until the fault, at around 70 seconds, the system operates nominally, as it did in Figure 4. However, the injected fault begins at the start of the driving activity, causing an extra, unexpected load of 1.5 Ω to be added to the existing driving load, resulting in a higher than nominal activity load.

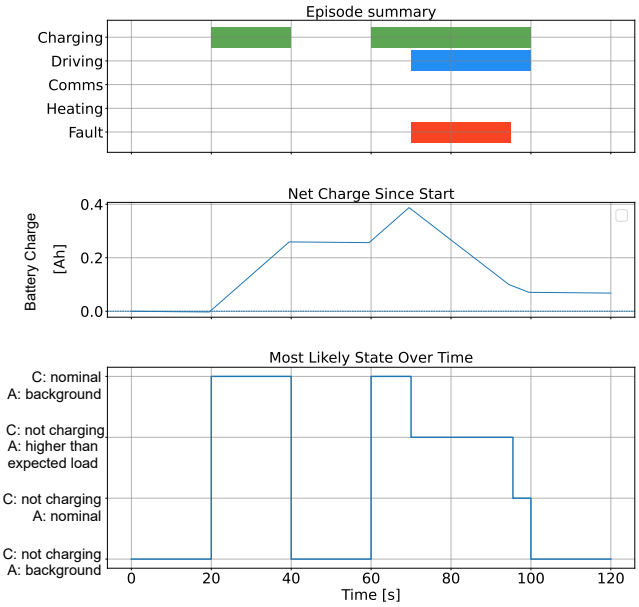


Figure 5. Integrating Mini-ME with a Driving Activity and an Injected Fault

During the fault period we see the net charge decrease rapidly and Mini-ME quickly adjusts its estimation to diagnose the fault: *Charge Rate = not charging, Activity Rate = higher than expected load.* When the fault period ends, around 95 seconds, Mini-ME transitions its estimate to reflect the nominal driving activity load and respective background load. It is also evident that the net charge between 95 seconds and 100 seconds has a less aggressive negative slope since there is no longer an extraneous load.

3.3. Case 3: Current and Observation Noise

There are many observed currents within the simulated circuit. To evaluate robustness under realistic sensing conditions, zero-mean Gaussian noise was injected into the continuous charging and activity load current signals, prior to symbolic abstraction. This approach preserves the physical consistency of the simulation while introducing uncertainty at the sensing layer. The added noise did not affect the estimated diagnosis. This is due to Mini-ME's probabilistic approach and the way it uses thresholds to take the quantitative sensor observation and map it to a qualitative label.

To evaluate how Mini-ME responds to perception errors, a symbol-misclassification experiment was conducted in which the physics remained unchanged but the perception layer occasionally produced incorrect observations. More specifically, at each time step, each *Charge Rate* and *Activity Rate* observation was independently subject to a 10% probability of

being replaced with an alternative qualitative label, i.e., an observation ”flip” to an alternative qualitative level. More specifically, !0% of the time, a nominal measurement value would be instead reported as an adjacent discretized value with equal likelihood. For instance, a ”nominal” activity observation could be flipped and reported as ”lower than expected” or ”higher than expected”. This setup simulates realistic perception uncertainty, where symbolic observations may occasional be corrupted by noisy sensor measurements. A 10% flip probability was chosen to introduce noticeable but non-dominating uncertainty, providing a moderate noise regime that evaluates the robustness of Mini-ME to intermittent symbolic misclassifications without overwhelming the underlying system dynamics. The results are presented in Figure 6.

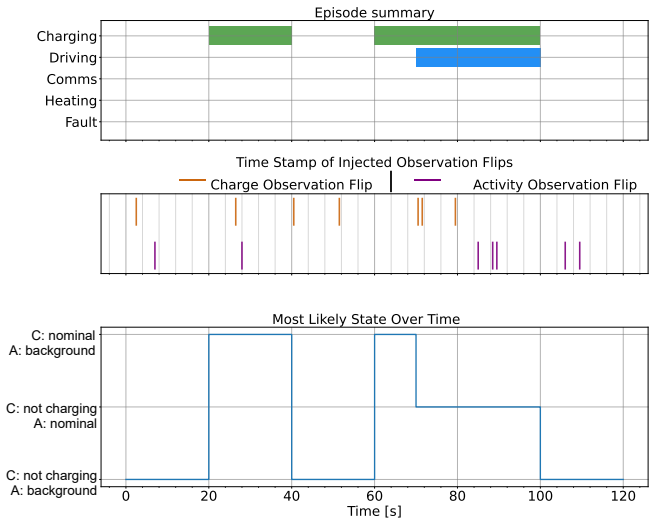


Figure 6. Integrating Mini-ME with Sensor Noise and Observation Flips with a Probabilistic Observation Model

In Figure 6, the middle subplot indicates the time steps at which observation flips occur for the activity and charge rates. During the driving segment beginning at 70 second mark, multiple *Activity Rate* observation flips are visible in the middle subplot, corresponding to brief transitions in the estimated state to different joint modes (e.g., *Activity Rate* flipping from *nominal* to *higher than expected load*). However, these deviations are transient and do not lead to sustained divergence. The most likely state estimate remains stable throughout the episode, reflecting consistency between observations and the expected operating regime. This behavior arises from Mini-ME's Bayesian update mechanism, which incorporates prior belief and transition dynamics, in addition to current observations. As a result, isolated misclassifications are insufficient to overcome strong self-transition probabilities, and only repeated or sustained observation errors produce enough ”inertia” to change the estimated mode.

These results are expected given Mini-ME's probabilistic observation model and instantaneous qualitative filtering updates at each time step. One bad symbol is not enough evidence; instead, Mini-ME accumulates evidence over time and the state changes only after enough contradictory observations are received. In other words, we are trading some degree of timing responsiveness for robustness to noise and spurious observations. This is tunable based on the characteristics of the probabilistic observation model.

To prove how crucial the probabilistic observation model is for Mini-ME, Figure 7 shows how Mini-ME responds to noise with a deterministic binary observation model. Here, Mini-ME was compiled and run using a more rigid observation model, resulting in each sensor input having a large impact on the estimated mode of the system, as demonstrated by the multiple errors in the most likely state diagnosis as a result of each injected flip in the observation stream. When Mini-ME is constrained by a deterministic observation model, this results in a belief state that is consistent with the spurious observation.

Comparing the results in Figures 6 and 7 highlights the role of the observation and transition models in shaping inference behavior.

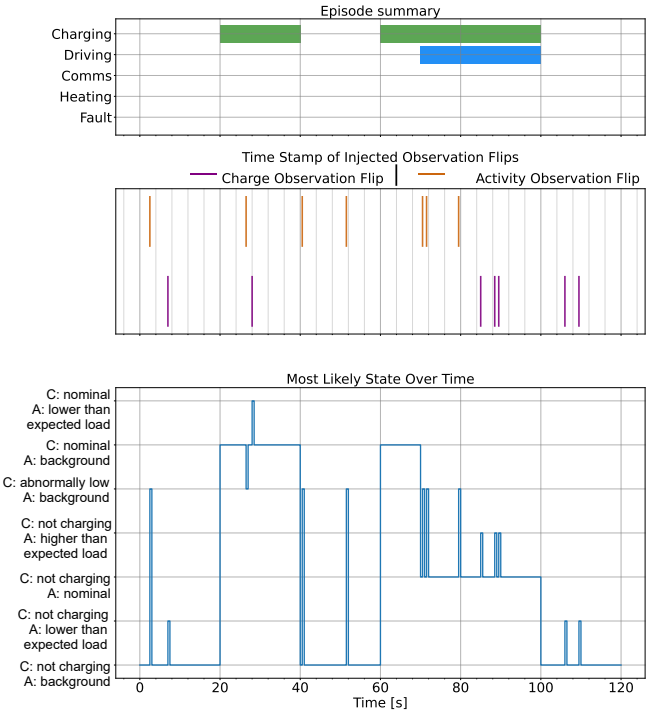


Figure 7. Integrating Mini-ME with Sensor Noise and Observation Flips without a Probabilistic Observation Model

When observation likelihoods are softer, the transition model is used to prevent isolated noisy observations from dominating belief updates. When the observations are treated more

deterministically, the system becomes correspondingly more reactive but also more sensitive to noise and spurious observations.

This comparison illustrates an important property of Mini-ME: its inference behavior is governed by the relative weighting of the observation and transition models. These models can be specified from domain knowledge or learned from data, enabling the framework to generalize across applications, while balancing robustness and responsiveness according to the characteristics of the model input pipeline.

3.4. Case 4: Faults and Noise

Combining the injected fault scenario from Figure 5 and the injected observation flips from Figure 6, Figure 8 demonstrates Mini-ME's ability to return stable estimates despite sensor noise and observation flips, when utilizing a probabilistic observation model.

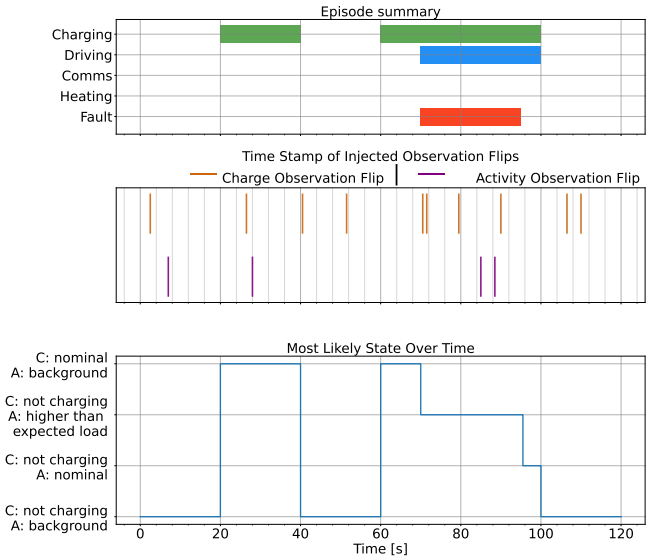


Figure 8. Integrating Mini-ME with Sensor Noise, Observation Flips and a Fault with a Probabilistic Observation Model

In contrast, Figure 9 shows the effect of utilizing a deterministic observation model: similar sensitivity to noise and spurious observations as was seen in Figure 7.

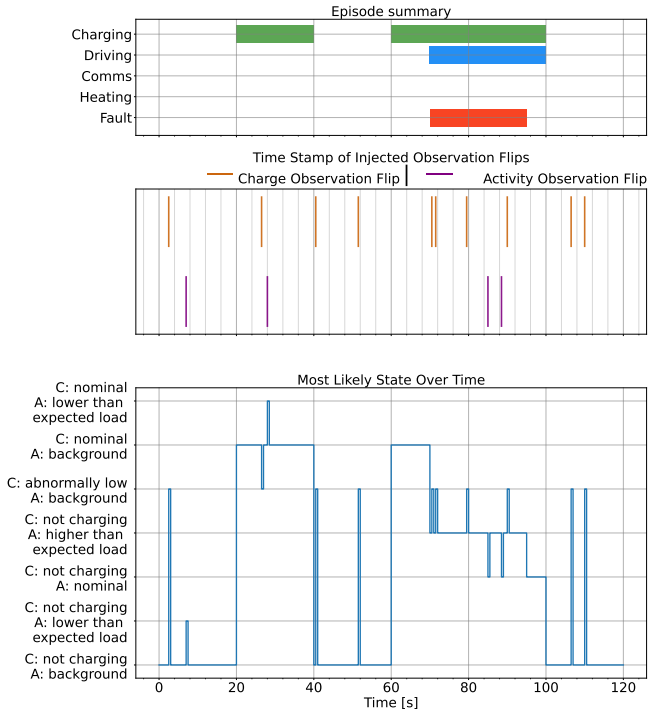


Figure 9. Integrating Mini-ME with Sensor Noise, Observation Flips and a Fault without a Probabilistic Observation Model

4. BENCHMARKING AGAINST CONFLICT-DIRECTED BEST FIRST SEARCH

To evaluate Mini-ME's real-time efficiency, we implemented a simplified, Livingstone-inspired conflict directed best-first search baseline for mode estimation. Livingstone, developed by NASA and flown on DS1, was a model-based autonomy system designed to enable deep-space missions to perform real-time onboard fault detection, diagnosis, and recovery by reasoning over system behavior (Williams & Nayak, 1996; Hayden, Sweet, & Shulman, 2005).

4.1. Comparison 1: Livingstone-Inspired Baseline with a Z3 Solver

Livingstone utilized conflict-directed best first search (CBFS) to solve the optimization problem of inferring a system's current state and determining the optimal control actions needed to meet configuration goals (Williams, Ingham, Chung, Elliott, & Hofbaur, 2004). We re-implemented CBFS using modern tools and assessed its performance in the context of our rover power subsystem simulation example. This implementation is referred to as CBFS-Z3. This baseline performs discrete mode estimation at each time step by searching over candidate assignments of component modes that are consistent with the previous state, the issued commands, and the current observations. Again, in the case of the simplified battery simulator, the two component modes include *Charge*

Rate and *Activity Rate*.

Transition probabilities from the Mini-ME input model were converted into additive costs using negative log likelihood. These costs were accumulated across temporal transitions and used to prioritize nodes in a best-first search implemented with a priority queue. Because no separate heuristic term was introduced, the search operates as a uniform-cost variant of A* (Williams & Ragno, 2003).

Constraint consistency was enforced using the Z3 SMT solver (Moura & Bjørner, 2008). Each candidate partial assignment was checked for satisfiability against logical constraints encoding command semantics, mutual exclusivity of modes, temporal persistence assumptions, and observation compatibility. However, unlike Mini-ME, which compiles the probabilistic and logical structure offline and performs lightweight belief updates online, this best-first search approach invokes a constraint solver repeatedly at runtime. Spacecraft models are large and while using conflicts helps to focus the search, the repeated solver calls are still computationally expensive for use in real-time, particularly in a computationally-constrained onboard computing environment.

It should be noted that these Livingstone-inspired CBFS baselines were implemented with a non-probabilistic observation model. This is because CBFS works best when observations act as hard constraints, helping to prune the diagnostic space. In fact, Livingstone was originally designed under the assumption that observations are logical constraints, not soft evidence (Williams & Nayak, 1996). When the observation model is deterministic, only a few states are consistent, focusing the search to return the single best explanation.

Figure 10 shows the same injected fault schedule run for Mini-ME as in Figure 5.

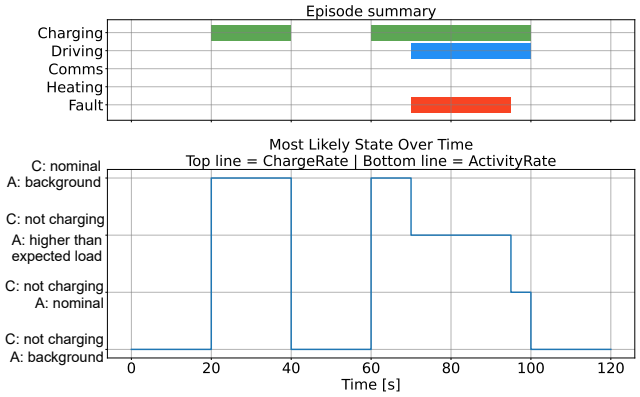


Figure 10. Integrating Mini-ME without a Probabilistic Observation Model with a Driving Activity and an Injected Fault

To make a more fair comparison, here, Mini-ME was run without a probabilistic observation model since the CBFS baseline was designed to use a deterministic observation model. Regardless, this did not change Mini-ME's accuracy for this scenario. These results demonstrate Mini-ME's ability to quickly diagnose this fault. However, this also means that when noise perturbs the observed symbols, small changes in the observational input can change which explanation has the lowest cost. This leads to abrupt switching between candidate mode estimates, even when the underlying system state has not changed.

Figure 11 shows the same injected fault schedule run for CBFS-Z3. During nominal operation, the most probable hypothesis is quickly verified as consistent, so only a small number of node expansions and solver calls are required. At the start and end of the injected fault window, the observations deviate from the expected nominal behavior and thus the search must explore additional hypotheses, leading to increased node expansions, solver calls, and conflict counts. The spikes in these metrics directly reflect the additional reasoning required to reconcile unexpected observations with model constraints.

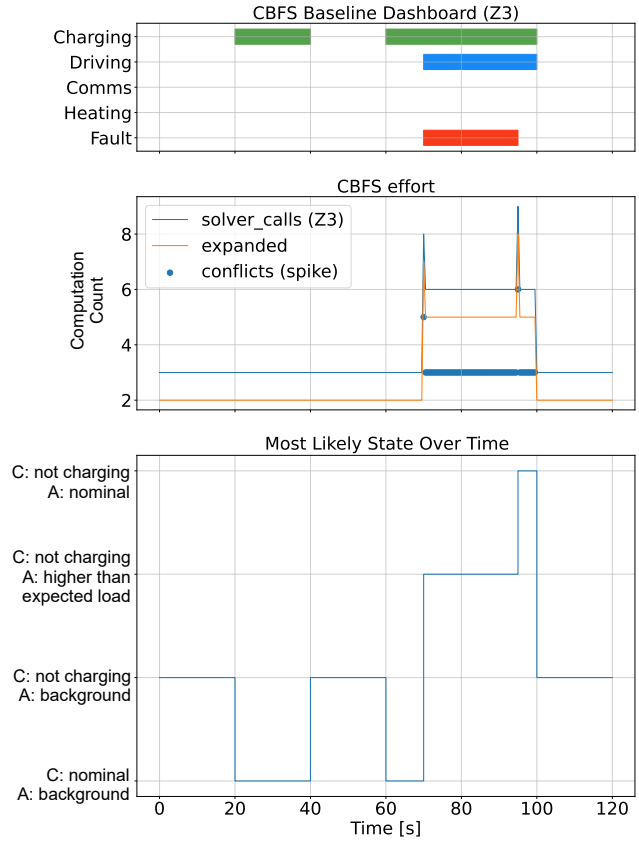


Figure 11. Integrating CBFS-Z3 with a Driving Activity and an Injected Fault

The solver call count is consistently one greater than the number of expanded nodes because each node expansion triggers a satisfiability check, and an additional solver call is used to validate the final candidate assignment before termination.

4.2. Comparison 2: Livingstone-Inspired Baseline with Unit Propagation

Next, to compare more closely to what was flown on DS1, a Livingstone-inspired CBFS was implemented utilizing unit propagation instead of calling a solver (Nayak & Williams, 1997). This implementation is referred to as CBFS-UP. Unit propagation is a linear-time logical inference procedure that repeatedly assigns values to variables that appear in clauses with only one remaining unassigned literal until either a contradiction is detected or no further forced assignments exist (Qu, 2006). Similar to the DS1 implementation of Livingstone, a Logical Truth Maintenance Systems (LTMS) unit propagation was used (Nayak & Williams, 1997). During runtime, this is a more efficient process since, unlike CBFS-Z3, CBFS-UP does not search for satisfiable solutions. It is a propagation-based consistency check that the CBFS search calls repeatedly to detect conflicts and decide whether to prune or accept a node. While unit propagation accelerates local consistency checking, the CBFS-UP algorithm itself still relies on search to explore candidate mode assignments. As a result, runtime performance still depends on the size of the hypothesis space and the number of conflicts encountered during search. The results can be shown in Figure 12.

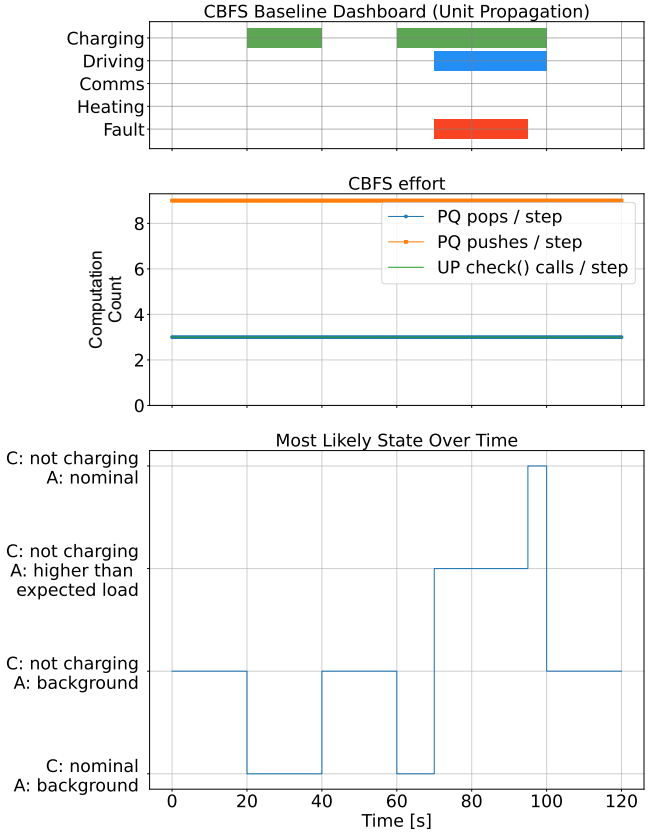


Figure 12. Integrating CBFS-UP with a Driving Activity and an Injected Fault

In Figure 12, priority queue (PQ) pops represent the number of hypotheses explored from the priority queue, PQ pushes represent how many new hypotheses were added to the queue, and finally UP check calls represent how many times the unit propagation consistency checker was called (overlaying the PQ pops line in this figure).

To quickly compare with Figure 11, the fewer calls does not mean that CBFS-UP is more efficient than CBFS-Z3, as these are two different satisfiability approaches. Additionally, the absence of conflicts in the CBFS-UP baseline is not due to the observation model but rather the limited reasoning power of unit propagation. It can only detect contradictions when something becomes forced by unit clauses.

While both the Z3-based solver and the unit propagation approach compute a maximum a posteriori explanation at each timestep, the choice of solver primarily affects search efficiency rather than robustness to observation noise, especially with a battery model where most mode combinations are logically valid and the differences are driven mostly by observation likelihood.

4.3. CPU and Memory Characterization

The main difference between the two Livingstone-inspired approaches is how they check consistency. CBFS-Z3 constructs full logical constraints, checks satisfiability globally, and finds conflicts precisely to perform complete logical reasoning. CBFS-UP instead applies logical implications, propagates forced assignments, and detects some contradictions, but it is incomplete compared to a full SAT solver.

The fundamental difference between Livingstone and Mini-ME lies in how the system model is used during inference. Livingstone performs diagnosis through online search over candidate system states, using CBFS to generate and prune hypotheses consistent with observations. In contrast, Mini-ME compiles the system model offline into a finite-state representation of admissible joint modes and transitions. Runtime inference therefore reduces to a belief update over this compiled state space rather than a search procedure, and runtime complexity depends primarily on the size of the compiled state representation.

Table 1 shows the memory and runtime cost of the three online inference implementations on the injected fault schedule shown in Figures 10, 11, and 12, all of which do not use a probabilistic observation model. It also shows the memory usage of Mini-ME, on the same scheduled scenario, with a probabilistic observation model, showing the negligible difference the critical addition makes to memory usage and computation time.

Mini-ME (both with and without a probabilistic model) exhibits near-constant memory usage during inference, since belief updates are performed over a precompiled set of ad-

missible joint states and do not require maintaining a growing search frontier. In contrast, both CBFS baselines allocate additional memory during inference to store partial assignments, the queue of priorities, and accumulated conflicts. The CBFS-UP implementation is lighter-weight, memory-wise, than the CBFS-Z3 implementation. However, because unit propagation performs weaker propagation than a full solver, it tends to prune fewer inconsistent partial assignments early, placing it between Mini-ME's compiled filtering and Z3-based constraint reasoning in the compute-pruning tradeoff.

Small run-to-run variation in all runtimes and (Resident Set Size) RSS was observed, even under identical deterministic schedules. This variation is attributed primarily to Python interpreter overhead, operating system scheduling, memory allocation behavior, etc. rather than changes in the underlying symbolic inference workload.

Table 1. Online Inference Memory Usage and Computation Cost

Method	RSS before (MB)	Peak RSS (MB)	ΔPeak (MB)	Inference Time (s)
Mini-ME	88.8	89.05	0.24	0.0085
CBFS-UP	90.4	92.2	1.8	1.8848
CBFS-Z3	114.4	116.2	1.8	6.6815
Mini-ME (Prob. Obs.)	89.2	89.4	0.2	0.0537

5. CONCLUSION

These results from running Mini-ME against a simplified rover power subsystem simulation demonstrate accurate recovery of hidden system modes during charge-discharge cycles and thermal transients, maintaining consistent fault isolation under sensor noise and varying load conditions. From a responsiveness perspective, Mini-ME achieves about 2–3 orders of magnitude faster runtime performance compared to our implementations of CBFS-based approaches. Mini-ME also requires about 7 times less inference effort per step than the CBFS methods and has an overall lower memory usage to both methods, in particular, about 30% lower memory than CBFS-Z3. By maintaining a continuous probabilistic belief over symbolic system activity modes and employing a probabilistic observation model, the Mini-ME framework enables robust, autonomous system health management under noise and uncertainty.

Beyond improved runtime efficiency, the compiled inference approach used by Mini-ME also enables scalability for complex systems. Admissible system modes and transitions are enumerated offline, reducing runtime inference to lightweight belief updates over the compiled state representation rather than search through combinatorial hypothesis space. As system complexity increases, search-based diagnosis methods, like CBFS, must explore increasingly large candidate sets, whereas Mini-ME maintains predictable inference costs once compilation is complete. This property is particularly valuable for embedded autonomy systems and spacecraft software, where bounded computational cost is critical.

5.1. Future Work

Future work will include integration of a more detailed rover behavior model and more advanced rover power system simulator that could incorporate a solar array as a power source, including, for example, the variations of exposure to the sun as it moves in the sky relative to the rover. Future work will also explore scaling and distribution of the Mini-ME architecture, to advance fault-tolerant autonomy for future missions. This includes extending this compiled mode estimation framework to multi-component interactions within the proposed Endurance rover (e.g., the thermal and mobility subsystems). The application used in this paper was relatively simple for the purpose of illustrating the approach but more complex models, including couplings between multiple systems could be modeled. In addition, future efforts will investigate combining the model-based Mini-ME framework with data-driven machine learning approaches, enabling learned probabilistic observation and transition models to compliment the underlying system model. Techniques such as Expectation Maximization could be used to learn these distributions from data, improving both robustness and adaptability in uncertain or evolving environments.

ACKNOWLEDGMENT

This research was partially carried out at the Jet Propulsion Laboratory, California Institute of Technology, under a contract with the National Aeronautics and Space Administration. This work was also partially funded by the Boeing Company.

REFERENCES

Aaseng, G., Do, M., Frank, J., Fry, C., & Sweet, A. (2023). *Integrating planning, diagnosis, and execution for vehicle systems management* (Tech. Rep.). Intelligent Systems Division, NASA Ames Research Center. Retrieved from https://ntrs.nasa.gov/citations/20230004265

Albee, A., Battel, S., Brace, R., Burdick, G., Casani, J., Lavell, J., ... Dipprey, D. (2000, March). *Report on the loss of the mars polar lander and deep space 2 missions* (Tech. Rep. No. JPL D-18709). NASA Jet Propulsion Laboratory. Retrieved 2026-05-23, from https://ntrs.nasa.gov/api/citations/20000061966/downloads/20000061966.pdf (NASA Technical Report)

Ayton, B., Reeves, M., Timmons, E., Williams, B. C., & In-

gham, M. D. (2020). Toward information-driven and risk-bounded autonomy for adaptive science and exploration. In *Aiaa ascend conference.* Cambridge, MA.

Bernard, D., Dorais, G., Gamble, E., Kanefsky, B., Kurien, J., Man, G. K., ... Tung, Y.-W. (1999). Spacecraft autonomy flight experience: The ds1 remote agent experiment. In *Proceedings of the aiaa conference.* Pasadena, CA.

Chung, S. H., Van Eepoel, J. M., & Williams, B. C. (2001). Improving model-based mode estimation through offline compilation. In *Proceedings of the i-sairas 2001 conference* (p. –). Montreal, Canada. Retrieved from https://groups.csail.mit.edu/mers/old-site/papers/isairas01_minime.pdf

Hasan, A., Tahavori, M., & Midtiby, H. S. (2023). Model-based fault diagnosis algorithms for robotic systems. *IEEE Access*, *11*, 2250–2258. doi: 10.1109/ACCESS.2022.3233672

Hayden, S. C., Sweet, A. J., & Shulman, S. (2005). *Lessons learned in the livingstone 2 on earth observing one flight experiment* (Tech. Rep.). NASA Ames Research Center. Retrieved from https://ntrs.nasa.gov/api/citations/20060006365/downloads/20060006365.pdf

Ingham, M., Hasnain, Z., Amini, R., Ardito, S., Bandyopadhyay, S., Bocchino, R., ... Rouquette, N. (2024). Onboard planning and execution of mobility and telecommunications for the endurance lunar rover. In *Aiaa ascend conference.* Pasadena, CA.

Jagdale, S. (2023). *Modeling li-ion batteries with equivalent circuit technology.* Retrieved 2025-11-19, from https://www.powerelectronicsnews.com/modeling-li-ion-batteries-with-equivalent-circuit-technology/

Keane, J. T., et al. (2023). *Endurance: Lunar south pole–aitken basin traverse and sample return rover* (Tech. Rep.). NASA. Retrieved from https://science.nasa.gov/wp-content/uploads/2023/11/endurance-spa-traverse-and-sample-return.pdf

Moura, L. D., & Bjørner, N. (2008). Z3: an efficient SMT solver. In *Tools and algorithms for the construction and analysis of systems, TACAS 2008, 14th international conference, held as part of ETAPS 2008, budapest, hungary, march 29-april 6, 2008, proceedings* (Vol. 4963, pp. 337–340). Springer. Retrieved from https://doi.org doi: 10.1007/978-3-540-78800-3_24

Murnane, M., & Ghazel, A. (2023, Mar). A closer look at state of charge (soc) and state of health (soh) estimation techniques for batteries. *Analog Devices Technical Articles*. https://www.analog.com/en/resources/technical-articles/a-closer-look-at-state-of-charge-and-state-health-estimation-tech.html.

Muscettola, N., Nayak, P. P., Pell, B., & Williams, B. C. (1998). Remote agent: To boldly go where no ai system has gone before. *Artificial Intelligence*, *103*(1-2), 5–47. Retrieved from https://www.sciencedirect.com/science/article/pii/S000437029800122X

National Aeronautics and Space Administration. (2025). *What is the south pole–aitken basin?* https://science.nasa.gov/resource/south-pole-aitken-basin/.

Nayak, P. P., & Williams, B. C. (1997). Fast context switching in real-time propositional reasoning. In *Proceedings of the national conference on artificialintelligence* (pp. 50–56).

Qu, S. (2006). *Fast incremental unit propagation by unifying watched-literals andlocal repair* (Unpublished master's thesis). Massachusetts Institute of Technology.

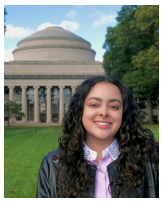
Williams, B. C., Ingham, M. D., Chung, S., Elliott, P. H., & Hofbaur, M. (2004). Model-based programming of fault-aware systems. *AI Magazine*, *24*(4), 61–75.

Williams, B. C., Ingham, M. D., Chung, S., & H.Elliott, P. (2003, January). Model-based programming of intelligent embedded systems and roboticspace explorers. *Proceedings of the IEEE*, *91*(1), 212–237.

Williams, B. C., & Nayak, P. P. (1996). A model-based approach to reactive self-configuring systems. In *Proceedings of the thirteenth national conference on artificial intelligence (aaai-96)* (pp. 971–978). Portland, Oregon, USA: AAAI Press / The MIT Press. Retrieved from https://www.aaai.org/Papers/AAAI/1996/AAAI96-144.pdf

Williams, B. C., & Ragno, R. (2003, June). Conflict-directed a* and its role in model-based embedded systems. *Special Issue on Theory and Applications of SatisfiabilityTesting, Journal of Discrete Applied Math*, *155*(12), 1562–1595.

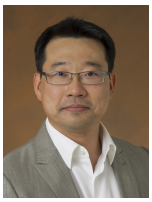
BIOGRAPHIES



Annabel R. Gomez is a Graduate Research Assistant at the MIT Computer Science and Artificial Intelligence Laboratory (CSAIL) in the Model-based Embedded Robotics Systems (MERS) group with Prof. Brian Williams. She received her Master of Science degree from the Department of Aeronautics and Astronautics at MIT in 2025 and her Bachelor of Science degree in Mechanical Engineering, with a concentration in robotics, from the California Institute of Technology (Caltech) in 2023. Her current research builds upon the project she worked on, with the NASA Jet Propulsion Laboratory, during the summer of 2025, and focuses on autonomous robotics for health management. Her previous research projects include using mode estimation for human robot collaboration.

Zaki Hasnain received his B.Sc in Engineering Science & Mechanics at Virginia Polytechnic Institute & State University, Blacksburg, Virginia, USA. He received his M.Sc in Computer Science and Ph.D. in Mechanical Engineering from University of Southern California, Los Angeles, CA, USA.

Michel Ingham is the Chief Technologist of JPL's Autonomous Systems and Advanced Software Division, responsible for spearheading and coordinating research and technology efforts across the division, and integrating technology work more broadly across JPL. He has led several NASA, JPL and DARPA research and development activities, focused in the areas of spacecraft autonomy, digital engineering, and model-based software systems engineering. He received his Doctoral and Master's degrees from MIT's Department of Aeronautics and Astronautics, and his Bachelor's degree in Honours Mechanical Engineering from McGill University in Montreal, Canada.



Seung H. Chung directs the Ground Software & Systems Engineering section at NASA's Jet Propulsion Laboratory, where he oversees the end-to-end lifecycle of Ground Data Systems (GDS) and Deep Space Network (DSN) data services. An expert in systems engineering and formal AI methods, Dr. Chung has over 25 years of experience advancing model-based system design, automated activity planning and execution, and automated fault management for complex aerospace systems. He earned his PhD in Autonomy and SM in Aeronautics and Astronautics from the Massachusetts Institute of Technology, following a BS from the University of Washington.

Brian C. Williams is a Professor of Aeronautics and Astronautics at MIT, a principal investigator at MIT's Computer Science and Artificial Intelligence Laboratory. Prof. Williams' research concentrates on model based-autonomy – the creation of long-lived autonomous systems that can explore, command, diagnose and repair themselves using online automated reasoning coupled to machine learning. Prof. Williams co-invented the Remote Agent model-based autonomous control system, which received a NASA Space Act Award in 1999, and was a member of the NASA DS1 flight team, which demonstrated Remote Agent. He was a member of the Caltech JPL Advisory Council and Technical Division Advisory Board, and a member of the Young Panel, which assessed future Mars missions in light of the Mars Climate Orbiter and Polar Lander incidents.