

A Hybrid Method of Assurance Cases and Testing for Improved Confidence in Autonomous Space Systems

Ben Smith¹, Martin S. Feather² and Terry Huntsberger³
Jet Propulsion Laboratory, California Institute of Technology, Pasadena, CA, 91109

Autonomous systems react intelligently to their environments, making them capable of handling many possible conditions, but challenging to test. We are investigating a new test development method that aims to maximize the confidence to be achieved by combining Assurance Cases with High Throughput Testing (HTT). Assurance Cases, developed for safety-critical systems, are a rigorous argument that the system satisfies a property (e.g., the Mars rover will not tip over during a traverse). They integrate testing, analysis, and environmental and operational assumptions, from which the set of conditions that testing must cover is determined. In our method, information from the Assurance Case is used to determine the test coverage needed, and then input to HTT to generate the minimal test suites needed to provide that coverage.

I. Introduction

Autonomous systems are increasingly important for space exploration, but are challenging to test. Autonomy enables spacecraft to investigate environments and phenomena that would otherwise be out of reach. Spacecraft and rovers operate at the far end of long light-time delays with limited communications bandwidth. By making decisions onboard, instead of through operators on Earth, they can operate in dynamic and uncertain environments. Autonomy enables rovers to drive through uncertain terrain while avoiding hazards¹; investigate transient phenomena like dust devils²; select promising science targets³; and land on distant planets (e.g., Mars⁴). However, before autonomy software is given control we must be confident that it will do what we expect and will not endanger the spacecraft or mission objectives.

Autonomous systems are challenging to test because they must make the correct decision in a wide range of situations. The space of situations is enormous, so exhaustive testing is impractical, and even an ambitious test program will cover only a fraction of the situations. So how do you use limited test resources effectively? In addition, once you are done, how confident are you in the system?

The traditional testing approach is a combination of simulation, sub-assembly testing, field-testing, and operational constraints. The expected operational envelope limits the space of situations and responses that has to be considered. Operational procedures then limit the autonomy capabilities to that envelope. Simulation tests cover as much of that input space as possible, and are backed by sub-assembly and field tests of increasing fidelity. Confidence is informed by typical factors such as coverage and defect rates. Confidence is also obtained operationally by starting with the minimum level of autonomy and raising it over time (e.g., as was done for validation of the autonomous science target selection system mentioned above³).

In this paper, we introduce a new assurance methodology for autonomous systems, which makes more effective use of test resources and provides higher confidence. The method is based on a combination of *Assurance Cases* and *High Throughput Testing (HTT)*. An Assurance Case is a rigorous, well-structured argument for why a system meets critical requirements. A compelling argument can increase confidence by providing a sound, complete, and reviewable case. In our methodology, an Assurance Case is coupled with High Throughput Testing to automatically generate *efficient*, high-coverage test suites. Parameters and values that describe the input space are derived from the argument structure. An HTT algorithm then generates the test suite.

The remainder of this paper describes the approach in more detail, and our ongoing work to implement and evaluate it in the context of a pilot study.

¹ Senior, Planning and Sequencing Systems, JPL, 4800 Oak Grove Drive, Pasadena, CA 91109.

² Principal, Software Assurance and Assurance Research, JPL, 4800 Oak Grove Drive, Pasadena, CA 91109.

³ Principal, Autonomous Systems Division, JPL, 4800 Oak Grove Drive, Pasadena, CA 91109.

II. Approach

Our work is motivated by the desire to make more use of autonomy in space systems, provided its use can be confidently assured. Our approach combines Assurance Cases with High Throughput Testing to improve confidence and optimize effectiveness of scarce test resources. We are evaluating our approach in the context of the autonomous traverse subsystem of the upcoming Mars 2020 rover mission, the next rover in NASA’s Mars Exploration Program. It should be emphasized that this is a pilot study to evaluate the effectiveness of this approach, and not part of the M2020 validation effort *per se*. As described in Ref. 5, “*For Mars 2020 it is anticipated that much of the traverse distance between ROIs will be done autonomously to maximize the time the science team has to explore within the ROI.*” – “ROI” stands for Region Of Interest, where detailed science activities are to be conducted. The two aspects of M2020’s autonomous traverse we are studying are the mission requirements to achieve a given traverse rate (expressed in terms of kilometers driven over a number of sols [Martian days]), and, crucially, to do so safely (i.e., not fail catastrophically).

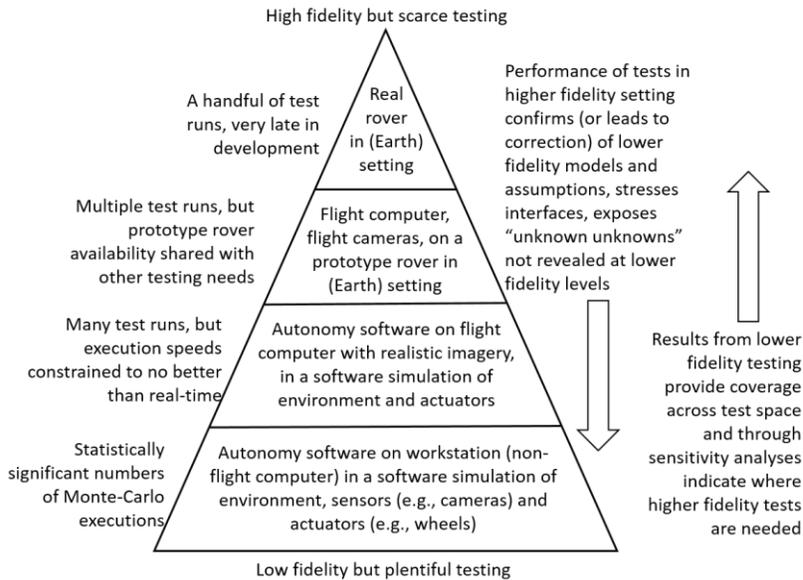


Figure 1. A (simplified) hierarchy of test levels.

One of the approaches to achieving confidence in autonomy capabilities, as followed for NASA’s previous Mars rovers (MSL’s Curiosity rover, the two MER rovers Spirit and Opportunity, and before them Pathfinder’s rover Sojourner), is a combination of simulation, sub-assembly testing, field testing, and operational constraints. Fig. 1 illustrates (a simplification of) how a hierarchy of levels of test fidelity is used for this. Factors that constrain space systems to this kind of approach include the one of a kind nature of NASA’s space missions. High fidelity artifacts are often not available until late in the development cycle, so early testing necessarily focuses on lower fidelity engineering testbeds. A further

complication is of course the inability to recreate on Earth, or even know, the environment in which the missions will execute, which drive testing to cover a wide range of possibilities “just in case”.

The approach we are following seeks to maximize the confidence to be achieved by the finite amount of testing possible given the above constraints. To do so, we combine two existing approaches – Assurance Cases (described in Section II.A), and High Throughput Testing (described in Section II.B). The nature of our combination is covered in Section II.C.

A. Assurance Cases

We repeat definition of Assurance Case found in Ref. 6: *An assurance case is an organized argument that a system is acceptable for its intended use with respect to specified concerns (such as safety, security, correctness).* As further explained there, “*Assurance Case*” is a relatively new term that fully encompasses the older term “*Safety case*” ... to place other concerns such as security and dependability alongside concerns for safety.

We picked Assurance Cases as the means to develop, organize and present the arguments that an autonomous system satisfies critical properties. For example, a critical property in our domain of interest is “the rover will not over during an autonomous traverse.” To do so could be mission ending – the kinds of rovers deployed on Mars have no ability to right themselves were this to occur. Considerable time and effort (many reviews, inspections, tests and analyses) is expended to assure that space missions will not suffer such catastrophic failures, given their high cost. When missions seek to benefit from increasing amounts of autonomy, they must thoroughly address the assurance of that autonomy to have the confidence to allow its control of mission-critical activities.

For complex software systems where high dependability is needed, use of Assurance Cases was one of the recommendations of a National Research Council study⁷ (note – that document uses the equivalent term “Dependability Case”). Assurance Cases have been featured in space systems studies and applications, e.g.:

- a retrospectively constructed Dependability Case⁸ of the User Spacecraft Clock Calibration System as developed by NASA Goddard
- a launch abort detection system⁹
- the software involved in spacecraft “safe mode”¹⁰
- assurance arguments for unmanned aircraft systems¹¹

Since autonomy employs software algorithms and implementations atypical as compared to traditional control software, there is natural concern that existing software V&V practices are not necessarily well suited to provide sufficient assurance. Hence, our particular interest in Assurance Cases, because their foundation on a rigorous argument encourages a thorough consideration from first principles of organizing evidence (e.g., test results) to give confidence in the critical property in question. We are encouraged by a recent workshop¹² on assurance for autonomous systems for the closely related field of aviation, in which several of the participants called out assurance cases as a promising approach. For our studies, we adopted the widely used Goal Structuring Notation (GSN)¹³ as the graphical format for assurance cases.

In our use of Assurance Cases, we are particularly interested in employing Assurance Case *patterns*¹⁴ – common structures in Assurance Case arguments. Patterns capture typical and well thought out portions of argument structures. We seek to use them to reap the advantages of not having to reinvent those argument structures ourselves (or worse, making avoidable mistakes when constructing our own argument portions) and achieving regularity so that readers (especially reviewers) of assurance arguments will know what to look for and be able to rapidly focus their attention on the system-specific aspects of the arguments.

B. High Throughput Testing (HTT)

High Throughput Testing (sometimes also referred to as combinatorial testing) is a method for reducing the total number of test cases while maintaining a predetermined coverage level of the factors or variables (e.g., wheel slippage, density of hazards, etc.) that characterize the conditions to which the system under test (SUT)¹⁵ is subjected. Previous studies have indicated that interactions between factors produce failures that would not be caught using a single factor test analysis¹⁶. HTT methods attempt to *efficiently* provide covering arrays¹⁷ for multi-way interactions between factors. For example, a system with 10 binary factors would require 1,024 test cases for coverage of *all* combinations of the factors, but only 13 tests would suffice to cover all 3-way combinations of the factors. A related alternate approach for the reduction in the total number of test cases is DOE (Design Of Experiments), which provides orthogonal estimates of each factor for the purposes of quantitative analysis such as least squares regression¹⁸. A limitation of the DOE method is that almost all designs assume two to three levels (discretizing of a continuous factor, e.g., angle of slope) for each factor (rarely the case), which leads to a need for all combinations to be tested if this assumption is violated.

The covering array produced using HTT for software analysis differs from that used in our system. For assurance testing of autonomous systems, there is typically a need to provide situational coverage¹⁹ of the operations and environment (see Fig. 2) that the SUT will encounter. Many of the environmental context derived factors will be continuous, and the number bins for discretizing the range of values becomes important for the effectiveness of the HTT situational coverage. There may be values within the range where an abrupt change in SUT performance will occur and may be missed in the situational covering array produced by a HTT analysis. For example, there will be an angle of slope combined with certain terrain characteristics that will suddenly cause a SUT to start to slip. These types of performance boundaries can be found through Gaussian process regression (GPR) techniques such as those used in the Range Adversarial Planning Tool (RAPT) system²⁰.

C. Hybrid Method: Assurance Cases combined with HTT

In addition to their individual contributions, we believe Assurance Cases and High Throughput Testing can be used synergistically to provide further value to assurance.

Our vision for the overall flow of our approach is depicted in Fig. 2. To date we have performed its steps manually in order to assess feasibility.

Assurance Case Authoring: The information from which we construct the Assurance Case comes from a variety of sources: knowledge of the way the mission, its autonomy in particular, is to be operated; knowledge of the mission’s environment; traditional requirements levied on the design; test planned for those requirements (typically expressed in a test matrix); other information, e.g., hazard analyses, review materials. We build the Assurance Case using an authoring tool (currently we are using the AdvocATE toolkit²¹ for this purpose)

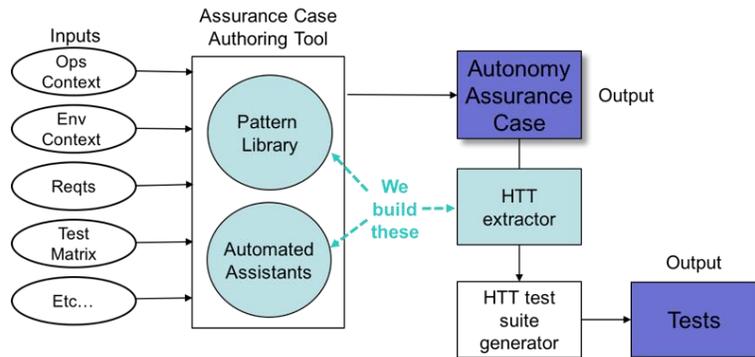


Figure 2. Our vision for combining an Assurance Case and High Throughput Testing.

making use of automated assistants (e.g., to translate requirement hierarchies into the backbone of an assurance case) and patterns from a library of such that we have found useful for autonomy assurance.

Test Determination: The key step of our combined method is to extract from the Assurance Case the test conditions needed to have confidence in the autonomous system, and feed those conditions into the HTT test suite generator to yield an effective and efficient set of tests.

The hierarchical structure of the Assurance Case leads down to its “leaf” evidence nodes (typically tests or analyses) supporting the claims of the lowest-level goals. Groups of evidence nodes indicate the conditions that tests must address. The argument structure also tells us about the parameter space that those tests need to cover. Goals in the argument provide context about the environment and artifact that suggest parameters. We can extract these parameters from the Assurance Case and generate covering test suites with HTT. This provides an explicit, documented link between the Assurance Case and the test suites, which improves confidence and test efficiency. If the Assurance Case changes during development, it is easier to update the parameters and re-generate the test suite, thereby reducing retest costs.

D. Pilot study

We are engaged in a pilot study to, it is hoped, show the benefits of Assurance Cases, HTT, and the hybrid of these. As mentioned above, our pilot study is focused on the autonomous traverse subsystem of the upcoming Mars 2020 rover mission. Using this, we have the objectives to show that:

- Sufficiency and practicality of Assurance Cases for autonomy
 - Assurance Cases are capable of expressing cogent arguments for assurance of autonomy
 - Existing guidance and tooling for Assurance Case construction are sufficient
 - Assurance Case “patterns” (existing ones, and new ones as might be discovered in the course of our study) are applicable to ease the construction and comprehension of Assurance Cases
- Assurance Cases can guide the generation of efficient testing
 - Test conditions can be extracted from Assurance Cases to serve as input to HTT
 - HTT test suites generated as a result are efficient – efficiency is particularly important when test resources are scarce, as is so for the higher fidelity tests of Fig. 1.

III. Results

A. Assurance Cases – Overall results

We developed (portions of) two Assurance Cases for the autonomous traverse subsystem of the upcoming Mars 2020 rover mission. One addresses the mission requirements to achieve a given traverse rate (expressed in terms of kilometers driven over a number of sols [Martian days]); the other, the requirement to do so safely (i.e., not fail catastrophically). Our sources of information from which to construct these cases was a combination of: the project’s existing pre-decisional material on the autonomy design, technology, and plans for testing; insights from discussions with the autonomy lead; and past experiences of our third author with earlier Mars rovers traverse capabilities.

We found Assurance Cases capable of expressing the crux of the arguments for, respectively, traverse *efficacy* and traverse *safety*.

- For traverse efficacy, our Assurance Case traced the top-to-bottom requirements hierarchy that specifies how the rover will achieve its autonomous traverse objectives. We made repeated use of the *requirements breakdown pattern*²², in which a parent requirement is subdivided into several child requirements (typically allocated to sub-components of the component to which the parent requirement was allocated). This applies to match the hierarchical design of the rover (system, subsystem, component, and a similar hierarchical structure within the software architecture). We were also able to develop some simple forms of automated assistance (see Fig. 2) to translate the project’s existing requirement hierarchies into the backbone of our Assurance Case.
- We describe our experience developing an Assurance Case for traverse *safety* in detail in the subsection that follows.

For evaluating whether our cases expressed cogent arguments, we have so far relied on determining whether people other than ourselves can comprehend our Assurance Cases – we found this to be so when portions of our cases were viewed both by an expert in the M2020 mobility system, and by the reviewers of our research task, who are all familiar with JPL’s approach to system V&V, and have varying levels of familiarity with rover mobility. A more systematic evaluation of the perspicuity of our Assurance Case is something we aim to perform in the near future. Ideally, we would like to be able to show that use of Assurance Case itself improves the developers’ and reviewers’ confidence in the autonomous systems it describes. In general, evaluating the benefits of Assurance/Safety Cases has long been a challenge (e.g., Ref. 23) and remains so to this day²⁴. Part of this is due to the difficulty in measuring confidence conveyed by an Assurance Case²⁵, so far we can progress in this direction remains to be seen.

We adopted the widely used Goal Structuring Notation (GSN)¹³ as the graphical format to represent both our Assurance Cases. For creating and editing we initially made use of the Astah software (<http://astah.net.editions.gsn>), and subsequently the AdvoCATE toolkit²¹. The assurance case diagrams shown in this paper were created using the latter; see the Appendix for a very brief guide to the notations used herein.

For guidance, we found useful Kelly’s “six-step method for the development of goal structures”²⁶ and Spriggs’ book²⁷. We were also keen to make use of safety case “patterns” as advocated in Ref. 14. Further motivation for using patterns came from the experience reported in Ref 28, recounting a successful experiment in which patterns proved effective when used by a domain expert after relatively modest amounts of training in assurance cases. If we are to be successful in introducing assurance cases into mainstream project use, availability of guidance, and ease of adoption, are critical considerations.

B. Details of our Assurance Case for traverse safety

We now give some details of our second Assurance Case, the one that presents the argument for the safety of the rover’s traverse.

Our traverse safety case focused on the mission-critical aspects of autonomous driving (e.g., that it will not go the wrong way and tip over or drive off a cliff). It follows the style of safety and assurance cases reported in the literature. Obviously there is no threat to human safety when operating a rover on Mars, but the cost of these missions warrants high assurance that they will not fail prematurely (i.e., before they have completed their science objectives). We picked a thread through this that leads to the critical role played by the Visual Odometry (VO) component. VO is the rover’s means of determining where it is – essential if it is to stay on route and avoid hazards that have been pinpointed on a map of the terrain (e.g., as identified using high resolution imagery from orbital spacecraft). Conveniently, we had the original VO test plans from MER and MSL (already deployed Mars rovers) to serve as a point of comparison as regard test suite efficiency.

In the course of developing this Assurance Case we made repeated use of standard Assurance Case patterns, including development of what we think is a new one. Indeed, most of the development of our Case is achieved by the application of standard patterns instantiated on our system specifics, refining the Case to the level at which evidence (testing, simulation, analyses, etc.) applies to provide evidence in support of the leaf goals. This is seen from the very start, as illustrated by the top of our case shown in Fig. 3, where we take care to spell out the scope of risks we are concerned with. The bifurcation into “terrain-interaction” risks (e.g., collision with a large rock, driving over a cliff) and “non-terrain interaction” risks (e.g., execution of the autonomous driving software corrupting some other mission critical software) allows the arguments that follow to be more readily assessed as complete, and also to match the forms of evidence applicable. For example, “terrain interaction” risks, as its name suggests lead to consideration of all the physical hazards to driving potentially present in the driving environment, while “non-terrain-interaction” risks are those more standard risks that arise while executing any kind of software (e.g., that it

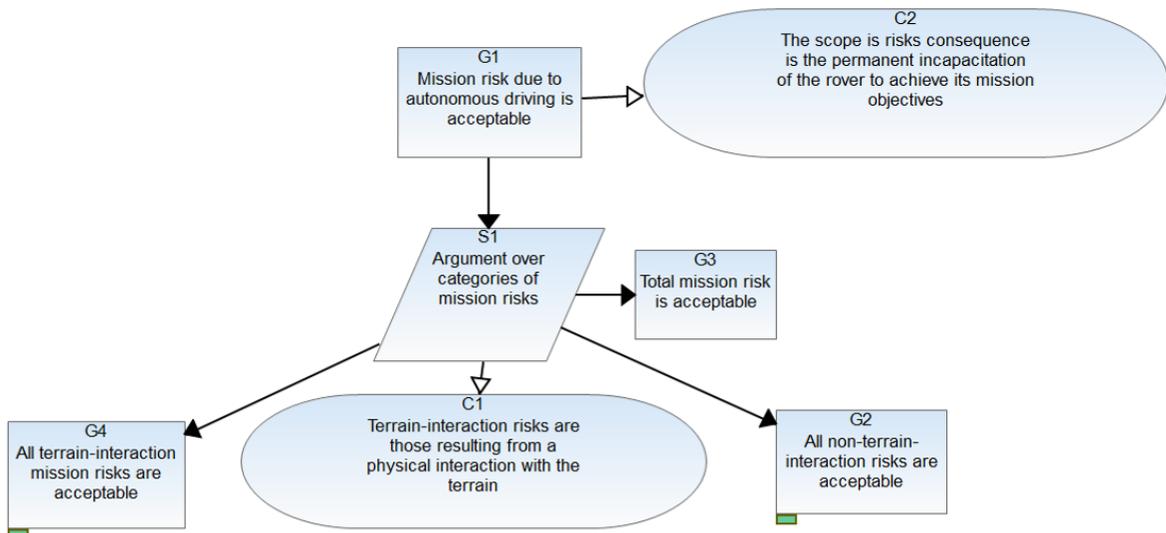


Figure 3. Top of assurance case.

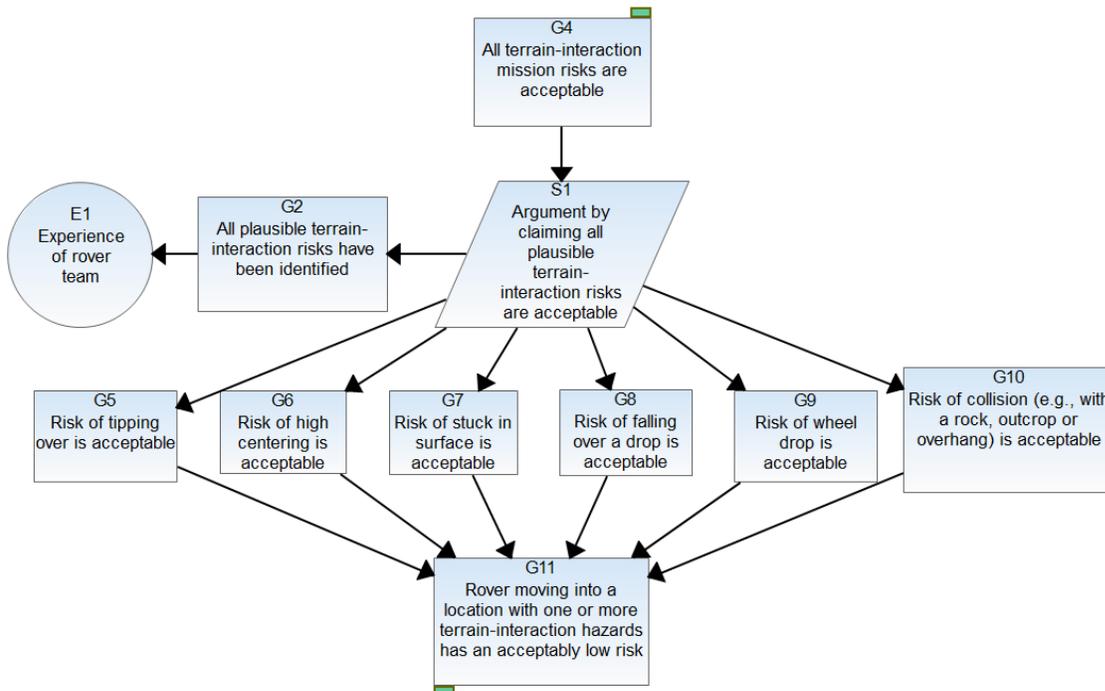


Figure 4. Illustration of structuring an argument over risks.

interferes with or corrupts some other software). The former category’s risks are the focus of our attention and lead to consideration of how to test in various settings, etc., while the latter category’s risks are amenable to more traditional forms of software assurance (e.g., static analysis of code to show well-behaved memory accesses).

Fig. 4 illustrates structuring the argument’s expansion of “All terrain-interaction mission risks are acceptable”, the result of a standard “argument over risks” pattern modeled after the “Hazard Avoidance Pattern”¹⁴ (in our setting we prefer to use the phrasing “Risk ... is acceptable”). The risks listed are for purposes of illustration.

Note our inclusion of the evidence node “Experience of rover team” – we believe this to be a useful addition to the standard pattern, since in our setting people might question whether we can be confident in the completeness of risk identification for operations on a distant planet.

We also encountered some instances of “concerns” that the project had yet to determine were plausible risks requiring mitigation. At this stage in the development they have a plan to further assess this risk, and, if it turns out to be considered plausible, then to include it in the same avoidance-based approach used to address the other risks.

We extended our assurance case to cover the plan. In this respect we are following the approach set forth in Section 5.3 of Ref. 29 where it states:

... the concept of adequate safety requires that safety is addressed throughout all phases of the system life cycle. This translates into the development of safety objectives that span the full life of the system, from concept studies to closeout. Correspondingly, the RISC must also address the full system life cycle, regardless of the particular point in the life cycle at which the RISC is developed. This manifests in the RISC as two distinct types of safety claims:

- Claims related to the safety objectives of the current or previous phases argue that the objectives have been met.
- Claims related to the safety objectives of future phases argue that a ‘roadmap’ has been established for the satisfaction of objectives yet to be met, i.e., that necessary planning and preparation have been conducted and that commitments are in place to meet the objectives at the appropriate time.

The next step of our argument that we show here, Fig. 5, is the breakdown of the goal “Rover moving into location with one or more terrain-interaction hazards has an acceptably low risk”. Here we apply a “safety monitoring” pattern, which combines an activity *intended* to be free of risk with a separate monitor checking for a risk about to materialize, and halting the activity in time to avoid that risk. Safety monitoring systems are commonly employed, and are covered in published literature on safety and assurance cases, e.g., Refs. 30 and 31.

Further down in our assurance case we reach the part addressing a classical autonomy sense-decide-act portion of the rover design, in this case the sensing of terrain hazards, the decision of a route to follow that avoids the identified hazards, and the action of driving in a manner

that remains within the planned route. The part of this concerning sensing led us to the rover’s means of determining where it is – essential if it is to stay on route and avoid hazards that have been pinpointed on a map of the terrain (e.g., as identified using high resolution imagery from orbital spacecraft). There is no equivalent of GPS on Mars; a rover’s position can be determined by ground control’s scrutiny of high resolution imagery of its surrounding terrain. For autonomous operation without frequent such involvement of ground control, the rover maintains knowledge of its location using Visual Odometry (VO). From Ref. 32’s description of VO used on the MER rovers:

Our Visual Odometry system computes an update to the 6-DOF rover pose (x, y, z, roll, pitch, yaw) by tracking the motion of “interesting” terrain features between two pairs of stereo images in both 2D pixel coordinates and 3D world coordinates. A maximum likelihood estimator applied to the computed 3D offsets produces the final motion estimate. However, if any internal consistency check fails, too few feature points are tracked, or the estimation fails to converge, then no motion estimate update will be produced and the initial estimate (nominally based on wheel odometry and the IMU) will be maintained.

This proved interesting from an assurance case perspective, in several respects:

- It led us to introduce a (new?) pattern: “Confident Self Assessment” to represent the consistency check mentioned in the above.
- It is at the level where the testing needed to provide the evidence of the adequacy of the consistency check is determined – and hence is the point at which we connect our assurance case to High Throughput Testing

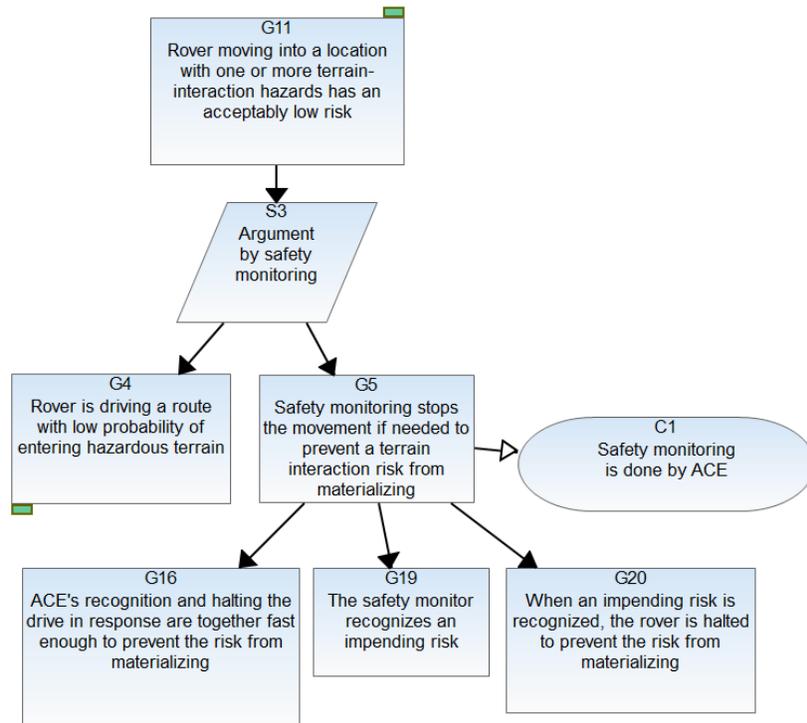


Figure 5. Result of applying a “safety monitoring” pattern.

- Testing of VO would be done in several venues – some in a purely computer simulation with simulated imagery, with actual images taken by flight-like cameras (but still in a simulation), in operation on a physical rover platform in Mars-like terrain (e.g., the JPL “Mars yard” or desert areas in and around Southern California that mimic Martian terrain)

In the current rover design, lack of a result from VO causes a halt to the drive rather than continue, thus preserving safety (but at the expense of progress). Another, similar, example of self assessment was exhibited by the MER rovers during their spacecrafts’ Entry Descent and Landing³³. They used the Descent Image Motion Estimation System (DIMES) for horizontal velocity estimation during descent, by feature tracking between subsequent images of the ground. DIMES was to report a velocity estimate only when it had a high probability of being correct, and used various consistency checks to confirm this. We thought it worthwhile to make a simple pattern “Confident Self Assessment” and applied it at this point in our assurance case.

Consideration of testing arises naturally at this stage, as the means to show that as a result of self assessment, VO will either return a correct result, or not return anything. To know whether testing is sufficient hinges on a hazard analysis of how VO could potentially incorrectly calculate a location update. There are four

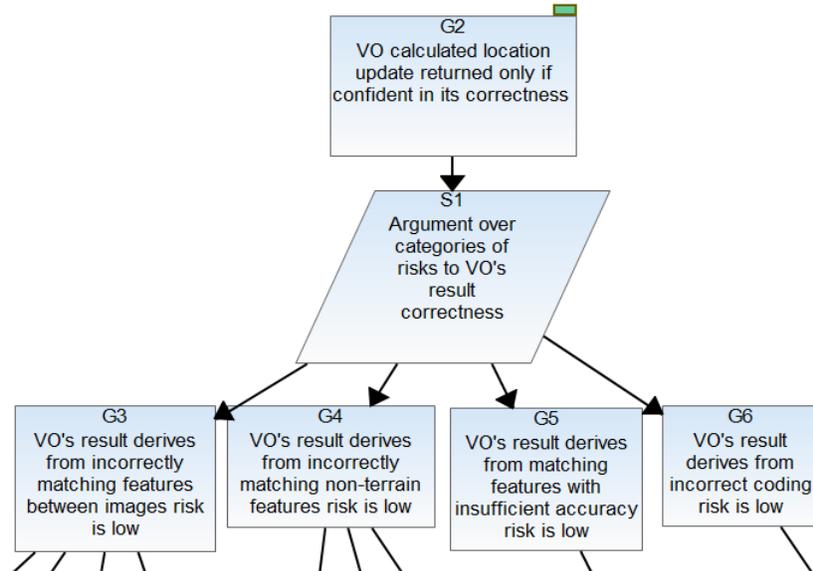


Figure 6. Four categories of risks that could lead to VO returning an incorrect result.

broad cases to consider, as seen in this portion of our assurance case shown in Fig. 6:

- VO’s result derives from incorrectly matching features between images, possible in the conditions, or combinations of conditions, of:
 - repetitive terrain (visually identical features at different locations)
 - featureless terrain (texture with too few features to use for matching)
 - challenging lighting conditions (e.g., low light, glare, or directly overhead casting few shadows)
 - degraded optics
 - little or no overlap between images (e.g., because the rover slipped, skidded, rotated a lot, or traversed a long distance in its move)
- VO’s result derives from incorrectly matching *non*-terrain features, possible in conditions of:
 - the rover’s own shadow, or portions of the rover itself, in the images (the latter excluded by design and operation)
 - dust on the lens
 - failed pixels in the camera
- VO’s result is from matched features that yield insufficiently accurate location update, possible in the conditions of:
 - matched features are too distant, offering little parallax from which to compute location update accurately
 - insufficient camera resolution
- VO’s implementation is incorrectly coded

The first three categories relate to the need to provide situational coverage¹⁹ of the operations and environment that the SUT will encounter, for which testing in high fidelity settings is preferable. The final category, of incorrect implementation (in software and/or firmware) is more suited to assurance via analysis (e.g., static code analysis) and

testing in simulated settings where large numbers of tests can be run, exercising the code paths through many scenarios. We focus our attention on the first three categories and how we extract from them the information to feed into HTT for generation of efficient test suites that will be used in high fidelity testing.

Should testing reveal that VO is prone to returning an incorrect result under some circumstances, the project could respond by:

- Operationally avoiding those circumstances (e.g., by ground control recognizing repetitively rippled terrain from orbit imagery and defining it as a keep-out zone). The policy to do this would be recorded in the Assurance Case as an assumption. Traversing the entire case to gather together all such assumptions would be a means to extract the full set of operational procedures the Case depends on.
- Extending VO’s self assessment to recognize those circumstances and inhibit returning a result (e.g., when the angle of the sun would project the rover’s shadow into the scene), and re-testing.
- Deciding the likelihood of those circumstances is so low as to be of little concern.
- Deciding the possibility of a terrain-interaction risk is low even if VO returns an incorrect result (e.g., featureless terrain is devoid of large rocks, so VO returning an incorrect result would not threaten the rover). The Assurance Case helps make this determination by providing the context – the chain of argument leading to this risk – to the decision makers.

C. HTT extraction and HTT results

The Assurance Case has reached the level at which the factors and their values that potentially pose a hazard to VO have been identified. Extracting them from the bulleted list in the previous section yields Table 1. To provide

	VO hazard	Factor	Value(s)
Incorrectly matching features	repetitive terrain	Terrain_Texture	Rippled
	featureless terrain	Terrain_Texture	Smooth
	challenging lighting conditions	Lighting	Low, High, Vertical
	degraded optics	Optics	Degraded
	little or no image overlap	Ground_Interaction	Low_Slip, High_Slip, Low_Skid, High_Skid
Matching non-terrain features	rover's own shadow	Lighting	Self_Shadowed
	dust on lens	Optics	Degraded
	failed pixels in camera	Optics	Degraded
Insufficient accuracy	matched features distant	Distant_Features	Present
	insufficient camera resolution	Optics	Degraded

Table 1. Assurance Case identified VO hazards, and their factors and values.

the input to HTT we also add in the nominal values for factors. The resulting combination of factors and values, where all the values for a given factor are unioned to form a set of values, is shown in Table 2. Currently we do this extraction manually; future work is to introduce automation into this step to form the “HTT extractor” box in Fig. 2.

Given this information, HTT is then applied to generate efficient test suites providing desired levels of coverage of the combinations of nominal and hazardous values. In this example there are $2 \times 5 \times 5 \times 4 \times 2 \times 4 = 1,600$ combinations of all the six factors’ values. Numerical summaries of the HTT analysis results are shown in Fig. 7 (HTT also indicates what the actual tests would be, of course).

The high fidelity testing of VO is performed on the VO algorithm after the algorithm has been tuned (i.e., its internal parameter setting optimized through field tests and operational experience). Once these parameters have been determined, VO algorithm performance is dependent on the factors identified in Table 2. HTT results show that all *three*-way combinations of factor values can be completely covered with 100 tests. By comparison, we have been told by the developers that for VO on earlier rovers, a hand-crafted test suite of approximately 160 tests was used, and covered only *two*-way combinations. This confirms that HTT can yield more efficient test suites in this example.

The coverage percentage can be used to prioritize the tests such that a parameter (e.g. Optics – nominal or degraded) can be first tested for nominal performance, then for degraded if time/funding allows.

Factor	Values (union of nominal and hazard)		#
	Nominal value(s)	VO hazard value(s)	
Distant_Features	Absent	Present	2
Ground_Interaction	Nominal	Low_Slip, High_Slip, Low_Skid, High_Skid	5
Lighting	Nominal	Low, High, Vertical, Self_Shadowed	5
Motion	Short distance, Small rotation	Long_Distance, Large_Rotation	4
Optics	Nominal	Degraded	2
Terrain_Texture	Medium, High	Smooth, Rippled	4

Table 2. Inputs for HTT.

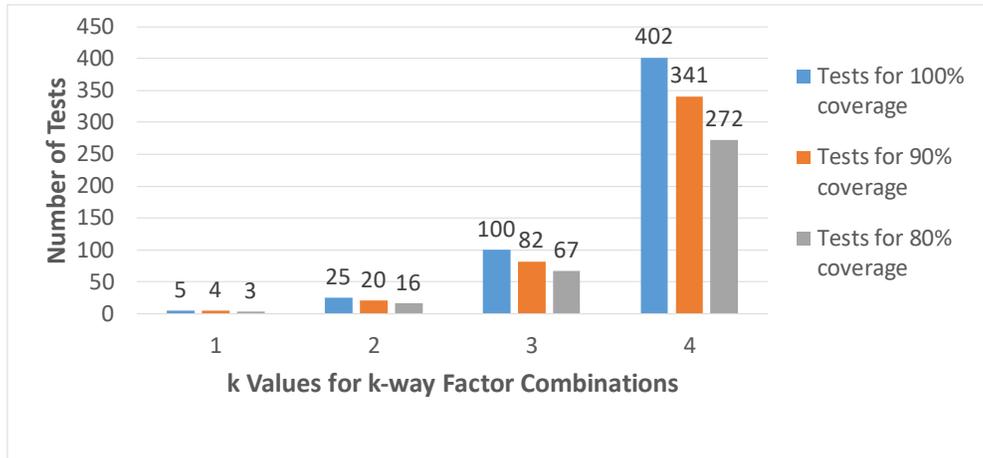


Figure 7. Numerical summaries of HTT analysis results.

IV. Conclusion

We have presented our hybrid method, combining Assurance Cases and High Throughput Testing, designed to blend the advantages of both approaches. It integrates testing, analysis, and environmental and operational assumptions, from which the set of conditions that high-fidelity testing must cover is determined. Information from the Assurance Case is used to determine the test coverage needed, and fed into HTT to generate the minimal test suites needed to provide that coverage.

Our efforts to date have focused on demonstrating this approach on some mission-critical aspects of a Mars rover. The results of this effort show support for our primary objectives, namely that Assurance Cases are both sufficient and practical as a way to present the argument for assurance of autonomy, and that in combination with HTT can guide the generation of efficient test suites, particularly for the high fidelity tests needed to cover autonomy’s operational and environmental conditions.

Our Assurance Cases covered both a “progress” goal and a “safety” goal. In developing them we found existing guidance and tooling sufficient for our purposes. Use of “patterns” proved beneficial for constructing and organizing our autonomy arguments, just as others have promoted and used them effectively in assurance cases for other kinds of systems. We also found that others not previously familiar with Assurance Cases could readily comprehend the arguments they presented.

In order to demonstrate our hybrid approach we carried the “safety” Assurance Case through to the point where environmental and operational conditions that would potentially challenge a key component of the autonomous system were identified. We then extracted the information of what factors and values for those factors would need to be covered by testing. Feeding this into HTT then yielded efficient test suites. We were encouraged to see that the result of this was HTT’s generation of a more efficient and more comprehensive test suite than the original manually generated set.

Appendix

Fig. 8 illustrates those features of the Goal Structuring Notation (GSN) used in the fragments of our assurance case appearing in this paper.

- Goal, portrayed as a rectangle, contains a statement phrased as a proposition (i.e., be either true or false).
- Strategy, portrayed as a diamond, described how a goal is decomposed into subgoals such that if each of the subgoals is true, then the goal is true.
- Evidence, portrayed as a circle, describes something that shows the goal from which it linked is true.
- Context, portrayed as a round-ended rectangle, provides additional information when needed.

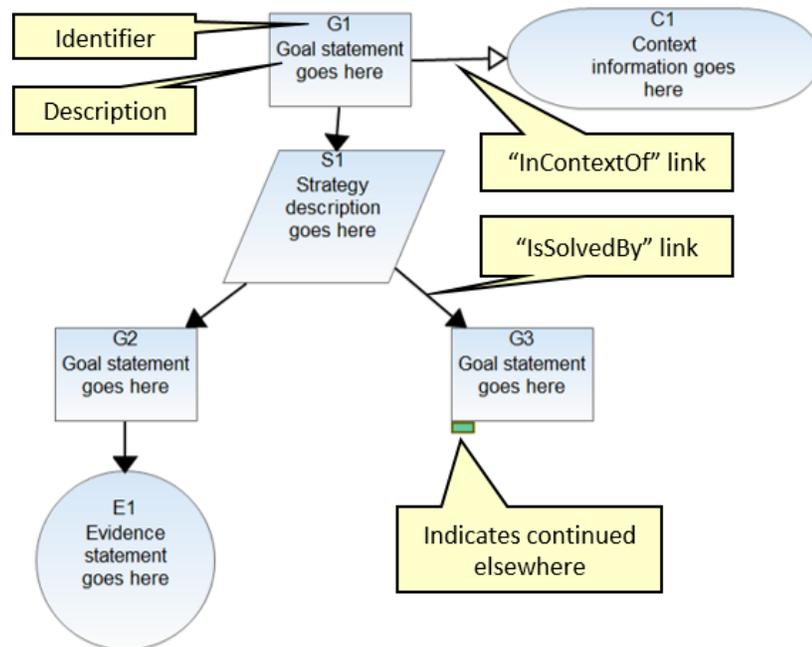


Figure 8. Guide to this paper's use of Graphical Structuring Notation.

Each of these has an identifier (unique in the diagram in which these occur) and a textual description. The “IsSolvedBy” arrows indicate the structure of the argument, i.e., the arrow points from an item to the item that supports it (e.g., from a goal to the evidence that shows the goal to be true). Conventionally, argument structures composed of these elements are organized in a top-down fashion. Argument structures can and do get quite large, so it is common to decompose them into pieces, allowing an argument to continue on another page.

Note that the notation does not preclude goal statements that are false, and arguments that are invalid; some form of review is necessary to assure this is not the case when an argument is presented.

Acknowledgments

This research was carried out at the Jet Propulsion Laboratory, California Institute of Technology, under a contract with the National Aeronautics and Space Administration and funded through the internal Research and Technology Development program. The authors thank Mike McHenry for providing us plentiful information on M2020, and Ewen Denny for allowing us use of AdvocATE for creating and editing assurance cases.

Reference herein to any specific commercial product, process, or service by trade name, trademark, manufacturer, or otherwise, does not constitute or imply its endorsement by the United States Government or the Jet Propulsion Laboratory, California Institute of Technology.

References

- ¹Maimone, M., Johnson, A., Cheng, Y., Wilson, R., and Matthies, L., “Autonomous Navigation Results from the Mars Exploration Rover (MER) Mission,” *Experimental robotics IX*, pp. 3-13, 2006.
- ²Greeley, R., Whelley, P.L., Arvidson, R.E., Cabrol, N.A., Foley, D.J., Franklin, B.J., Geissler, P.G., Golombek, M.P., Kuzmin, R.O., Landis, G.A., and Lemmon, M.T., “Active dust devils in Gusev crater, Mars: observations from the Mars exploration rover spirit,” *Journal of Geophysical Research: Planets*, 111(E12), 2006.
- ³Francis, R., Estlin, T., Gaines, D., Bornstein, B., Schaffer, S., Verma, V., ... & Thompson, D. (2015, October). “AEGIS autonomous targeting for the Curiosity rover's ChemCam instrument,” *Applied Imagery Pattern Recognition Workshop (AIPR)*, IEEE pp. 1-5, 2015.
- ⁴Kornfeld, R. P., Prakash, R., Devereaux, A. S., Greco, M. E., Harmon, C. C., and Kipp, D. M., “Verification and validation of the Mars science laboratory/curiosity rover entry, descent, and landing system,” *Journal of Spacecraft and Rockets*, 51(4), pp. 1251-1269, 2014.

- ⁵Ono, M., Rothrock, B., Almeida, E., Ansar, A., Otero, R., Huertas, A., & Heverly, M., "Data-driven surface traversability analysis for Mars 2020 landing site selection," *IEEE Aerospace Conference*, pp. 1-12, 2016.
- ⁶Reinhart, D.J., Knight, J.C., and Rowanhill, J., "Understanding What It Means for Assurance Cases to 'Work'," NASA/CR-2017-219582, 2017
- ⁷National Research Council, *Software for Dependable Systems: Sufficient Evidence?* The National Academies Press, Washington, DC, 2007.
- ⁸Weinstock, C. B., Goodenough, J. B., and Hudak, J. J. (2004). Dependability cases (No. CMU/SEI-2004-TN-016). CARNEGIE-MELLON UNIV PITTSBURGH PA SOFTWARE ENGINEERING INST., 2004.
- ⁹Feather, M. S., and Markosian, L. Z., "Building a safety case for a safety-critical NASA space vehicle software system," *IEEE Fourth International Conference on Space Mission Challenges for Information Technology (SMC-IT)*, IEEE, pp. 10-17, 2011.
- ¹⁰Nguyen, E. A., and Ellis, A. G., "Experiences with assurance cases for spacecraft safing," *IEEE 22nd International Symposium on Software Reliability Engineering (ISSRE)*, IEEE, pp. 50-59, 2011.
- ¹¹Denney, E., and Pai, G., "A methodology for the development of assurance arguments for unmanned aircraft systems," *33rd International System Safety Conference (ISSC 2015)*, 2015.
- ¹²Brat, G., Davies, M., Giannakopoulou, D., and Neogi, N., "Workshop on Assurance for Autonomous Systems for Aviation," NASA/TM-2016-219466, 2016.
- ¹³Kelly T, and Weaver R. "The goal structuring notation—a safety argument notation," *Proceedings of the Dependable Systems and Networks Workshop on Assurance Cases*. Citeseer, 2004.
- ¹⁴Kelly, T., and McDermid, J., "Safety case construction and reuse using patterns," *Safe Comp* 97, pp. 55–69, 1997.
- ¹⁵Kuhn, D.R., Kacker, R.N., and Lei, Y., "Practical Combinatorial Testing," NIST Special Publication 800-142, Oct. 2010.
- ¹⁶Kuhn, D.R., Wallace, D.R., and Gallo, A., "Software Fault Interactions and Implications for Software Testing," *IEEE Transactions on Software Engineering*, 30(6), pp. 418-421, 2004.
- ¹⁷Bryce, R., Rajan, A., and Heimdahl, M.P.E., "Interaction Testing in Model Based Development: Effect on Model Coverage," *IEEE 13th Asia Pacific Software Engineering Conference (APSE'06)*, 2006.
- ¹⁸Kuhn, D.R., and Reilly, M.J., "An Investigation of the Applicability of Design of Experiments to Software Testing," *27th NASA/IEEE Software Engineering Workshop, NASA Goddard Space Flight Center*, 4-6 December, 2002.
- ¹⁹Alexander, R., Hawkins, H., and Rae, D., "Situation coverage - a coverage criterion for testing autonomous robots," Technical Report YCS-2015-496, Department of Computer Science, University of York, February 2015.
- ²⁰Mullins, G.E., Stankiewicz, R.G., Hawthorne, R.C., Appler, J.D., Biggins, M.H., Chiou, K., Huntley, M.A., Stewart, J.D., and Watkins, A.S., "Delivering Test and Evaluation Tools for Autonomous Unmanned Vehicles to the Fleet," Johns Hopkins APL Technical Digest, Volume 33, Number 4, pp. 279-288, 2017.
- ²¹Denney, E., Pai, G., and Pohl, J., "AdvoCATE: An assurance case automation toolset," *Computer Safety, Reliability, and Security*, pp. 8-21, 2012.
- ²²Denney, E., & Pai, G., "A formal basis for safety case patterns," *International Conference on Computer Safety, Reliability, and Security*, pp. 21-32, 2013.
- ²³Vectra Group Limited, "Literature Review on the Perceived Benefits and Disadvantages of UK Safety Case Regimes," Report No. 402083-R01, 2003. URL: <http://www.hse.gov.uk/research/misc/sc402083.pdf> [cited 4 June 2017].
- ²⁴Rinehart, D.J., Knight, J.C., and Rowanhill, J., "Understanding what it means for assurance cases to 'work'," NASA/CR-2017-219582, 2017.
- ²⁵Duan, L., Rayadurgam, S., Heimdahl, M., Ayoub, A., Sokolsky, O., & Lee, I., "Reasoning about confidence and uncertainty in assurance cases: A survey," *Software Engineering in Health Care*, 2014.
- ²⁶Kelly, T. P., "A six-step method for the development of goal structures," York Software Engineering, Flixborough, UK, 1997.
- ²⁷Spriggs, J., *GSN-The Goal Structuring Notation: A Structured Approach to Presenting Arguments*, Springer Science & Business Media, London, 2012.
- ²⁸Alexander²⁰⁰⁸Alexander, R., and T. Kelly. "Simulation and prediction in safety case evidence," International System Safety Conference, 2008.
- ²⁹Yamamoto, S., and Matsuno, Y., "An evaluation of argument patterns to reduce pitfalls of applying assurance case," in *Assurance Cases for Software-Intensive Systems (ASSURE), 2013 1st International Workshop on*, pp. 12-17, 2013..
- ³⁰NASA, Dezfuli, H., Benjamin, A., Everett, C., Smith, C., Stamatelatos, M., & Youngblood, R., "NASA System Safety Handbook, Volume 1; System Safety Framework and Concepts for Implementation" NASA/SP-2010-580, 2010
- ³¹Hawkins, R., Clegg, K., Alexander, R., and Kelly, T., "Using a Software Safety Argument Pattern Catalogue: Two Case Studies," *Computer Safety, Reliability, and Security*, pp. 185-198, 2011.
- ³²Kane, A., "Runtime monitoring for safety-critical embedded systems," CMU, 2015.
- ³³Maimome, M., Cheng, Y., and Matthies, L., "Two years of Visual Odometry on the Mars Exploration Rovers," *Journal of Field Robotics* 24(3), pp. 169-186, 2007.
- ³⁴Johnson, A., Wilson, R., Cheng, Y., Goguen J., Leger, C., San Martin, M., and Matthies, L., "Design Through Operation of an Image-Based Velocity Estimation System for Mars Landing," *International Journal of Computer Vision* 74(3), pp. 319–341, 2007.