Operations for Autonomous Spacecraft: Downlink Analysis of Onboard Decisions and Execution Anomalies

Sriramya Bhamidipati
Jet Propulsion Laboratory
California Institute of Technology
4800 Oak Grove Dr.
Pasadena, CA, 91109
sriramya.bhamidipati@jpl.nasa.gov

Federico Rossi
Jet Propulsion Laboratory
California Institute of Technology
4800 Oak Grove Dr.
Pasadena, CA, 91109
federico.rossi@jpl.nasa.gov

Rebecca Castano
Jet Propulsion Laboratory
California Institute of Technology
4800 Oak Grove Dr.
Pasadena, CA, 91109
rebecca.castano@jpl.nasa.gov

2024 IEEE Aerospace Conference | 979-8-3503-0462-6/24/$31.00 ©2024 IEEE | DOI: 10.1109/AERO58975.2024.10520940

***Abstract*—Future space exploration missions will heavily rely on autonomous planning and execution (APE) technology to improve spacecraft reliability and reduce operational costs. However, this will require a complete revamp of ground operations, i.e., from current practice of specifying pre-planned sequences to specifying high-level goals that will later be elaborated by the onboard APE based on spacecraft's state and perceived environment. Particularly, determining the mission outcomes during downlink is a challenging task. In this paper, we reconstruct what the spacecraft has executed onboard (i.e., as-executed) using downlinked channelized data, EVRs, and, critically, spacecraft models; We also quantitatively compare as-executed from the "actual" run with ground-based prediction simulations. To do this quantitative comparison, we design an N-dimensional dynamic time warping (DTW) technique based on which we formulate two similarity scores: (a) one related to executed tasks whose cost function is based on interval-based generalized intersection over union; and (b) other related to spacecraft states whose cost function is based on normalized Manhattan distance. Through a simulated case study of multiple flybys in Neptune-Triton system, we demonstrate that our technique successfully quantifies the similarity between the as-executed actual and predicts, and assess its "in-family" versus "out-of-family" behavior. To lower the associated false positives/negatives, we also design a multi-objective assessment metric that is a weighted summation of the task- and timeline-related similarity scores.**

TABLE OF CONTENTS

1. INTRODUCTION

Advanced onboard autonomy technologies are extremely beneficial f or f uture s pace m issions t hat l ack a n a-priori knowledge about the target space destination and/or experience dynamically-changing environmental features. Some examples of such situations include rovers exploring icy giant moons, coordinated deep space fleets, a nd f ast fl yby spacecrafts. These autonomy technologies aid in increasing science return, improving spacecraft reliability, reducing operational costs, or even achieving goals that cannot be attained through regular ground operations due to communication limitations, such as high latency and limited bandwidth, or short mission lifetime. Examples of such technologies include autonomous fault management [1], planning, scheduling, and execution [2, 3], and selection of scientific targets [4]. Designing mission operations to facilitate onboard autonomous planning and execution (APE) remains an active area of research, and is the key focus of this paper.

An APE system allows a spacecraft to plan and execute activities opportunistically, based on previous observations and onboard resource availability. For instance, an APE module can on-the-fly decide to perform follow-up observations of a detected event of interest (say, a transient plume) if sufficient power, thermal, and memory resources are available, potentially increasing science returns compared to deploying pre-planned sequences. Supporting such an APE requires a paradigm shift in ground operations [5], i.e., from the current practice of uplinking sequences to deploying only high-level goals which are onboard elaborated by an APE.

This paper builds upon our previous work on operations for autonomous spacecraft [6–9]. In these previous works, we developed and tested user interfaces related to both uplink and downlink that support ground operations for an APE. The developed downlink tools provide an explanation of the APE's onboard decisions through user interfaces that capture what decisions were made onboard, and expain why these decisions were made in response to the perceived environment and spacecraft states, and the high-level goals uplinked by ground operators. A case study on the operations of an autonomous spacecraft performing multiple flybys of the Neptune-Triton system was simulated [7] and an in-depth design analysis was conducted, wherein JPL ground operators assessed the developed user interfaces and identified directions for future development.

Determining the mission outcomes for an APE after downlink still remains a challenging task - to the author's knowledge, no quantitative tools and analyses exist as of now. The downlink operations must be able to reconstruct the APE's decisions and identify any execution anomalies. Identifying execution anomalies is a significantly more complex task compared to the current practice of confirming the nominal execution of a pre-planned sequence, whose dependency on the spacecraft state is comparatively more predictable [10]. Particularly, defining "anomalous" behavior for an APE is not straightforward as an APE might have choose from a large number of possible execution paths - this choice depends on spacecraft states and sensed environments and the ground operators are unable to predict this execution path in advance.

Contributions

In this paper, we propose new workflows that can interact with our previously-developed user interfaces and support downlink operators in the quantitative analysis of the APE's onboard decisions. Our key objectives are:

1. Reconstructing "as-executed" tasks and spacecraft states (and uncertainties) in a way intuitive for operators by exploiting downlinked channelized data, event notifications [commonly referred to as EVRs at the Jet Propulsion Laboratory (JPL)], and, critically, spacecraft models.
2. Quantitatively comparing as-executed outcomes from the "actual" run with "predictions" from ground-based Monte Carlo (MC) simulations to assess whether the APE's execution is "in-family" to what has been observed on the ground.

Note that, "as-executed actual" depicts what a spacecraft has executed during the actual mission while "as-executed predicts" refers to what has been executed by the spacecraft during ground-based MC simulations. To clarify, this paper focuses on the "execution aspect of APE and does not deal with "planning".

The quantitative tools being designed to assess the "in-family" behavior of the as-executed actual needs to have the following desirable properties: (a) shift invariant, i.e., lack of sensitivity to differences in start times between actual and ground predicts; (b) robust to mismatches in initial values, e.g., states and resource values are unlikely to perfectly match; and (c) able to deal with sparse data or different sampling rates, which depends on the available communication bandwidth from the spacecraft and the available ground resources to perform a large number of MC simulations.

To perform predicts versus actual comparison, we propose the use of Dynamic Time Warping (DTW) [11], which is a well-known technique in the field of speech recognition to cope with different speaking speeds and accents. DTW evaluates two temporal sequences (which may vary in speed or do not perfectly sync up) based on a user-defined cost formulation to provide a measure of similarity. In this paper, we design N-dimensional DTW to formulate two similarity scores, of which one is related to executed tasks and the other is related to spacecraft states. Particularly, we compare the spacecraft states between the as-executed actual and any predict by modeling the associated cost as a normalized version of the Manhattan distance [12]. Manhattan distance is a distance measure that is calculated by taking the absolute sum of distances between the coordinates along different axes. On the other hand, the similarity measure for executed tasks is based on the theory of set-based generalized intersection over union (GIoU) [13], where each task is represented as a set interval based on its start and end times of execution. GIoU is a generalized version of intersection over union (IoU) that compares the overlap between two arbitrary shapes (in terms of areas or volumes). This generalization keeps the major properties of IoU, namely non-negativity, scale invariance, symmetry, triangle inequality while rectifying its weakness (i.e., penalizing the case when no overlap exists between two shapes). The use of N-dim DTW ensures shift invariance, robustness against initial value mismatch and sparse data.

We also design a multi-objective assessment metric is a weighted summation of the similarity scores associated with tasks and spacecraft states. For increased robustness against false alarms, we independently extract a few as-executed predicts with highest match to that of actual based on tasks and spacecraft states, and thereafter, compute the multi-objective assessment metric for the ones common between both.

While it is not possible to anticipate all possible scenarios that the spacecraft will encounter, the uncertainty related to the environment (e.g., likelihood that a plume is detected) and the spacecraft itself (e.g., the likelihood of components failures) can be modelled on the ground to a certain extent. An exhaustive modeling of these uncertainties is an active research area and beyond the scope of this paper. In this paper, we assume the ground-based MC simulations to be adequately representative of the possible execution paths. With this assumption in mind, we can draw parallels between "in-family" behavior and nominal execution of an APE system, and similarly, "out-of-family" and execution anomaly.

Although our framework has been generalized to accommodate different onboard planning and scheduling technologies, in this paper, we utilize a flight-proven planning and execution system, namely MEXEC (Multi-mission EXECutive [3]). MEXEC that has been demonstrated on the ASTERIA CubeSat and was used on the JPL's Europa Lander Surface Mission Autonomy project. For validation, we utilize our previously-developed simulation study on multiple flybys of the Neptune-Triton system that has a MEXEC-based APE [7]. We perform a number of MC simulations of which one is marked "as-executed actual" (to represent in-flight APE behavior) while the rest are "as-executed predicts". We demonstrate that our tool can quantitatively evaluate the similarity between the as-executed actual and predicts, and assess its "in-family" versus "out-of-family" behavior.

Organization

The rest of the paper is organized as follows. Section 2 provides an overview on MEXEC-based APE framework. Section 3 explains the reconstruction of as-executed. Section 4 describes our N-dimensional DTW that performs quantitative comparison between as-executed predicts and actual. Section 5 discusses our simulation setup, experimental results and analysis. Section 6 provides concluding remarks and lay out directions for future research.

2. BACKGROUND

This section provides a background on MEXEC. As explained in Section 1, this paper considers MEXEC as the core APE onboard the autonomous spacecraft. In MEXEC [3], task networks are used by ground operators to represent their high-level goals. Goals are expressed in the form of tasks, which essentially indicate a desired change to the spacecraft. Tasks are identified by the following key elements: 1) unique ID, 2) a priority value to inform scheduling order, 3) the conditions (referred to as constraints) to be satisfied for a task to proceed for execution, and 4) the expected after-effects (referred to as impacts) of task execution on spacecraft states. Some examples of tasks are taking three images using wide-angle camera, logging magnetometer when Triton is in view.

The temporal sequence of each spacecraft state (also termed as a resource) that appears as either a constraint and/or an impact is referred to as timeline. Timelines are categorized either as internal, which portray states managed internally by MEXEC, or as external, which represents state values reported by the spacecraft sensors (logged in the state database). Impacts from tasks change the timeline by assigning a specific value, specific change in value or a specific change in rate of the value. State updates received from the state database are also applied to the timelines to keep them

synchronized with the latest values logged by the sensors.

An overview of the MEXEC functionality is as follows: The ground operators uplink a task network to the spacecraft, which is read by the planner of the onboard APE system. The planner schedules the tasks and checks for conflicts using the system state from the state database. When close to the start time of a task, the planner passes it to the executive, which performs real-time constraint checking to ensure safe execution of the task. Task execution updates are sent from the executive to the planner to keep the planner informed in case re-planning is necessary.

The role of the executive is as follows: The executive takes care of task execution and handles adjustments needed for any deviations in execution. The MEXEC executive converts constraints and control conditions into Boolean expression trees, which are later evaluated at each mode of task execution on a per task basis. The evaluation is based on the spacecraft state reported from the state database or estimated from internal state propagation using spacecraft models. As time progresses, each task transitions to a new mode and any faults are reported to the planner - there are a total of six modes [3]:

1. Idle: If the pre-execution constraints are not satisfied within a certain timeout interval, the task is considered to have failed. If the constraints are satisfied, then the task gets pushed to the MEXEC executive for execution. Until the task gets committed, it is considered to be in idle mode.
2. Command dispatch: The executive dispatches the command associated with the task and starts monitoring any constraints that need to be maintained during task execution.
3. Command running: The executive expects a command response to indicate that the command was dispatched to the respective subsystem/instrument and has been received successfully. The task is being executed in this stage.
4. Wait end: The command completion is received indicating the task completion by the subsystem/instrument. The executive waits until wait end condition gets satisfied.
5. Cleanup: The executive performs the clean up protocols.
6. Complete: If the command response comes back with an error, or does not come back within a defined timeout interval, the task fails. Otherwise, the task completes with success after the end duration of the task has been reached.

3. Reconstruction of As-Executed

This section explains the reconstruction of an as-executed, given downlinked EVRs, spacrcraft models and channelized data. We consider an MEXEC-based APE framework, an overview of which is explained in Section 2.

Figure 2 provides an overview of the schema of our as-executed. Our as-executed is designed to be a JavaScript file (.js) and consists of two main components: one related to executed tasks and the other related to recorded timeline values. We extract task information from EVRs and timeline values from the state database and internal state propagation of spacecraft models. The task network and the EVRs are standardized in the current release of the MEXEC (version 1.2.0 at the time of writing this paper), and hence, the same pre-processing pipelines can be utilized for any mission.

As shown in Figure 2, our schema for each task in as-executed is designed to include the following information: start, end, value and properties. The start and end correspond to times when the task transitions to the command running mode and the complete mode, respectively. The value refers to the task name, which is obtained either from task network or internally assigned by MEXEC based on task ID. The properties provide additional information related to mode transitions, task duration, task ID, task result and overall result. Particularly, the six mode transition times discussed in Section 2 are logged in as-executed, i.e., idle, command dispatch, command running, wait end, cleanup and complete. Note that inactive and commit times listed in the schema are for downlink redundancy and are not relevant for this paper's focus. These intermediate mode transition times provide ground operators with more granularity regarding how a certain task execution progresses over time. The task result outputs either failure or success of the task execution and constraint checking by the MEXEC executive. On the other hand, the overall result for any task corresponds to the failure or success in achieving the relevant science goals and in logging desired observations. For example, when the MEXEC onboard an autonomous spacecraft that is experiencing large vibrations interfaces with a wide-angle camera to take three blurry images, the task result is logged to be a success but the overall result is reported as a failure. This is because, while the executive has successfully completed all the mode transitions without any error, the science goal of logging useful science images was unsuccessful.

schema.js

```
window.timelineData =
{
    "start": "double",
    "end": "double",
    "now": "double",
    "timelines": [
    {
        "name": "tasks",
        "type": "activity",
        "value_type": "string",
        "units": [
            {"start": "double",
                "end": "double",
                "value": "string",
                "properties": { "name": "string",
                                "id": "integer",
                                "start_time": "double",
                                "duration": "double",
                                "commit_time": "double",
                                "inactive_time": "double",
                                "idle_time": "double",
                                "cmd_dispatch_time": "double",
                                "cmd_running_time": "double",
                                "wait_end_time": "double",
                                "cleanup_time": "double",
                                "complete_time": "double",
                                "task_result": "integer",
                                "overall_result": "integer" }
            }
        ]
    }
    {
        "name": "tasks",
        "type": "activity",
        "value_type": "string",
        "units": [
            {...}
        ]
    }
    {
      "ID": "integer",
      "name": "string",
      "type": "string",
      "value_type": "string",
      "interpolation": "string",
      "units": [
        {"start": "double",
        "value": "double",
        "properties": { "mean": "double",
                        "covariance": "double" }
        }
      ]
    }
    {
      "ID": "integer",
      "name": "string",
      "type": "string",
      "value_type": "string",
      "interpolation": "string",
      "units": [
        {...}
      ]
    }
  ]
}
```

Figure 1: **Overview of onboard as-executed comprising of executed tasks and recorded timelines.**

Unlike tasks which are obtained from standardized EVRs, timelines can be extracted either from EVRs or channelized data (based on mission constraints). For each timeline, we design our schema in as-executed to include the following information: ID, name, type, value-type, interpolation, start, value and properties. Similar to a task ID, timeline ID is a unique identifier for each timeline, wherein a positive ID number represents an external timeline and negative ID indicates an internal one. The type, value-type and interpolation of any timeline are obtained from task network. The "start" corresponds to the time at which the timeline "value" has been reported by the state database. For internal timeline whose value change is only triggered by tasks, the start and

value in as-executed aligns with either the task start time and pre-condition impact value or the task end time and post-condition impact value. These pre- and post-condition impact models are defined in task network. Under properties, the mean and covariance estimates of the timeline are logged. We estimate the mean and covariance using a Kalman filter-based state estimation technique, which consists of a measurement update and a predict step. To perform the predict step, we leverage the impact models pre-defined in the task network. On the other hand, we utilize the extracted timeline values from EVRs and/or channelized data as measurements to perform the measurement update step. These state estimates provide interesting insights regarding how closely the impact models comply with the actual spacecraft states when a certain task is executed. Additionally, the state estimates are also beneficial in interpolating the timeline behavior in the event of an onboard anomaly or missing data during downlink.

4. Predicts Versus Actual Comparison

To assess whether what an autonomous spacecraft has executed onboard falls in-family with what is expected, we perform a quantitative comparison between as-executed predicts and actual. As explained in Section 1, we design an N-dim DTW to compute two similarity scores: one based on comparison of tasks between as-executed actual and predicts and the other based on comparison of timelines between as-executed actual and predicts. We also design a multi-objective assessment metric that is a weighted average of task and timeline-based similarity scores.

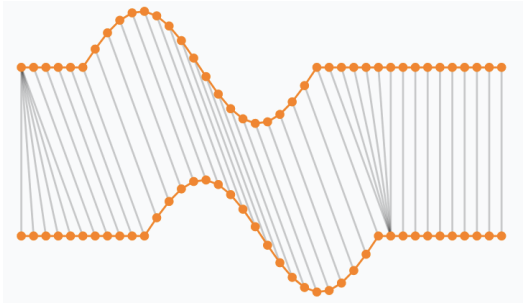


Figure 2: **Illustrative of a 1-dim DTW taken from [14].**

Considering Δt to be the sampling rate of the executive and T_{total} to be the scenario duration, we denote $K = 1 + \frac{T_{total}}{\Delta t}$ to be the number of discrete timesteps at which tasks and timelines are sampled. Algorithm 1 shows the pseudocode for N-dim DTW that considers two N-dim temporal sequences of length K and cost function C as inputs and outputs the similarity score $p \in \mathbb{R}$. We denote the task and timeline related similarity scores by $p_{task} \in [0,1] \subset \mathbb{R}$ and $p_{timeline} \in [0,1] \subset \mathbb{R}$, respectively. Note that values closer to 1 for p^j_{task} and $p^j_{timeline}$ indicate a higher similarity between the "as-executed actual" and any jth predict, while lower values closer to 0 indicates otherwise. To obtain a dissimilarity score instead, remove 1 in the return output of Algorithm 1.

To execute N-dim DTW, we need to model the cost formulae for tasks and timelines, which are explained in the below subsections.

Algorithm 1 Pseudocode for N-dim DTW for a given cost

Two sequences $\mathbf{A}, \mathbf{B} \in \mathbb{R}^{K \times N}$ and cost function C

for $p \in [0, \ldots, N]$ **do**
for $q \in [0, \ldots, N]$ **do**
  $dtw[p,q] = inf$
end for
end for

for $p \in [0, \ldots, N]$ **do**
for $q \in [0, \ldots, N]$ **do**
  $dtw[p,q] = C(\mathbf{A}_p \in \mathbb{R}^{1\times N}, \mathbf{B}_q \in \mathbb{R}^{1\times N}) +$
  $min(dtw[p-1,q], dtw[p,q-1], dtw[p-1,q-1])$
end for
end for

return $p = 1 - \big(dtw[N,N]/max(dtw)\big)$

Task: Cost Formulation using GIoU

To apply the N-dim DTW technique, we need to first manipulate the task representation from that based on start, end and mode transition times (in as-executed) to that of a temporal sequence. Given N tasks, we represent each nth task as a discrete, temporal sequence $\mathbf{X} \in \mathbb{R}^{K \times N}$ comprising of values among $\{1,2,3,4,5,6\}$. For any nth task at any kth time step ($k \in K$), we define $1,2,3,4,5,6$ to represents the modes in the following order: idle, command dispatch, command running, wait end, cleanup and complete.

The cost function used in N-dim DTW for comparing executed tasks in as-executed actual with that in any jth predict (considering a total of M predict simulations) is defined as follows (note that cost is lower if there exists higher similarity between actual and predict):

$$C(\mathbf{X}_{actual}, \mathbf{X}_{predict,j}) = \sum_{i}^{\{1,2,3,4,5,6\}} w^i_{task}\, tGIoU\Big(A(\mathbf{X}_{actual}, i), A(\mathbf{X}_{predict,j}, i)\Big) \tag{1}$$

where w^i_{task} denotes the user-defined weight given to any ith mode such that $\sum_i w^i_{task} = 1$, $tGIoU(\cdot,\cdot)$ denotes the task-based GIoU metric (see Algorithm 2) and $A(\mathbf{X}, i)$ (see Eq. (2)) denotes the N-dim set interval formulated from a task-based temporal sequence $\mathbf{X}$ and the desired ith mode. Note that, $tGIoU$ ranges between $[0,1]$ with 1 indicating a close similarity between any ith task mode being compared while 0 indicating otherwise.

$$\begin{aligned} A(\mathbf{X}, i) = &[find\_first(\mathbf{X}^1 = i), find\_last(\mathbf{X}^1 = i)] \times \cdots \\ &[find\_first(\mathbf{X}^n = i), find\_last(\mathbf{X}^n = i)] \times \cdots \\ &[find\_first(\mathbf{X}^N = i), find\_last(\mathbf{X}^N = i)] \end{aligned} \tag{2}$$

where $find\_first$ denotes the first time step where the temporal sequence $\mathbf{X}$ equals i, which in-turn equals the time reported for the ith mode in as-executed .js file. Similarly, $find\_end$ denotes the last time step where the temporal sequence $\mathbf{X}$ equals i, which is in-turn the time reported for the $i+1$th mode in as-executed .js file. See Section 3 for more details on the schema for the as-executed .js file.

Essentially, we compare the tasks across two as-executed by analyzing the similarity between the six task mode duration. This is beneficial in penalizing the tasks that are successfully dispatched to the spacecraft subsystem/instrument but do not reach complete mode due to errors or timeouts. Note that our current cost formulation does not penalize the tasks whose overall result is different from that of task result - this will be explored in our future work.

Algorithm 2 Pseudocode for task-based GIoU metric $tGIoU(\mathbf{A}_1, \mathbf{A}_2) \in [0, 1]$, refer to [13] for more details.

Two set intervals $\mathbf{A}_1, \mathbf{A}_2 \in \mathbb{R}^N$

Given $\mathbf{A}_1$ and $\mathbf{A}_2$, find the smallest enclosing convex object $\mathbf{C} \in \mathbb{R}^N$.

$$IoU = \frac{\left|\mathbf{A}_1 \cap \mathbf{A}_2\right|}{\left|\mathbf{A}_1 \cup \mathbf{A}_2\right|}$$

$$g = IoU - \frac{\left|\mathbf{C} \setminus (\mathbf{A}_1 \cup \mathbf{A}_2))\right|}{|\mathbf{C}|}$$

return $1 - g$

Timelines: Cost Calculation using NMD

Given N number of total timelines, we represent each nth timeline as a continuous, bounded, temporal sequence $\mathbf{Y} \in \mathbb{R}^{K \times N}$. The bounds for each nth timeline are defined in task network. Utilizing the theory of NMD, we formulate the cost for timeline comparison as follows (note that cost is lower if there exists higher similarity between actual and predict):

$$C(\mathbf{Y}_{actual}, \mathbf{Y}_{predict,j}) = \sum_{n \in N} w^n_{timeline} \frac{\left|\tilde{\mathbf{Y}}^n_{actual} - \tilde{\mathbf{Y}}^n_{predict,j}\right|}{2} \quad (3)$$

where $w^n_{timeline}$ denotes the user-defined weight given to any nth timeline such that $\sum_{n \in N} w^n_{timeline} = 1$ and $\tilde{\mathbf{Y}}^n_{actual} = \frac{\mathbf{Y}^n_{actual} - l^n}{u^n - l^n} \in [0, 1]$ and $\tilde{\mathbf{Y}}^n_{predict,j} = \frac{\mathbf{Y}^n_{predict,j} - l^n}{u^n - l^n} \in [0, 1]$ with $[l^n, u^n]$ representing the known bounds of the nth timeline. Note that, the multiplying factor 2 is added to normalize the cost function to lie between $[0, 1]$. Additionally, note that, there is no correlation between the notations N and n in this section and the earlier one on tasks, we only re-used these notations for the sake of brevity.

Multi-Objective Assessment Metric

Using Algorithm 1, 2 and Eqs. (1) and (2), we independently compute the set of predicts associated with the highest task-based similarity scores when compared against "as-executed actual". We denote these set of predicts by L_{task} and the corresponding costs by $p^1_{task}, \cdots, p^{|L_{task}|}_{task}$, where $|\cdot|$ denotes the cardinal number of the set. Similarly, using Algorithm 1 and Eqs. (3), we compute the predict set (defined by $L_{timeline}$) exhibiting highest timeline-based similarity scores, namely $p^1_{timeline}, \cdots, p^{|L_{timeline}|}_{timeline}$. Given that the impact models associated with tasks govern the timeline values, there is expected to be a significant correlation between that of task and timeline-related similarity scores. Ideally, the set of predicts exhibiting highest task-based similarity score should match that of the predict set exhibiting the highest timeline-based similarity scores, i.e., $L_{task} == L_{timeline}$. However, in reality, these two sets may not be equal given measurement noise and missing data (if any) in timeline values and unmodeled components of mode transitions in tasks.

In Eq. (4), we design a multi-objective assessment metric $p_{closest} \in [0, 1]$ that provides a quantitative measure on how close the as-executed actual is to that of predicts.

$$p_{closest} = \max_{j \in \{L_{task} \cap L_{timeline}\}} \frac{\mathbf{p}^j_{task} + \mathbf{p}^j_{timeline}}{2} \quad (4)$$

As a by-product, we can identify the $j_{closest}$ as-executed predict whose input hyperparameters can provide useful insights regarding the unknown/perceived state of the environment.

5. Experimental Setup and Results

We validate our proposed workflows, namely reconstruction of as-executed and comparison of predicts and actuals, using a simulation case study.

Simulation Setup

We utilize our prior work [6–9] on Neptune-Triton simulation setup, which involves five classes of high-level science goals with varying autonomy capabilities (see Figure 3): monitoring, mapping, targeted observations, event-driven opportunistic observations, and opportunistic monitoring. With these science goals in mind, a number of MC simulations exercising the instrument suite and variability in the perceived environment state and spacecraft initial states were defined. Some examples of variable science goals that impact the observation time, power, and available data volume include detecting plumes on the limb of Triton, variability in magnetospheric activity, and detecting storms at Neptune. This is an interesting case study because the high latency, low available bandwidth, short duration of flybys, and dynamic scientific phenomena makes the use of APE system an attractive option for fulfilling the science objectives.

Results and Analysis: Reconstruction of As-executed

We utilize our previously-developed downlink user interface, known as Plan Reconstruction tool [8], for validating and visualizing the reconstruction of as-executed. The Plan Reconstruction tool plays back what the spacecraft planned to do in incremental steps (i.e., plan) based on estimated timeline (state and resource) values, and shows what tasks were actually executed (i.e., as-executed). This can be leveraged by the downlink operators to assess what the onboard planner did and determine the cause of its decisions. In Figure 4, we overlap the as-executed with that of a plan generated by the planner for a nominal run (i.e., with no anomalies or instrument failures). To alert the ground operators of the status of executed goals, indicators show either waiting to be "scheduled" (i.e., idle mode), executing one among the other "task modes", or "success/failure" after task execution. Qualitatively, we validate that both our as-executed tasks and timelines, which are generated from downlinked EVRs, bear close resemblance to the plan generated by the planner (we see some expected variations in start times between plan and as-executed). We also observe that our as-executed accurately captures the mode transitions, which provide useful insights regarding how the task execution progresses over time and impacts different timeline values.

Results and Analysis: N-dim DTW

For this set of results, we utilize a simplified simulation scenario of the Neptune-Triton case study, wherein we generate

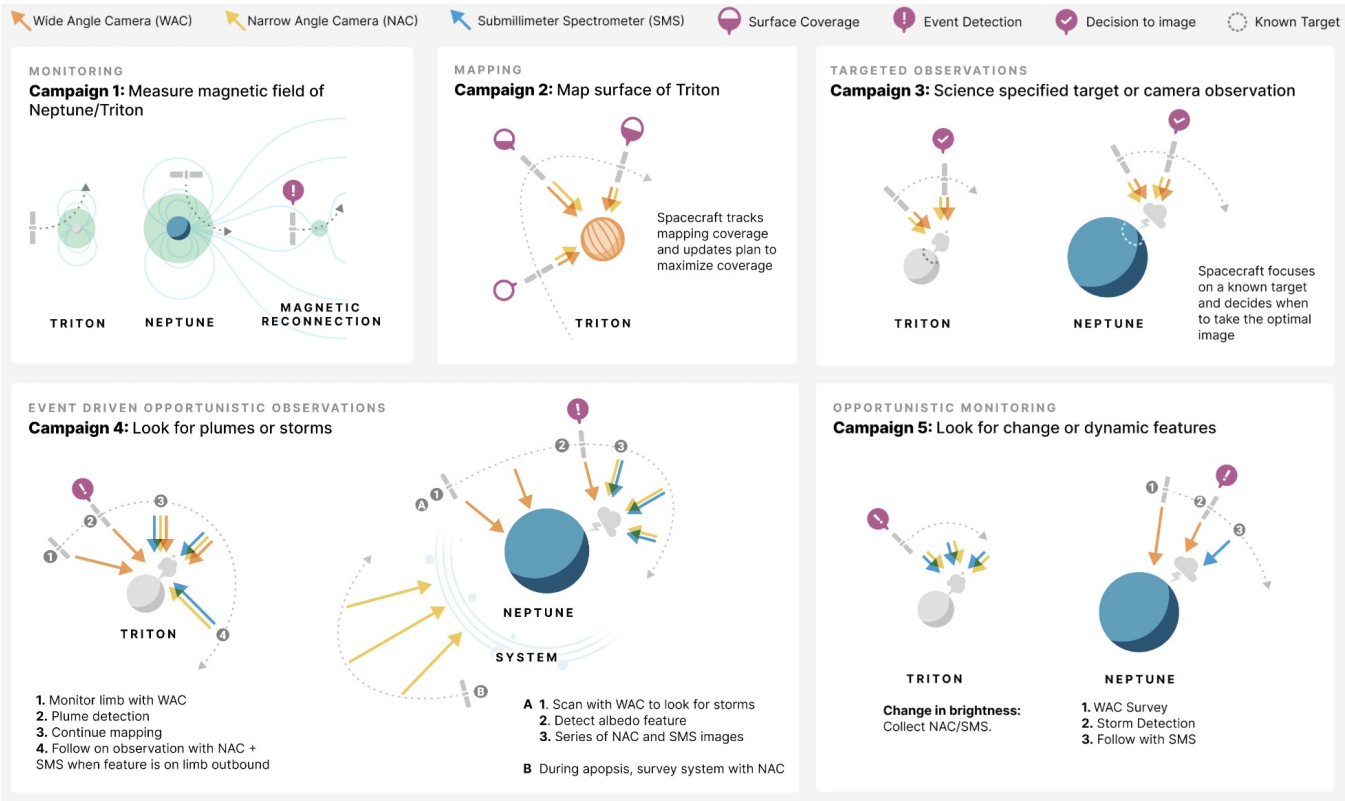


Figure 3: **Overview of science goals in our Neptune-Triton simulation study. See our prior work [8] for more details.**

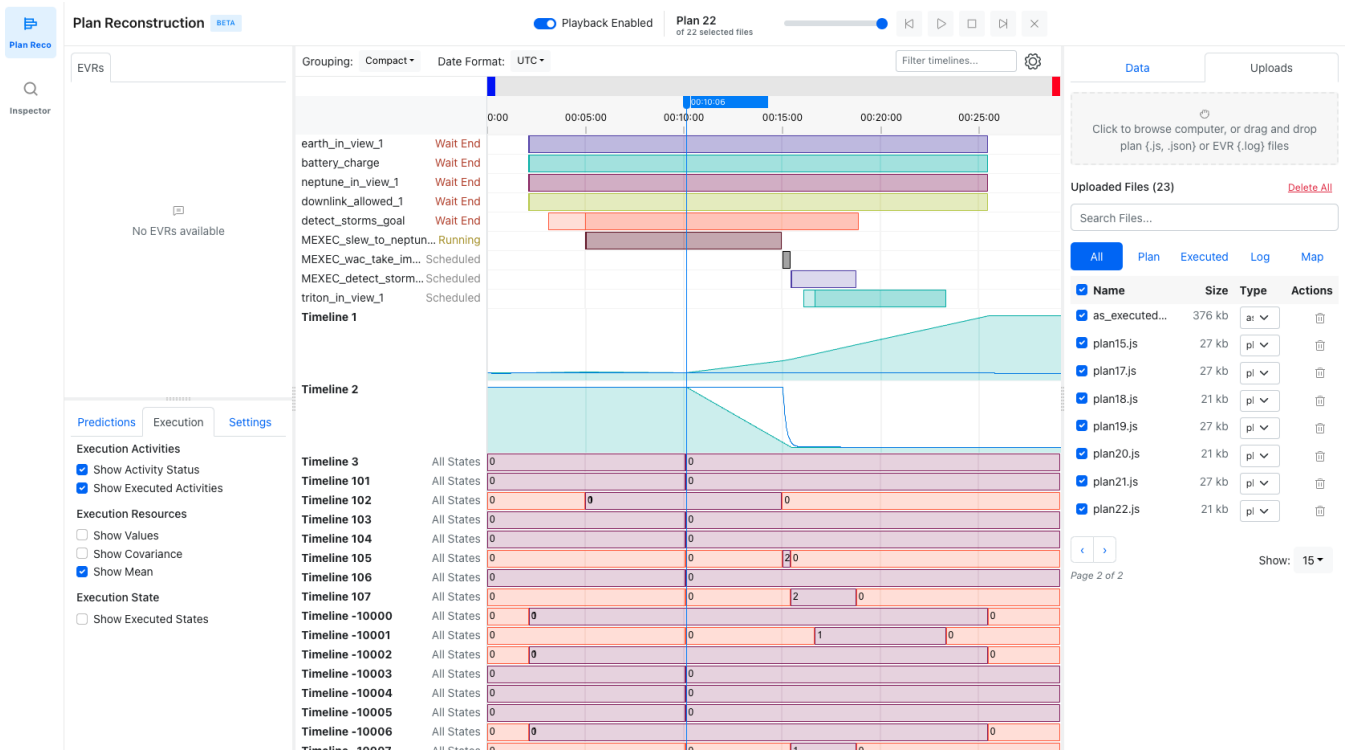
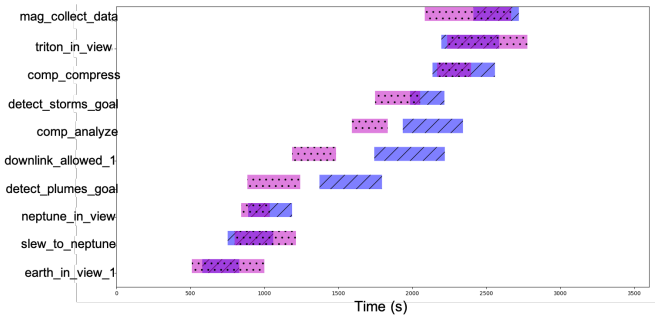
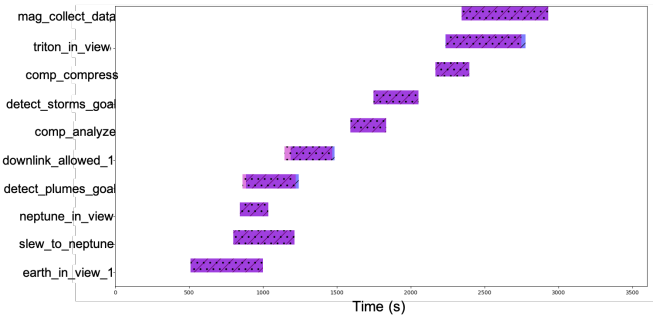


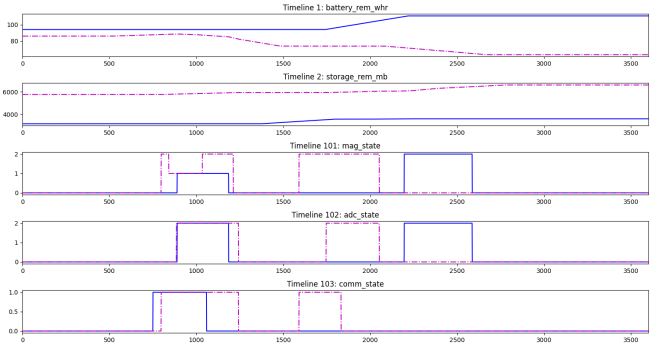
Figure 4: **Validation of the reconstructed as-executed against a plan generated by the planner.**



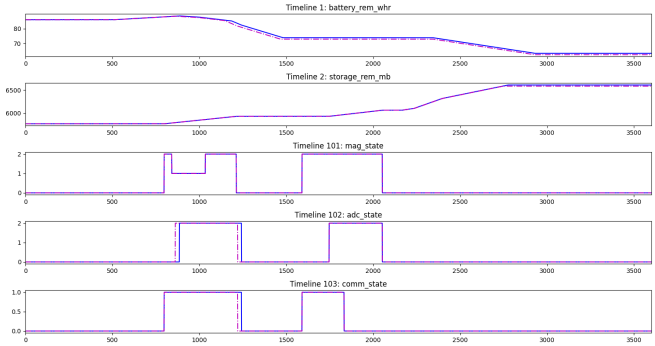
(a) **Task across base predict and "in-family" actual**



(b) **Task across base predict and "out-of-family" actual**

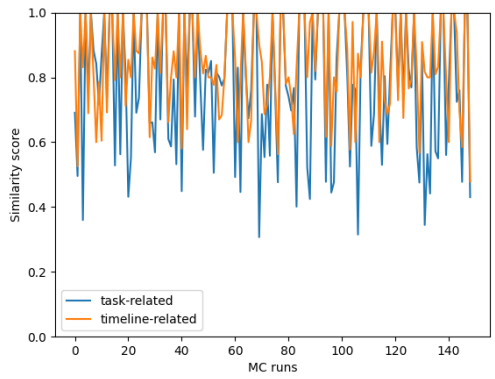


(c) **Timeline across base predict and "in-family" actual**

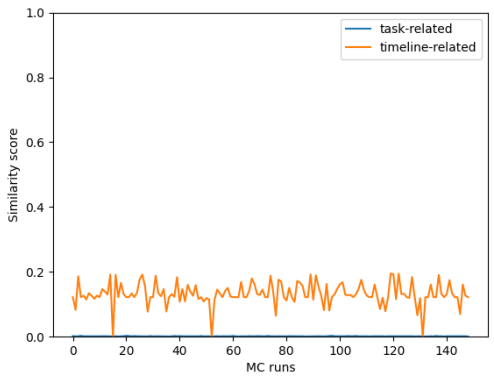


(d) **Timeline across base predict and "out-of-family" actual**

Figure 5: **Comparison of base predict (in blue with hashed markers for tasks and solid for timelines) with that of "in-family" and "out-of-family" actual run (in magenta with dotted markers for tasks and dashed for timelines).**



(a) **High similarity scores indicating in-family**



(b) **Low similarity scores indicating out-of-family**

Figure 6: **Task and timeline-based similarity scores for (a) In-family; and (b) Out-of-family comparison of as-executed actual with respect to predicts. Our proposed N-dim DTW accurately identifies the as-executed actual that lies in family with the ground-based MC simulations.**

one run termed "base predict" with 10 tasks, 5 timelines and a total duration of 1 hr. We reconstruct the as-executed of base predict run, and thereafter, emulate the tasks and timelines with simulated biases and added measurement noise to create a predict database of 150 MC simulations. Additionally, we generate two validation scenarios termed as "as-executed actual", wherein one actual run is in "in-family" with that of the MC predicts while the other is "out-of-family".

Figure 5 qualitatively compares the as-executed between one base predict (shown in blue) and the two actuals (shown in magenta). When comparing the base predict with in-family actual, most of the task execution times and timeline trends closely align, however, no such similarities are observed in comparison with out-of-family actual. Figure 5(a) and 5(c) showcases a close alignment between the as-executed tasks and timelines, thereby indicating an in-family match. On the other hand, Figures 5(b) and 5(d) showcases significant differences in the trends of both tasks and timelines, which indicates an out-of-family behavior.

We also demonstrate the quantitative comparison. When comparing as-executed predicts and with an "in-family" actual, Figure 6(a) showcases high similarity scores for both

tasks (with mean=0.78 and standard deviation=0.21) and timelines (with mean=0.85 and standard deviation=0.14). Similarly, when comparing as-executed predicts and with an "out-of-family" actual in Figure 6(b), we demonstrate that our proposed technique showcases low similarity scores for both tasks (with mean=$1.4e^{-3}$ and standard deviation=$3e^{-3}$) and timelines (with mean=0.13 and standard deviation=0.033). We, thus, demonstrate that our proposed N-DTW accurately identifies both in-family and out-of-family behavior.

6. SUMMARY

To summarize, we delve into the explainability aspect of the downlink operations for an autonomous spacecraft. We proposed new techniques that will enable the ground operators to reconstruct what an autonomous planner and executive (APE) has executed onboard (termed as as-executed actual in this paper). We also designed an N-dimensional DTW technique and subsequently, a multi-objective assessment metric, that quantitatively validates whether the as-executed actual lies "in-family" with the as-executed predicts (obtained from ground-based Monte Carlo simulations). Through simulated experiments, we successfully validated that our proposed approach can accurately estimate whether the reconstructed as-executed actual aligns with what is expected on the ground and if any anomalies have occurred.

This is the first comprehensive work that performs reconstruction of what the spacecraft's APE has executed onboard and identifies any anomalies. Our work will provide beneficial insights for the safe deployment of APE systems on future autonomous space missions. While our work has been demonstrated for an autonomous spacecraft in the Neptune-Triton system, we want to emphasize that these proposed workflows and algorithms can be generalized to other MEXEC-based missions, e.g., the Europa Lander Surface Mission Autonomy project, and also other APE systems like that to be deployed onboard the Mars 2020 Perserverance rover in near-future.

Future work will continue to focus on the integration of these new workflows and algorithms with the Plan Reconstruction tool (developed in our prior work). We also aim to conduct a detailed study on the increase in compute cost as the number of tasks and timelines increase (which is a likely scenario as the missions become more complex in the future).

ACKNOWLEDGMENTS

The research was carried out at the Jet Propulsion Laboratory, California Institute of Technology, under a contract with the National Aeronautics and Space Administration (NASA) (80NM0018D0004).

We want to thank Bennett Huffmann for their help in interfacing our as-executed with plan reconstruction tool. We also want to thank Gregg Rabideau for the insightful discussions regarding logging of downlinked data from MEXEC.

REFERENCES

[1] K. Kolcio, L. Fesq, and R. Mackey, "Model-based approach to rover health assessment for increased productivity," in *2017 IEEE Aerospace Conference*, 2017, pp. 1–13.

[2] W. Chi, J. Agrawal, S. Chien, E. Fosse, and U. Guduri, "Optimizing parameters for uncertain execution and rescheduling robustness," *Proceedings of the International Conference on Automated Planning and Scheduling*, vol. 29, pp. 501–509, may 2021.

[3] M. Troesch, F. Mirza, K. Hughes, A. Rothstein-Dowden, R. Bocchino, A. Donner, M. Feather, B. Smith, L. Fesq, B. Barker, and B. Campuzano, "Mexec: An onboard integrated planning and execution approach for spacecraft commanding," in *Workshop on Integrated Execution (IntEx) / Goal Reasoning (GR), International Conference on Automated Planning and Scheduling (ICAPS IntEx/GP 2020)*, October 2020, also presented at International Symposium on Artificial Intelligence, Robotics, and Automation for Space (i-SAIRAS 2020) and appears as an abstract. [Online]. Available: https://ai.jpl.nasa.gov/public/papers/IntEx-2020-MEXEC.pdf

[4] R. Francis, T. Estlin, G. Doran, S. Johnstone, D. Gaines, V. Verma, M. Burl, J. Frydenvang, S. Montaño, R. C. Wiens, S. Schaffer, O. Gasnault, L. DeFlores, D. Blaney, and B. Bornstein, "Aegis autonomous targeting for chemcam on mars science laboratory: Deployment and results of initial science team use," *Science Robotics*, vol. 2, no. 7, p. eaan4582, 2017. [Online]. Available: https://www.science.org/doi/abs/10.1126/scirobotics.aan4582

[5] D. Dvorak, M. Ingham, R. Morris, and J. Gersh, "Goal-based operations: An overview," in *AIAA Infotech@Aerospace 2007 Conference and Exhibit*. American Institute of Aeronautics and Astronautics, may 2007.

[6] R. Castano, T. Vaquero, F. Rossi, V. Verma, M. Choukroun, D. Allard, R. Amini, A. Barrett, J. Castillo-Rogez, N. Dhamani, R. Francis, M. Hofstadter, M. Ingham, A. Jasour, M. Jorritsma, E. V. Wyk, and S. Chien, "Operations for autonomous spacecraft: Workflows and tools for a neptune tour case study," in *Lunar and Planetary Science Conference (LPSC)*, March 2022. [Online]. Available: https://ai.jpl.nasa.gov/public/documents/papers/castano-ops-for-autonomy-LPSC-2022.pdf

[7] R. Castano, T. Vaquero, V. Verma, F. Rossi, D. Allard, A. B. R. Amini, J. Castillo-Rogez, M. Choukroun, A. Dadaian, N. Dhamani, R. Francis, R. Hewitt, M. Hofstadter, M. Ingham, C. Sorice, E. V. Wyk, and S. Chien, "Operations for autonomous spacecraft: A neptune tour case study," in *Outer Planets Assessment Group (OPAG)*. USRA, March 2021. [Online]. Available: https://ai.jpl.nasa.gov/public/documents/papers/castano-et-al-OPAG2021.pdf

[8] R. Castano, T. Vaquero, F. Rossi, V. Verma, E. V. Wyk, D. Allard, B. Huffman, E. Murphy, N. Dhamani, R. Hewitt, S. Davidoff, R. Amini, A. Barrett, J. Castillo-Rogez, M. Choukroun, A. Dadaian, R. Francis, B. Gorr, M. Hofstadter, M. Ingham, C. Sorice, and I. Tierney, "Operations for autonomous spacecraft," in *IEEE Aerospace Conference*, March 2022. [Online]. Available: https://ai.jpl.nasa.gov/public/documents/papers/castano-etal-AERO2022.pdf

[9] T. S. Vaquero, F. Rossi, R. Castano, A. Jasour, E. van Wyk, N. Dhamani, A. Barrett, B. Huffman, and M. Jorritsma, "A knowledge engineering

framework for mission operations of increasingly autonomous spacecraft," in *Workshop on Knowledge Engineering for Planning and scheduling (KEPS), International Conference on Automated Planning and Scheduling (ICAPS)*, June 2022. [Online]. Available: https://ai.jpl.nasa.gov/public/documents/papers/vaquero-et-al-KEPS2022.pdf

[10] T. Uhlig, F. Sellmaier, and M. Schmidhuber, Eds., *Spacecraft Operations*. Springer Vienna, 2015.

[11] "Dynamic time warping," in *Discrete-Time Processing of Speech Signals*. IEEE, 2009.

[12] S. Craw, "Manhattan distance," in *Encyclopedia of Machine Learning and Data Mining*. Springer US, 2016, pp. 1–1.

[13] H. Rezatofighi, N. Tsoi, J. Gwak, A. Sadeghian, I. Reid, and S. Savarese, "Generalized intersection over union: A metric and a loss for bounding box regression," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2019, pp. 658–666.

[14] R. Tavenard, "An introduction to dynamic time warping." [Online]. Available: https://rtavenar.github.io/blog/dtw.html

BIOGRAPHY

Sriramya Bhamidipati *is a robotics technologist in the maritime and multi-agent autonomy group at NASA JPL. Her research interests include GPS, computer vision, unmanned aerial vehicles and space robotics. Before joining JPL, she worked as a postdoctoral scholar in the navigation and autonomous vehicles lab at Stanford University. She received her Ph.D. in Aerospace Engineering at the University of Illinois, Urbana-Champaign in 2021, where she also received her M.S in 2017. She obtained her B.Tech. in Aerospace from the Indian Institute of Technology, Bombay in 2015.*

Federico Rossi *is a Robotics Technologist at the Jet Propulsion Laboratory, California Institute of Technology. He earned a Ph.D. in Aeronautics and Astronautics from Stanford University in 2018. His research focuses on optimal control and distributed decision-making in multi-robot systems, with applications to planetary exploration and coordination of fleets of self-driving vehicles for autonomous mobility-on-demand.*

Rebecca Castano *is the Directorate Technologist for the Interplanetary Network Directorate at JPL. In this role, she works on ensuring that relevant new technologies are developed, matured, validated, and infused. She is also the JPL Program Manager for the Center Innovation Fund (CIF). Prior to her directorate position, Dr. Castano was the Division Technologist for Division 39, the Mission Systems and Operations Division. She earned a Ph.D. in Electrical Engineering from the University of Illinois with her dissertation in the area of computer vision. She has contributed to autonomy software flying on Earth orbiters and Mars rovers.*