Risk-Aware HTN Planning Domain Models for Autonomous Vehicles and Satellites

Ebaa Alnazer, Ilche Georgievski, Marco Aiello
Service Computing Department, IAAS
University of Stuttgart
Stuttgart, Germany
e-mail: {ebaa.alnazer | ilche.georgievski | marco.aiello}@iaas.uni-stuttgart.de

Abstract*—The real world is characterised by uncertainty and risks. When modelling it as a domain for planning systems, this translates into action outcomes that can not be fully anticipated. In such environments, automating planning requires not only sophisticated algorithms but also domain models that adequately capture such complexity and unpredictability. However, existing AI planning domains often oversimplify these complexities, either because they are designed as benchmarks to evaluate planners or because they were created to test specific methods, frequently at the cost of broader realism. Autonomous vehicles and satellites are representative application examples that pose common planning challenges in dynamic, uncertain environments. Taking these, current domain models frequently omit critical features, such as uncertainty, risk, and the wide range of choices available to agents in achieving their objectives. Here, we contribute towards bringing these domains closer to reality by following a systematic approach to knowledge engineering and domain modelling that better captures these neglected aspects. Our models are implemented within the risk-aware Hierarchical Task Network (HTN) planning framework, which aligns with human-like reasoning and accommodates uncertainty and risk. By enhancing the realism of these two domains, our work increases their relevance for practical applications. Also, this work aims to drive the development of more capable AI planners and encourage the creation of more realistic domain models.

***Keywords*-Autonomous Vehicles; Satellites; HTN planning; Knowledge Engineering; Domain Models; Risk; Uncertainty**

I. Introduction

Autonomous systems such as satellites and autonomous vehicles are increasingly relied upon to perform complex tasks with minimal human intervention [1], [2]. Most satellite and driving operations often take the form of complex planning problems that go beyond familiar and straightforward tasks in everyday life, where planning is typically implicit. Whether navigating dynamic traffic scenarios or coordinating satellite activities, effective planning must account for the complexities and inherent characteristics of real-world environments, such as *uncertainty*, *risk*, and the facing of a broad range of available options for achieving goals. Failing to incorporate these factors can lead to plans that are impractical, unrealistic, and often incapable of accomplishing the intended tasks.

Accounting for these complexities is equally important when automating the planning process, which is of primary concern in Artificial Intelligence (AI) planning. In its simplest form, AI planning involves generating a course of action that, when executed in a given initial state of the world, will achieve a specified user objective [3]. The set of possible actions from which the planning system constructs this course of action is derived from knowledge about the application domain. These actions are encoded in a structured, templated representation known as a *domain model*. The domain model encapsulates the relevant domain knowledge in terms of action templates with preconditions and effects, and interactions among actions. The domain model becomes even more intricate when incorporating more features, such as uncertainty, or defining different levels of abstraction, such as in the case of Hierarchical Task Network (HTN) planning [4]. As a consequence, the practical utility of AI planning is tightly coupled with the precision, completeness, and correctness of the engineered domain model, as it directly affects the AI planning system's capabilities to produce and execute valid plans [5].

Engineering adequate domain models remains one of the major barriers to the wider adoption of planning technologies [6], [7]. This challenge stems from three key issues: **(1)** the lack of standard methodologies, tools, and frameworks to support the knowledge-engineering process [5], [8], **(2)** the prevalence of domain models designed for benchmarking and evaluating AI planning systems, rather than real-world applicability, and **(3)** the limited focus on capturing realistic domain aspects [9]. As a consequence of **(1)**, domain models are often developed in an ad-hoc manner, heavily relying on the expertise of knowledge engineers and the tools they use [8], [10]. As a consequence of **(2)**, benchmarks are oversimplified domain models with features that match the capabilities of the AI planners, e.g., the Satellite benchmark HTN domain model omits uncertainty [11]. As a consequence of **(3)**, the coverage of relevant application domains is limited, e.g., the autonomous vehicles domain remains largely unexplored in the AI planning literature.

To address these challenges, we systematically engineer and model two planning domains: Satellite and Autonomous Vehicles (AVs). In response to **(1)**, we follow a systematic approach for developing the two domain models aligned with existing knowledge-engineering processes [5], [10], [12], apply the conceptual framework for capturing realistic aspects in planning domains [9], and adopt the *risk-aware Hierarchical Task Network (HTN) planning framework* [13], which enables explicit modelling of risk and uncertainty through probability distributions over action costs, while leveraging the expressiveness and performance strengths of HTN planning itself. To move beyond **(2)**, we extend the benchmark Satellite domain model featured in the International Planning Competition (IPC) of 2020 by incorporating the realistic aspects of risk in terms of action costs, uncertainty, and variety of possible choices. To address **(3)**, we develop a domain model for AVs, which captures various realistic driving tasks and conditions. Lastly, we not only model both domains using a standard planning language, but also extend the Hierarchical Domain Definition Language (HDDL) [14] to allow specifying a probability distribution of action costs.

The remainder of the paper is organised as follows. Section II provides the necessary background. Section III presents the related work. Sections V and IV provide details about the knowledge engineering and modelling of the Autonomous Vehicles and Satellite domains, respectively. Section VI contains concluding remarks.

II. Background

A. Knowledge Engineering and Modelling in AI Planning

AI planning is a knowledge-based technique, meaning that to compute plans, an AI planning system requires relevant and adequate knowledge about the domain in which it is supposed to act [8], [12]. Engineering and modelling such knowledge constitute the first phases of the design and development of a deployable AI planning system [12]. In the first phase, relevant requirements should be identified and defined. Having such requirements is of utmost importance as it affects the adequacy of the intended domain model and the suitability of the planning system to address the challenges of the application domain. Thus, this phase is crucial as it provides the ingredients necessary to select a suitable planning type, design a planning domain model, and design or select the planning system. The main concern of the second phase is the selection of a suitable planning type; in our case, this is risk-aware HTN planning. In the third phase, the knowledge-engineering process focuses on formulating the domain knowledge to construct a domain model [8]. The domain model is an abstract conceptual description of the application domain of interest used to represent knowledge within a planning application. This conceptual description comes from the requirement specification obtained in the first phase and covers the dynamics of the domain, the kind of problems the planning engine will have to solve, and the kind of plans (solutions) that need to be provided as output [10]. Then, an explicit formal representation or encoding is created.

The domain model formally describes the persistent knowledge and represents entities invariant over every planning problem [10], [15]. These include objects with their relations and properties, actions that can change the state of the environment, and other constructs, such as tasks in HTN planning. A corresponding problem instance is needed to formally describe particular planning scenarios, which include the initial world state and the goal to be achieved. Domain models and problem instances are encoded in a de-facto standard syntax, such as HDDL and Hierarchical Planning Definition Language (HPDL) [16].

B. Realistic Aspects in Planning Domains

Accurate knowledge encoding and management in the third phase is crucial, as poor or incomplete knowledge can result in domain models that misrepresent the application domain [17], ultimately producing plans that fail in real-world execution [18]. Thus, a crucial step in the requirement analysis is the identification of relevant aspects characterising the application domain. We therefore apply an existing conceptual framework for identifying and categorising aspects of real-world planning domains, enabling the requirements analysis and aiding the knowledge-engineering process [9].

In this conceptual framework, one important aspect is the hierarchical relationship between tasks, where higher-level tasks are abstractions that can be decomposed into subtasks. The hierarchy naturally introduces structured causality, enabling reasoning across different abstraction levels. Additionally, multiple refinement options to achieve the same high-level task must be considered, where certain refinements may only be valid under specific constraints [11].

Other realistic aspects include the inherent uncertainty in real-world planning domains. It is especially important to study the sources of this uncertainty. When considering the executing agents, whether systems, humans, or a combination, uncertainty can originate internally, from the agent itself (e.g., system reliability, human limitations, irrational behaviour), or externally, from environmental factors beyond the agent's control. Both internal and external sources can be classified as either random (stochastic, rare, and unpredictable) or regular (pattern-driven and consistent).

Executing actions alters the environment and incurs costs, i.e., consumes resources, such as money, time, fuel, or effort, which are predictable in a certain world. Under uncertainty, however, action costs become variable, i.e., they are not always the same each time an action is performed. This variability stems from different sources of uncertainty: if the source is random, cost variability is unpredictable and cannot be addressed in offline planning. If the source is regular, the variability can be better defined. When cost distributions of actions are known or statistically inferable, actions are considered *risk-inducing*. When distributions instead reflect the decision maker's beliefs, the actions are *uncertainty-inducing*.

C. Risk-Aware HTN Planning

In [13], we developed risk-aware HTN planning, a framework that extends classical HTN planning with constructs that consider the uncertainty of real-world environments. It enables modelling of risk- and uncertainty-inducing actions through probability distributions over their costs and effects, where costs are defined as unbounded negative functions. The framework can be tailored for planning problems where actions have deterministic effects but variable costs. In our domain models, we adopt this variation as an initial step toward incorporating risk and uncertainty into HTN planning domain models.

The cost functions of actions can be of different types depending on the factors/sources of the costs. The cost function can be (1) *external* ($c^{ie}(a)$), i.e., it depends on external factors not explicitly modelled in the domain, such as market electricity prices or taxes, (2) *state-dependent* ($c^{is}(a)$), i.e., it is based on the system's current state, like the vehicle's charging level or position, (3) *constant* ($c^{ic}(a)$), i.e., it remains the same for every execution of an action, such as a fixed charging price, or (4) external and state-dependent, i.e., a *hybrid function* ($c^{ies}(a)$), which depends on both current state and external factors, where a denotes an action.

III. Related Work

Existing Satellite and AVs domains exhibit the discussed issues, that is, the limited realism. A version of the Satellite domain is modelled for the HTN planning track in the IPC 2020 [19], which, unfortunately, like most benchmark domains, tends to oversimplify several real-world aspects, such as risk and uncertainty, to enable planners to find valid solutions and evaluate their performance. In our proposed model, we build on this version by analysing sources of uncertainty and their effects on action costs, allowing us to incorporate risk. We also expand the range of tasks and increase the number of alternative methods for completing space missions. In our previous work, we focused particularly on extending the Satellite domain by providing alternatives for how captured images of spatial phenomena are sent to Earth, but did not consider uncertainty and risk [11]. In [20], the authors model a Satellite domain for onboard and online planning using a language based on an extended HTN representation. While their model captures several real-world aspects, such as unexpected events and resource consumption, it does not incorporate risk modelling, as planning is assumed to occur online. Another line of work on the Satellite domain focuses on modelling TV and communication satellites (e.g., [21]), which is semantically different from the space exploration satellite domain we propose.

Existing works that model the domain of autonomous vehicles for AI planning are rather scarce. In [11], we model an HTN planning domain for autonomous vehicles, taking into account various autonomous driving tasks, but do not consider realistic aspects such as uncertainty and risk. Several studies have explored traffic control problems in classical planning and extensions of it, focusing on automating multi-vehicle navigation to manage traffic [22]–[26]. Route planning research in AI planning is also relevant to the AV domain, with many works addressing marine environments and incorporating uncertainty (e.g.,[27]). Although autonomous underwater vehicles share some route planning tasks with autonomous vehicles, their environments and other tasks differ significantly.

IV. Satellite

The Satellite domain involves space applications with autonomous orbiting spacecraft that perform tasks such as imaging, data collection, navigation, or scientific research. The domain we extend originates from the partial-order track of the IPC-2020 benchmark for HTN planning, based on a NASA application where satellites conduct stellar observations by capturing images of spatial phenomena [19].

We chose this domain because it exemplifies real-world complexities and challenges for planning, including multiple satellite missions under strict resource constraints, namely, limited power, restricted target access, constrained time windows for downlinking data to ground stations, and high operational costs [28], [29].

A. Relevant Real-World Aspects

Following the vision presented in [11], the original domain is enhanced. We begin by identifying and gathering relevant aspects using the conceptual framework we proposed in [9], with particular emphasis on unaddressed factors in the original domain, such as additional stellar observation tasks and their associated complexities and interdependencies; sources of uncertainty; domain-specific quantities such as resources and variable action costs; and the objectives pursued by autonomous satellites.

Satellites perform multiple tasks to make stellar observations. These tasks are of multiple abstraction levels and have structured causality, where complex tasks are achieved by performing multiple subtasks [9]. Basically, the satellites are equipped with several observation instruments, each of which has specific modes like infrared, spectrograph, X-ray, and thermography, and has defined calibration targets (directions). Performing a space mission is a complex task that involves preparing a satellite and then taking an image. Preparing the satellite requires routing energy to an instrument, properly calibrating the instrument, and turning the satellite in the direction of the phenomenon to be captured. Finally, the satellite can capture an image of the targeted phenomenon. In actual operations, it is often necessary to activate multiple instruments simultaneously due to limitations in time and resources [30]. Therefore, the satellite must be capable of powering on several instruments at once and should be able to choose how many instruments to activate, taking into account, for example, the possibility of power failure.

Satellite mission complexity arises from the inherent uncertainty of space environments and the technologies used for the satellite operations and its instruments. Planning stellar observations requires accounting for uncertainties and their inherent randomness—specifically, what is known during the planning phase versus what becomes known only at execution time. Key sources of uncertainty, and their impact on resource consumption (e.g., time and power) include: (1) *Internal sources*, such as limitations of physical sensors that affect data quality (e.g., resolution, accuracy, noise), introducing uncertainty into action costs when these are tied to data quality [31]. Other internal uncertainties arise from the potential for power system failures (e.g., battery or solar array malfunctions [32]), which can result in power loss during the execution of actions like powering multiple instruments on the same satellite platform [33], ultimately increasing execution time. (2) *External sources* stem from the satellite's environment. Space weather events can induce anomalies such as temporary outages, power failures, and solar cell degradation [34]. Additionally, solar energetic particles or protons can penetrate satellite electronics and cause electrical failures [35].

The satellite's goal is to perform stellar observations that optimise factors like mission time and power consumption, while considering several aspects (e.g., risk and uncertainty), and the satellite and its instruments' states.

B. Domain Model Formulation

Next, we formulate the gathered knowledge as a prerequisite to model the Satellite domain.

1) Stellar Observation Tasks – Complexities and Relations: Knowledge about stellar observation tasks is represented as compound and primitive tasks. Compound tasks capture abstraction levels, causality, recursion, conditions, and alternatives, and can be decomposed by various methods into subtasks.

An abstract (or general) graphical overview of this HTN domain model is shown in Figure 1, where blue nodes represent compound tasks, orange nodes represent primitive tasks, and grey nodes $v_{m_0}, v_{m_1}, \ldots, v_{m_{11}}$ represent methods. Nodes encircled by green are our extensions. The domain includes a single top-level task, `do_observation`, which achieves a stellar observation mission by preparing the instrument and taking an image. This task can be decomposed using four methods, v_{m_0} to v_{m_3}, which represent different situational contexts rather than alternative strategies of performing the observation task. Their main differences lie in the preparation phase. Specifically, v_{m_0} includes `activate_instrument`, `turn_to`, and `take_image`; v_{m_1} omits instrument activation, assuming it is pre-calibrated, possibly from previous observations; v_{m_2} skips turning; and v_{m_3} performs only `take_image`.

Each satellite can have multiple instruments onboard. In the original domain, only one instrument can be powered per satellite. This is governed by switching off any active instrument before powering on another, and making the power unavailable after switching on an instrument. This behaviour is modelled through the methods v_{m_4} and v_{m_7}. v_{m_4} decomposes `activate_instrument` into `switch_off`, `switch_on` to switch off an already powered instrument and switch on the instrument corresponding to the required mode, respectively, and `auto_calibrate`. The latter is further decomposed by $v_{m_{10}}$ and $v_{m_{11}}$, where $v_{m_{10}}$ decomposes the task into `turn_to` to slew the satellite into the calibration direction and `calibrate` to calibrate the instrument, while $v_{m_{11}}$ skips turning, assuming the satellite is already aligned. v_{m_7} is applicable when there is no powered-on instrument, thus it skips switching off.

Real-world satellite missions often require activating multiple instruments simultaneously due to time and resource constraints [30], a capability missing from the original domain. This calls for additional methods for the `activate_instrument` task, allowing satellites to power multiple instruments concurrently and incorporating permissive decomposition into the domain model.

Powering several instruments in parallel risks power failure/loss if the satellite's energy source (e.g., solar panels, Radioisotope Thermoelectric Generators, or helium cells) cannot meet demand [33], potentially causing delays to recover. To model this, we add methods v_{m_8}, v_{m_9}, v_{m_5}, and v_{m_6}. v_{m_8} and v_{m_9} decompose the task into `overload` or

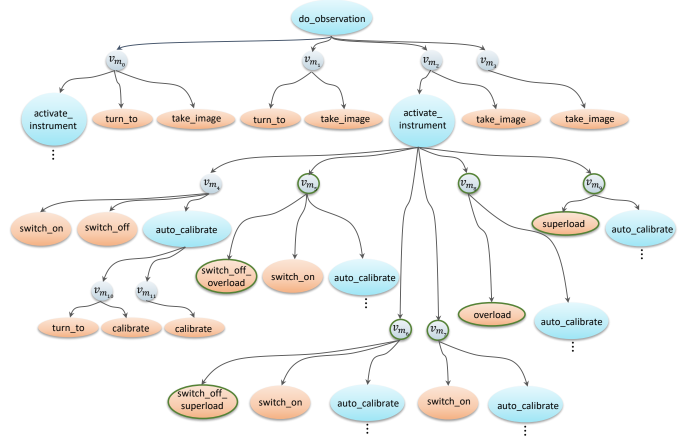


Figure 1. HTN model for the Satellite domain.

`superload` (to switch on second or third instruments), followed by `auto_calibrate`. v_{m_5} and v_{m_6} decompose the task into `switch_off_overload` or `switch_off_superload` to switch off two or three instruments at once, respectively, followed by `switch_on` and `auto_calibrate`. Thus, v_{m_4} and v_{m_8} are alternatives: to power a second instrument, the planner can either switch off the first or overload the satellite. Likewise, v_{m_5} and v_{m_9} offer a choice between switching off two instruments or adding a third, resulting in a "superload" with three active instruments.

2) Non-determinism and Action Costs: To model non-determinism and action costs, we analyse how internal and external uncertainty sources affect actions that rely on a stable satellite power system. We focus on the cost variability of `overload` and `superload`, which place higher demands on power. Specifically, considering that costs are indicative of the time required for actions to be completed, activating a second instrument via the `overload` action has a 99% probability of taking 15 units of time and a 1% probability of taking 100 units of time due to power failure. `superload` is riskier as the unfavourable outcome incurs more often and has a higher likelihood of occurrence, with a 95% probability of taking 15 units of time and a 5% probability of taking 400 units of time. All other actions are assigned a fixed cost of 15 units with 100% probability, enabling a clear comparison of planner behaviour under different risk attitudes concerning the simultaneous or sequential activation and deactivation of instruments. Here, we use the constant cost function type $c^{ic}(a)$. This means that the action costs are defined within the domain model and remain consistent across different problem instances. However, other types of cost functions and alternative cost distributions can also be considered, as the statistical inferences used to derive them may vary across different satellite domains.

3) Planning Problems Instances: Knowledge about planning problems is formulated so that the initial state contains the knowledge about the static and dynamic environment states. The static state includes each satellite's instrument, supported modes, and the calibration target of each instrument. The dynamic state includes the available power and current pointing direction of individual satellites. The initial task network includes all required observation missions, each targeting a phenomenon with a specified mode.

C. Domain Model Encoding

We model the planning knowledge formulated earlier directly into risk-aware HTN planning constructs. Listing 1 shows part of the domain model, including the `switch_on`, `overload`, and `superload` actions. `switch_on` is executed when the satellite's power is available and makes it unavailable. In contrast, `overload` and `superload` can be executed when the power is unavailable, but introduce a risk of power failure. That is, we assume that the `(power_avail)` predicate represents a restriction of maximum power for safety purposes that can be ignored when choosing to overload or superload the satellite. Recovery time from failure is modelled using probabilistic cost distributions via the : $costdist$ construct, followed by the probability distribution in the form of $(or(p_1(c_1)(p_2(c_2))\dots(p_n(c_n))$.

Listing 1. Switch on, overload, and superload actions in the Satellite domain.

```
(:action switch_on
 :parameters (?so_i - instrument ?so_s - satellite)
 :precondition (and (on_board ?so_i ?so_s)(power_avail ?
     so_s))
 :effect (and (power_on ?so_i)(not(calibrated ?so_i))(not(
     power_avail ?so_s)))
 :costdist (or (1(15))))

(:action overload
 :parameters (?so_i - instrument ?so_s - satellite)
 :precondition (and (on_board ?so_i ?so_s)(not(overloaded ?
     so_s))(not(power_avail ?so_s)))
 :effect (and (power_on ?so_i)(overloaded ?so_s)(overloads
     ?so_i ?so_s)(not(calibrated ?so_i)))
 :costdist (or (0.99(15)) (0.01(100))))

(:action superload
 :parameters (?so_i - instrument ?so_s - satellite)
 :precondition (and (on_board ?so_i ?so_s)(overloaded ?so_s
     )(not(power_avail ?so_s))(not(superloaded ?so_s)))
 :effect (and (power_on ?so_i)(superloaded ?so_s)(
     superloads ?so_i ?so_s)(not(calibrated ?so_i)))
 :costdist (or (0.95(15)) (0.05(400))))
```

V. Autonomous Vehicles

Autonomous vehicles are transportation means, typically for humans or working under human delegation, that can navigate without or with little human direct control. We choose this application domain as it exhibits realistic characteristics commonly found in real-world scenarios, many of which present significant challenges for both the modelling and solving of planning problems. These challenges originate from the domain's inherent complexity and uncertainty, such as road incidents and varying road conditions, risk factors, diverse driving tasks, resource constraints like travel time and fuel or charge levels, and the critical need to track the vehicle's state and its environment.

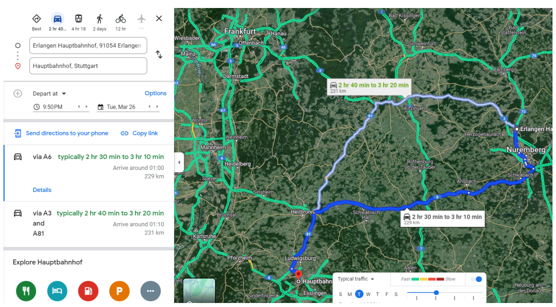
A. Relevant Real-world Aspects

We start by covering the aspects of driving tasks and their complexities and relations, the non-determinism in the domain, including the sources of uncertainty and their randomness, quantities in this domain, including resources and action costs and the variability of action costs as a consequence of non-determinism, and the objectives of an autonomous vehicle.

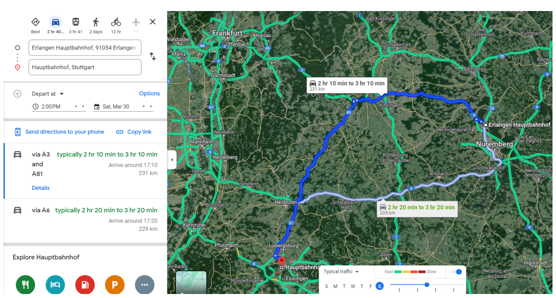
An autonomous vehicle performs various driving tasks to navigate routes and reach required destinations successfully. These driving tasks are of multiple abstraction levels and have structured causality, where complex tasks are achieved by performing multiple subtasks, such as route planning, navigation, and vehicle control [9]. For example, reaching a destination may require moving between intermediate locations, stopping, starting the engine, and managing turn signals. The vehicle must also handle road contingencies (e.g., pedestrians, construction), which consist of subtasks like stopping, dodging the incident, or restarting the engine. Environmental factors such as slippery roads introduce further complexity, requiring actions like activating the Electronic Stability Program (ESP) and adjusting speed. As in many real-world domains, these tasks can be achieved in various ways. For example, a vehicle might address poor road conditions by decelerating or accelerating with or without activating the ESP, and it may choose from various routes to reach a destination.

The complexity of AV driving tasks arises mainly from the dynamic and uncertain environment common to real-world domains [9]. Planning these tasks must account for uncertainty sources and their randomness, i.e., the amount of knowledge that can be defined when planning the driving tasks. Following our previous work [9], [13], we categorise uncertainty sources based on the vehicle's autonomy level: (1) non-autonomous (human-driven), (2) fully autonomous (no human intervention), and (3) semi-autonomous (shared control). These uncertainties may be internal, originating from the agent performing the tasks, or external, from the environment. (1) For human drivers, internal regular uncertainties often relate to variations in driving skills, habits, intentions, tactics, and speed. For instance, travel time and energy consumption can vary depending on a driver's speed preferences, habits, and tactics. Additionally, fatigue may lead to accidents, such as falling asleep at the wheel. The consequences caused by driver drowsiness have been statistically studied [36]. Unlike regular sources that can be statistically anticipated, some internal sources are random and rare, such as a driver having a stroke, making them difficult to predict. (2) For fully autonomous systems, internal regular uncertainties may result from control variability, such as the vehicle's ability to stabilise on a slippery road, leading to variable driving times and consumed energy. There are also some random internal sources that lead to unpredictable outcomes. For example, a flat tire will cause the car to stop. (3) Finally, semi-autonomous vehicles inherit a mix of these uncertainties, as both the human driver and the autonomous system contribute to the vehicle's operation.

External sources of uncertainty, both regular and random, can also affect cost variability. For example, a vehicle may fail to charge due to an unexpected station malfunction or encounter an unplanned roadblock from an accident. Such rare events are difficult to predict during planning, making action outcomes uncertain. Now consider external sources of uncertainty that are regular, such as weather conditions. Weather conditions, for instance, change constantly and cannot be predicted with full certainty. Due to the chaotic nature of the atmosphere and limitations in observation and modelling, forecasts inherently include uncertainty [37]. To express this, weather is often reported using probabilistic forecasts, where elements like temperature, wind, and precipitation are probabilistically quantified [38]. Another regular external source of uncertainty is traffic, which significantly impacts action cost variability. For example, when planning a route with Google Maps, the most used route planning application, estimated travel times for the same route vary due to regular factors like traffic. As shown in Figure 2, there is an estimation of the travelling time, i.e., a range of potential travelling times, for each route the vehicle can take, and these estimates differ between weekdays and weekends, with shorter times typically observed on weekends due to lighter traffic. Similarly, queues at charging stations represent another regular external uncertainty. To improve the quality of service at charging stations, studies such as [39] predict the probability of waiting times, which directly affect charging duration and overall trip time.



(a) Travelling times on a workday [40].



(b) Travelling times on a weekend [41].

Figure 2. Variability of travelling times in Google Maps.

In addition to the factors discussed, it is crucial to account for resource consumption, i.e., action costs such as time, money, energy, effort, or even human lives, which is a common concern in real-world domains [9]. The presence of uncertainty makes these costs variable. Therefore, understanding and modelling cost variability is essential for effective planning. For instance, uncertainty in weather forecasts affects driving costs; travelling between two locations in winter may require more time and energy if it is snowing. How much knowledge we have about the probability distribution of action costs depends on how much knowledge we have about the uncertainty sources. When uncertainty is represented probabilistically, such as through weather forecasts, associated delays can also be estimated probabilistically, a concept known as risk (see Section II-B). Conversely, random uncertainty sources like unpredictable accidents lead to costs, e.g., delays, injuries, financial loss, and environmental impact, that are difficult to quantify probabilistically. Such events can reduce road capacity, increase congestion and travel time, and potentially cause further incidents [42].

Given the above knowledge, the goal of the autonomous driving task is to find routes that optimise some objective (e.g., commuting time) while considering several aspects (e.g., risk, uncertainty, and alternative choices on how to achieve tasks), the vehicle's general state (e.g., current location), the states of its components (e.g., headlights), and the various environmental factors (e.g., weather and road conditions) to promote better safety and user experience.

B. Domain Model Formulation

1) Driving Tasks – Complexities and Relations: Tasks are formulated as compound and primitive tasks, where the relation between these tasks, i.e., the abstraction levels, structured causality, recursion, conditions, and alternatives, is formulated as hierarchical levels, where compound tasks can be decomposed by various methods, which represent the ways to achieve these tasks, into subtasks (compound and primitive). Thus, we can model the various driving tasks performed by the AV to navigate routes and reach required destinations successfully as HTN tasks, forming the HTN domain model for the AV domain. An abstract (or general) graphical overview of this HTN domain model is shown in Figure 3. The domain has one task, `drive`, at the highest level of the hierarchy, which enables travel between two locations. Three different methods can decompose this task, denoted as v_{m_1}, v_{m_2}, and v_{m_3}. The first method v_{m_1} is applicable when the vehicle does not have enough power to travel to the next location. To recharge, the vehicle must drive to a charging station (`drive`), recharge (`recharge`), and from there to its original destination (`drive`). The second method v_{m_2} is applicable when the vehicle is charged and has not reached its destination. In this case, the vehicle's engine should be cranked if it was not before (`start`), the lights are turned on if they were off and it is nighttime (`turnon`), and the vehicle moves one step to the next location (`move_step`). Then, the `drive` task is recursively decomposed again to move the vehicle to the next intermediate location until reaching the destination. The third method v_{m_3} becomes applicable if the vehicle is at the destination. This method decomposes the `drive` task to a single compound task `stop_vehilce` to stop the vehicle, which in turn is decomposed by two methods v_{m_4} and v_{m_5}. The former decomposes the task into a single primitive task to stop the engine. The latter is applicable when the engine is already off, and it decomposes the `stop_vehicle` into an empty task network, symbolised by the `nop` primitive task. These methods

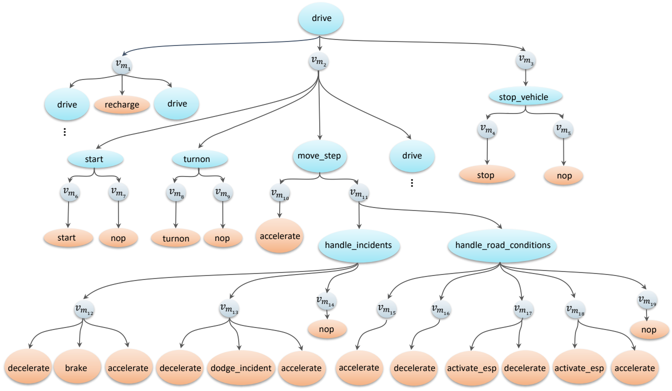


Figure 3. HTN model for the domain of Autonomous Vehicles. Blue nodes represent compound tasks, orange nodes represent primitive tasks, and grey nodes $v_{m_1}, v_{m_2}, \ldots, v_{m_{19}}$ represent methods.

implement a form of phantomisation that can be encountered in HTNs [43].

The `move_step` compound task is decomposed by two different methods v_m_{10} and v_m_{11}, based on whether the road is free from any complexity or not, respectively. In the first case, the `move_step` task is decomposed by $v_{m_{10}}$ into a single primitive task `accelerate`. In the second case, the task is decomposed by $v_{m_{11}}$ into two consecutive compound tasks `handle_incidents` and `handle_road_conditions`. The `handle_incidents` task can be decomposed by two different methods $v_{m_{13}}$ and $v_{m_{12}}$, based on whether the incident is still (e.g., rocks and construction work), or moving (e.g., pedestrians crossing the street). In the first case, the vehicle must decelerate, dodge the incident, and then accelerate again. In the second case, the vehicle should decelerate, brake to allow the moving incident to cross the road, and accelerate again. The `handle_road_conditions` task can be decomposed by four different methods $v_{m_{15}}$, $v_{m_{16}}$, $v_{m_{17}}$ and $v_{m_{18}}$. All these four methods are applicable when the road is, for example, icy, slippery, under construction, or has loose gravel, and they result in accelerating without caring about the road condition, only decelerating, activating the ESP and decelerating, activating the ESP but accelerating, respectively. These methods have the same preconditions which relate to road conditions being abnormal. This makes all four methods applicable at the same time during planning, and the planning agent always has the choice between these methods. We refer to this concept as *permissive decomposition* or *non-exclusive decomposition*. Note that both the `handle_incidents` and `handle_road_conditions` have an additional method each ($v_{m_{14}}$ and $v_{m_{19}}$) that decomposes the corresponding tasks into an empty task network when there are no incidents or road conditions, respectively. These methods constitute another form of phantomisation [43]. This modelling choice allows us to handle situations where there are road conditions and incidents at the same time between two connected locations.

2) Non-determinism: Uncertainty Sources and their Randomness: Here, we formulate the knowledge related to the non-determinism of the domain and the quantities. While we do not explicitly formulate the knowledge related to the various uncertainty sources encountered in the AV domain, we formulate the direct effects they have on the cost of driving actions performed in this domain, making them variable. In this domain model, we consider the effects of three sources of uncertainty, namely (1) the speed at which the pedestrians walk the pedestrian crossing, which is considered a regular external source of uncertainty, (2) the traffic on roads, which is also considered an external regular source of uncertainty, and (3) the ability of the autonomous vehicle to stabilise on slippery roads, which is considered an internal regular source of uncertainty.

3) Quantities: Resources and Action Costs: When this variability of costs comes from regular uncertainty sources, such as traffic jams, it can be described by a probability distribution that can be either known or statistically inferred (see Section II-B). Actions here are risk-inducing. In the present treatment, we define driving costs as the time needed to drive through the roads and deal with the various road complexities. That is, the existence of uncertain traffic jams during planning makes the estimation of travelling times variable (as shown in Figure 2). While we use risk-inducing actions and travelling times as costs to exemplify a possible formulation of the domain knowledge, uncertainty-inducing actions and other types of costs, such as fuel/power consumption, comfort of the ride, and road windingness, could be used.

Since in the AVs domain, the travelling costs can depend on the traffic jams and on the particular road itself, e.g., its length and conditions, we use a hybrid, external and state-dependent cost function $c^{ies}(a)$ to compute the costs (see Section II-C). In particular, the estimation of uncertain traffic jams comes from an external function, and each road's length and other properties represent a state-dependent cost function that is defined with respect to the ground planning problem. These two functions can be combined into one hybrid function that computes the probability distribution of travelling times.

4) Planning Problems Instances: Knowledge about planning problems is formulated such that the initial state contains the knowledge about the static and the dynamic states of the environment. The static state includes the road network, i.e., locations and routes connecting them, the location of still and moving incidents (e.g., construction works and pedestrians), the conditions of roads (e.g., slippery roads, roads with gravel, and normal roads), and the length of the roads. The dynamic states include the location of the vehicle. The initial task network in the planning problem is to move the vehicle from one location to another. The objects are the locations that the vehicle can navigate to and the various incidents.

5) Example of a Problem Instance: Let us consider an example of a planning problem with seven different locations denoted as S, l_1, l_2, l_3, l_4, l_5, and E, depicted in Figure 4. The vehicle is initially at S and should navigate to E while handling the various road complexities. We assume that the vehicle has enough power to navigate all the roads, and it is nighttime, so the vehicle has to turn on the lights. The roads

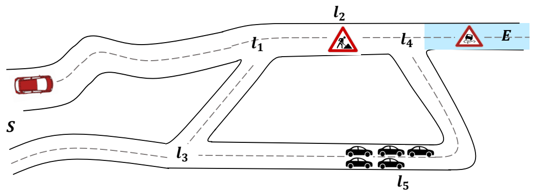


Figure 4. A problem instance in the AVs domain with seven locations S, l_1, l_2, l_3, l_4, l_5, and E, and various road complexities.

between S and l_1 and between S and l_3 are complexity-free. However, the first road is longer than the second. The road between l_1 and l_4 has constructions at location l_2. The road between l_3 and l_4 has a school area that is very crowded with pedestrians and traffic jams, i.e., moving incidents, at location l_5. The road between l_1 and l_3 is a highway free from any complexities, but has a variable level of traffic congestion. Finally, the road between l_4 and E is icy and slippery.

Figure 5 shows some possible bindings of the actions with the corresponding probability distribution of costs with respect to the ground planning problem and external traffic as sources of uncertainty. Note that we only show feasible bindings and ground actions that can be part of the computed plans. For example, the action of accelerating has a probability distribution of costs when the vehicle is accelerating on the road between S and l_1, different from the probability distribution of costs when accelerating on the road between S and l_3, since the length and traffic jams of these roads are different. The logic behind our assignments of the action variable costs is as follows. The road between S and l_1 is complexity-free and free of traffic jams. Thus, the time needed to drive through this road is certain, i.e., driving through this road takes four hours with 100% probability. On the other hand, although shorter, the road between S and l_3 has a variable traffic jam throughout the day. Driving through this road can take six hours with 20% probability, or two hours, in the best case, with 80% probability. Thus, taking the short road is riskier than taking the long road. The road between l_1 and l_4 has construction work at location l_2. Since the construction is considered a still incident, passing through this road, i.e., decelerating, dodging the constructions, and accelerating again, is done in a certain time under the assumption that this road does not include any traffic jam. In particular, cumulative deceleration on the road between l_1 and l_2 requires four hours, dodging the incident requires 0.4 hours, and accelerating on the road between l_2 and l_4 takes one hour. Unlike the road between l_1 and l_4, the road between l_3 and l_4 has multiple schools and heavy traffic at location l_5, which can lead to long waiting times. This is an external source of uncertainty since pedestrians have uncertain times and speeds at which they cross the road, which can make this area congested. Additionally, the roads between l_3 and l_5, and l_5 and l_4 have an uncertain level of traffic congestion. Thus, driving from l_3 to l_4, i.e., decelerating, braking, and accelerating, requires a variable amount of time, as shown in Figure 5. In particular, decelerating on the road between l_3 and l_5 takes one hour with 10% probability and three hours with 90% probability. Consider the school area is very crowded, and the vehicle might need to brake for a long time, waiting for the pedestrians to walk. Thus, braking before this area can take half an hour with a high probability of 90% and can, in the worst case, take three hours with a probability of 10%.

The road between l_1 and l_3 is a highway that is complexity-free. However, it has an uncertain level of traffic congestion. Thus, although the vehicle can accelerate on this road, with a small probability of 10%, it can take eight hours to reach l_3 when there is a high traffic congestion. With a high probability of 90%, the vehicle can travel from l_1 to l_3 within two hours since the highway is mostly free of traffic jams. Lastly, the road between l_4 and E is icy and slippery. If the agent chooses to decelerate without activating the ESP, the time needed to reach E will be variable and is based on the ability of the vehicle to stabilise on this slippery road. This choice can lead to six hours of driving with 20% probability and eleven hours of driving with 80% probability. If the agent chooses to decelerate after activating the ESP, it will need 10 hours, i.e., a known and certain amount of time, to reach E since this is the safest and most guaranteed option to choose. However, suppose the agent accelerates after activating the ESP. In that case, it will need an uncertain amount of time, depending on the vehicle's stability. This option is very risky since it might require 12 hours in the worst case with an 80% probability as the vehicle will probably lose stability. At the same time, there is a 20% probability that the vehicle will have good stability and reach its destination in two hours since it is accelerating. An even riskier option is to accelerate on this road without activating the ESP. In that case, the vehicle might need sixteen hours to reach location E with a probability of 90%, and, in the best case, it needs half an hour with a probability of 10%. Comparing the option of decelerating after activating the ESP with the option of decelerating without activating the ESP, the first option has a more guaranteed outcome, i.e., execution time, although both options have the same expected value, which is 10 hours. We can also see that the option of accelerating after activating the ESP is riskier than decelerating without activating the ESP, since, with an 80% probability, it might lead to a higher time than the outcome of only decelerating. Lastly, when comparing the options of decelerating without activating the ESP and only decelerating, we see that both options involve risk. However, unless the agent is highly risk-seeking, it is less likely that the first option is preferable since it has the probability of 20% of costing six hours compared to the second option, which has the probability of costing four hours less with the same 20% probability. At the same time, the first option has an 80% probability of costing eleven hours compared to twelve hours for the second option, with the same probability, which means a one-hour difference only. Note that, despite the differences in the risk level, all three options have the same expected value, which is ten hours. The only option that has a higher expected travelling time compared to all other options is accelerating without activating the ESP. That is why this option is the riskiest one and is only chosen if the agent

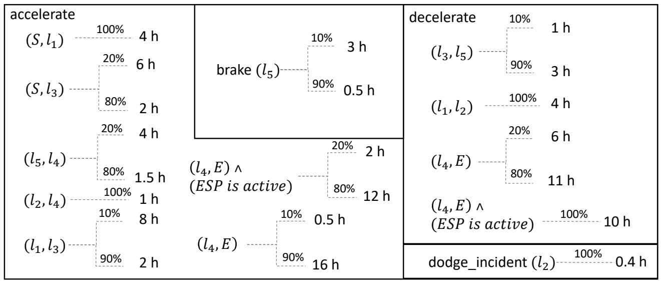


Figure 5. Actions with the corresponding possible bindings and probability distribution of costs (time).

is extremely risk-seeking. Additionally, we extend the possible roads that the vehicle can take by adding the possibility of taking a shortcut road that has a 90% probability of taking two hours and a 10% probability of taking eight hours. This extension increases the number of choices the agent should make according to its risk attitude, since taking the shortcut road can have a high risk of incurring long travelling times compared to taking several longer roads.

C. Domain Model Encoding

We encode the domain and problem instances using our extension of HDDL [14]. Compound tasks are modelled by providing the name of the task with the parameter list, as shown in Listing 2 for the `handle_incidents` compound task. Methods are modelled by providing the parameter list, the corresponding compound task, preconditions, and task network. For example, the `handle_incidents` compound task can be decomposed by three methods, where the first two are based on whether the incident is still or moving, which is ensured in the preconditions, and the third method constitutes a form of phantomisation.

Listing 2. Methods for handling incidents in the AVs domain model

```
(:method handle_incidents_0
 :parameters(?v - vehicle ?l1 ?l2 ?l3 - loc ?movinc -
    movingobs)
 :task(handle_incidents ?v ?l1 ?l2)
 :precondition(and (connected ?l1 ?l3)(connected ?l3 ?l2)(
    in-inc ?movinc ?l3))
 :ordered-subtasks(and (decelerate_incident ?v ?l1 ?l3)(
    brake ?v ?l3)(accelerate_incident ?v ?l3 ?l2)))

(:method handle_incidents_1
 :parameters(?v - vehicle ?l1 ?l2 ?l3 - loc ?stillinc -
    stillobs)
 :task(handle_incidents ?v ?l1 ?l2)
 :precondition(and(connected ?l1 ?l3)(connected ?l3 ?l2)(
    in-inc ?stillinc ?l3))
 :ordered-subtasks(and(decelerate_still_incident ?v ?l1 ?l3
    )(dodge_incident ?v ?l3 ?l2)(accelerate_still_incident
    ?v ?l3 ?l2)))

(:method handle_incidents_2
 :parameters(?v - vehicle ?l1 ?l2 - loc)
 :task(handle_incidents ?v ?l1 ?l2)
 :precondition(clearroad ?l1 ?l2)
 :ordered-subtasks())
```

Actions are modelled, as in HDDL, with a parameter list, preconditions, and effects, where risk is modelled via the construct `:costdist`. Listing 3 shows four decelerating and accelerating actions that can be performed to handle bad road conditions, each with cost distributions shown in Figure 5. Since the cost functions in this domain are hybrid, we preprocess these actions by expanding them into multiple variants based on road conditions, and directly assign the corresponding hybrid cost functions within the domain model.

Listing 3. Accelerating and decelerating actions of the AVs domain model that deal with bad road conditions.

```
(:action accelerate_bad_road
  :parameters (?v - vehicle ?l1 ?l2 - loc)
  :precondition (and (not (in ?v ?l2))(not(activated_esp ?v
     )))
  :effect (and (in ?v ?l2)(highspeed ?v))
  :costdist (or (0.9(16)) (0.1(0.5))))

(:action accelerate_after_esp
  :parameters (?v - vehicle ?l1 ?l2 - loc)
  :precondition (and (not(in ?v ?l2))(activated_esp ?v))
  :effect (and (in ?v ?l2)(highspeed ?v))
  :costdist (or (0.8(12)) (0.2(2))))

(:action decelerate_after_esp
  :parameters (?v - vehicle ?l1 ?l2 - loc)
  :precondition (and (not(in ?v ?l2))(activated_esp ?v))
  :effect (and (in ?v ?l2) (not(highspeed ?v)))
  :costdist (or (1(10))))

(:action decelerate_bad_road
  :parameters (?v - vehicle ?l1 ?l2 - loc)
  :precondition (and (not(in ?v ?l2))(not(activated_esp ?v)
     ))
  :effect (and (in ?v ?l2)(not(highspeed ?v)))
  :costdist (or (0.2(6)) (0.8(11))))
```

We model the objects existing in the planning problem, the initial task network is to drive to a destination, and the initial state is a list of predicates. Listing 4 shows the initial state of the problem instance illustrated in Figure 4.

Listing 4. Initial state of the problem instance illustrated in Figure 5.

```
(:init (connected s l1)(connected s l3)(connected l1 l4)(
     connected l3 l4)(connected l4 e)(connected l1 l2)(
     connected l2 l4)(connected l3 l5) (connected l5 l4)(
     clearroadlong s l1)(clearroad s l1)(clearroadshort s l3
     )(clearroad s l3)(in-inc construction l2)(in-inc
     bottleneck l5)(clearroad l4 e)(badroad l4 e)(in v0 s))
```

VI. Conclusions

Generating valid, capable, and executable plans for real-world scenarios requires domain models that accurately reflect the complexities of the target environments. We showed how to systematically engineer domain knowledge and model two representative domains, Satellite and Autonomous Vehicles, that embody common planning challenges in complex and dynamic environments. For the Satellite domain, we build upon existing models by explicitly incorporating elements of risk and uncertainty, and by expanding the set of methods available to achieve tasks. For the AV domain, we address a notable gap in the literature, namely, the fact that many existing works focus on traffic-level control or underwater vehicles, while few address the planning needs of individual AVs. Our approach and model fill this gap by capturing realistic driving tasks alongside key aspects such as uncertainty, risk, and the wide range of methods for tasks. Together, these contributions not only enrich the Satellite and AV domains but also provide a concrete path toward making AI planning more applicable to real-world deployment. By showing how realistic aspects can be systematically identified, incorporated, and formalised, our work lays the foundation for improving other domains and advancing planners that can operate effectively under real-world conditions.

References

[1] S. Kolski, D. Ferguson, M. Bellino, and R. Siegwart, "Autonomous driving in structured and unstructured environments", in *IV Symposium*, IEEE, 2006, pp. 558–563.

[2] D. Omeiza, H. Webb, M. Jirotka, and L. Kunze, "Explanations in autonomous driving: A survey", *T-ITS*, vol. 23, no. 8, pp. 10 142–10 162, 2021.

[3] M. Ghallab, D. Nau, and P. Traverso, *Automated Planning: theory and practice*. Elsevier, 2004.

[4] I. Georgievski and M. Aiello, "HTN planning: Overview, comparison, and beyond", *Artificial Intelligence Journal*, vol. 222, pp. 124–156, 2015.

[5] I. Georgievski, "Software Development Life Cycle for Engineering AI Planning Systems.", in *ICSOFT*, 2023, pp. 751–760.

[6] T. L. McCluskey, "Object Transition Sequences: A New Form of Abstraction for HTN Planners.", in *AIPS*, 2000, pp. 216–225.

[7] I. Georgievski, "Engineering AI Planning Systems", Habilitation Thesis, University of Stuttgart, 2025.

[8] T. L. McCluskey, T. S. Vaquero, and M. Vallati, "Engineering knowledge for automated planning: Towards a notion of quality", in *K-CAP*, 2017, pp. 1–8.

[9] E. Alnazer and I. Georgievski, "Understanding Real-World AI Planning Domains: A Conceptual Framework", in *SummerSoC*, Springer, 2023, pp. 3–23.

[10] M. Vallati and L. McCluskey, "A quality framework for automated planning knowledge models", in *ICAART*, SciTePress, 2021, pp. 635–644.

[11] E. Alnazer, I. Georgievski, and M. Aiello, "On bringing HTN domains closer to reality-the case of satellite and rover domains", in *SPARK workshop in ICAPS*, 2022.

[12] I. Georgievski, "Conceptualising software development lifecycle for engineering AI planning systems", in *CAIN*, IEEE, 2023, pp. 88–89.

[13] E. Alnazer, I. Georgievski, and M. Aiello, "Risk Awareness in HTN Planning", *arXiv preprint arXiv:2204.10669*, 2022.

[14] D. Höller *et al.*, "HDDL: An extension to PDDL for expressing hierarchical planning problems", in *AAAI conference on artificial intelligence*, vol. 34, 2020, pp. 9883–9891.

[15] J. R. Silva, J. M. Silva, and T. S. Vaquero, "Formal knowledge engineering for planning: pre and post-design analysis", *Knowledge Engineering Tools and Techniques for AI Planning*, pp. 47–65, 2020.

[16] I. Georgievski, "Hierarchical planning definition language", University of Groningen, Tech. Rep. JBI 2013-12-3, 2013.

[17] T. S. Vaquero, J. R. Silva, F. Tonidandel, and J. C. Beck, "itSIMPLE: towards an integrated design system for real planning applications", *KER*, vol. 28, no. 2, pp. 215–230, 2013.

[18] T. S. Vaquero, J. R. Silva, J. C. Beck, *et al.*, "Improving planning performance through post-design analysis", in *KEPS*, 2010, pp. 45–52.

[19] D. Pellier and H. Fiorino, "From classical to hierarchical: Benchmarks for the HTN track of the international planning competition", *arXiv preprint arXiv:2103.05481*, 2021.

[22] F. Jimoh, L. Chrpa, T. L. McCluskey, and S. Shah, "Towards application of automated planning in urban traffic control", in *ITSC 2013*, IEEE, 2013, pp. 985–990.

[20] F. D. S. Cividanes, M. G. V. Ferreira, and F. de Novaes Kucinskis, "An Extended HTN Language for Onboard Planning and Acting Applied to a Goal-Based Autonomous Satellite", *AESS*, vol. 36, no. 8, pp. 32–50, 2021.

[21] M. D. Rodríguez-Moreno, D. Borrajo, and D. Meziat, "An AI planning-based tool for scheduling satellite nominal operations", *AI Magazine*, vol. 25, no. 4, pp. 9–9, 2004.

[23] M. Vallati, D. Magazzeni, B. De Schutter, L. Chrpa, and T. McCluskey, "Efficient macroscopic urban traffic models for reducing congestion: A PDDL+ planning approach", in *AAAI conference on artificial intelligence*, vol. 30, 2016.

[24] M. Gulić, R. Olivares, and D. Borrajo, "Using automated planning for traffic signals control", *PROMET-Traffic&Transportation*, vol. 28, no. 4, pp. 383–391, 2016.

[25] T. McCluskey and M. Vallati, "Embedding automated planning within urban traffic management operations", in *ICAPS*, vol. 27, 2017, pp. 391–399.

[26] F. Ivankovic, M. Roveri, *et al.*, "Planning with Global State Constraints for Urban Traffic Control", in *CEUR-WS.org*, CEUR-WS, vol. 2987, 2021, pp. 1–5.

[27] T. X. Lin, M. Hou, C. R. Edwards, M. Cox, and F. Zhang, "Bounded Cost HTN Planning for Marine Autonomy", in *Global Oceans 2020: Singapore–US Gulf Coast*, IEEE, 2020, pp. 1–6.

[28] E. Turan, S. Speretta, and E. Gill, "Autonomous navigation for deep space small satellites: Scientific and technological advances", *Acta Astronautica*, vol. 193, pp. 56–74, 2022.

[29] D. Long and M. Fox, "The 3rd international planning competition: Results and analysis", *JAIR*, vol. 20, pp. 1–59, 2003.

[30] C. Powell, C. S. Ruf, S. Gleason, and S. C. Rafkin, "Sampled Together: Assessing the Value of Simultaneous Collocated Measurements for Optimal Satellite Configurations", *BAMS*, vol. 105, no. 1, E285–E296, 2024.

[31] V. Maggioni, C. Massari, and C. Kidd, "Errors and uncertainties associated with quasiglobal satellite precipitation products", in *Precipitation science*, Elsevier, 2022, pp. 377–390.

[32] G. A. Landis, S. G. Bailey, and R. Tischler, "Causes of power-related satellite failures", in *WCPEC*, IEEE, vol. 2, 2006, pp. 1943–1945.

[33] P. S. Morgan, "Fault protection techniques in JPL Spacecraft", 2005.

[34] H.-S. Choi *et al.*, "Analysis of GEO spacecraft anomalies: Space weather relationships", *Space weather*, vol. 9, no. 6, 2011.

[35] R. Horne *et al.*, "Space weather impacts on satellites and forecasting the Earth's electron radiation belts with SPACECAST", *Space Weather*, vol. 11, no. 4, pp. 169–186, 2013.

[36] C. C. Liu, S. G. Hosking, and M. G. Lenné, "Predicting driver drowsiness using vehicle measures: Recent insights and future challenges", *Journal of safety research*, vol. 40, no. 4, pp. 239–245, 2009.

[37] N. R. Council *et al.*, *Completing the forecast: Characterizing and communicating uncertainty for better decisions using weather and climate forecasts*. National Academies Press, 2006.

[38] *Enhancing Weather Information with Probability Forecasts*, https://shorturl.at/pvG03, Accessed: 2022-03-18, 2022.

[39] J. Antoun, M. E. Kabir, R. F. Atallah, and C. Assi, "A data driven performance analysis approach for enhancing the QoS of public charging stations", *T-ITS*, vol. 23, no. 8, pp. 11 116–11 125, 2021.

[40] *Google Maps directions for driving from Erlangen to Stuttgart*, https://shorturl.at/cosW7, Accessed: 2024-03-15, n.d.

[41] *Google Maps directions for driving from from Erlangen to Stuttgart*, https://shorturl.at/cKVWX, Accessed: 2024-03-15, n.d.

[42] P. Farradyne, "Traffic incident management handbook", *Prepared for Federal Highway Administration, Office of Travel Management*, 2000.

[43] I. Georgievski and M. Aiello, "Phantomisation in state-based HTN planning", in *ASPAI*, 2019, pp. 39–44.