CSI: Cobots - Deliverable A2

12/2020

Situation space description/model, prototype testing simulation, and preliminary evaluation of testing

Benjamin Lesage, University of York
Rob Alexander, University of York

Cooperative robots (cobots) [25] aim to allow human operators and robot workers to share the same workspace, and work jointly to achieve a common goal. Safety concerns tend to result in spaces where the cobot is isolated from the operators through physical barriers [1, 24]. These safety constraints tend to limit the level of cooperation between human and robot, and mitigate the advantages of deploying cobots in the industrial space.

The CSI: Cobot project [2] aims to alleviate the barriers deployed in cobot systems. The project proposes novel sensing and control techniques to improve cobots' awareness of their environment, especially regarding interactions with human operators. A crucial condition to the adoption of new techniques is achieving confidence in the overall safety of the system. This report focuses on the safety aspects of the CSI: Cobot project, in particular on the methods under investigation to understand the impact of changes in the system configuration.

Simulation-based techniques, e.g. the CARLA simulator [3], are a common approach to the evaluation of autonomous systems [4, 26, 27]. Simulations allow fast iterations over varied environmental configurations, including hazardous ones, without endangering the system itself, or its environment. However, the use of simulations for the evaluation of safety constraints raises important issues. First, the safety case must ensure the simulated environment is representative of the system under consideration [5, 28]. Second, the tools must decide on a set of configurations to drive the evaluation and evaluate confidence in the system safety [6].

Our approach relies on simulation-based, situation coverage testing to evaluate the safety of a cobot system. It relies on the safety analysis of the cobot system under consideration. The analysis identifies accidents and losses that arise from unsafe operation and the safety artifacts, generated as part of the analysis, notably capture undesirable situations which though not hazardous by themselves may lead to a loss. In the context of situation coverage [7] such situations inform our exploration of the system configurations and an evaluation of confidence in the system safety.

The present contribution focuses on an industrial use case involving the cooperation of a human operator and a robotic arm to process assemblies. The operator provides an unprocessed assembly at a designated work bench. The arm then picks up the assembly and carries it to a spot welder, where it is processed, before returning to the same work bench for a handover. All the processing occurs within a walled cage, with range sensors to ensure no operator is present while the welder is active or the arm is moving. Note that the general principles of our approach are not tied to the specific use case or tools we discuss in the following.

This report presents a preliminary approach to the safety assessment of our industrial case study. We first introduce the general principles of our approach (Section 1). We then discuss the prototype simulation tool our evaluation relies on, and its integration with the wider analysis (Section 2), before concluding with a preliminary evaluation of the analysis (Section 3). As this is an early evaluation of the framework, we highlight current progress, as well as limitations and open problems.

1 Analysis Framework

Our framework relies at its core on the artifacts produced by the safety analysis. Safety artifacts provide information on the entities involved in the system, their interactions, and how those may lead to hazards. The safety analysis thus informs multiple aspects of the testing harness. The simulator needs to capture those elements raised by the analysis, and be open to configuration for exploring identified undesirable situations. The method also needs to cope with an evolving system design as new techniques to remove some of the existing barriers are considered for evaluation.

This section first introduces an overview of our methodology, before focusing on the steps most relevant to the safety assessment, namely the safety analysis itself, the monitoring of safety events, and the exploration of the system configuration space. We defer the design of the Digital Twin and its integration with the framework to the next Section.

Overview

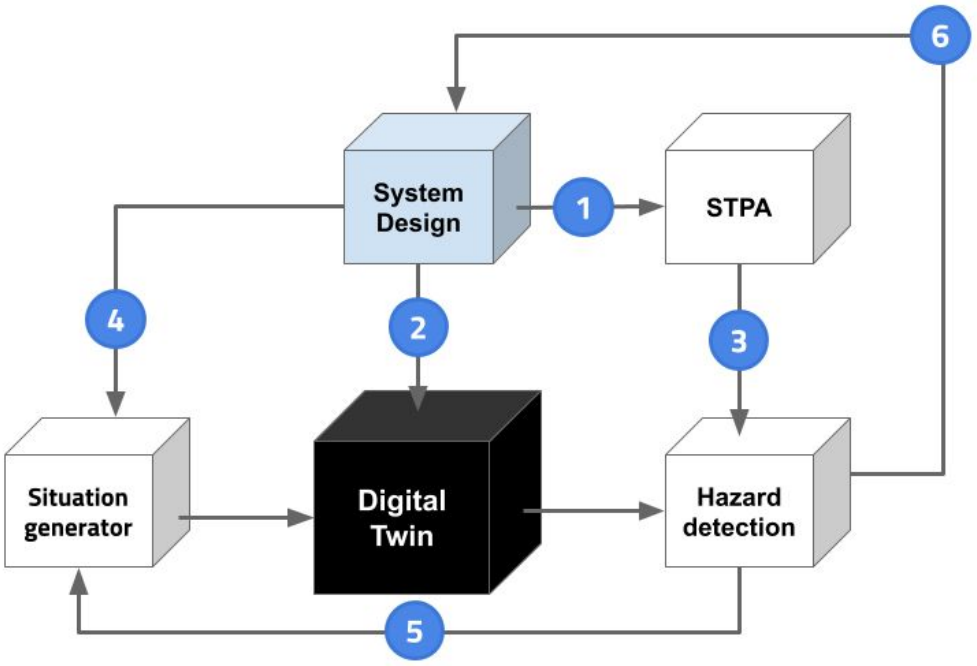

Figure 1. Overview of the Analysis Framework

Figure 1 presents the overall workflow of the analysis framework. The system design is at the core of the process as it informs all other steps, defining the environment, interacting entities, their behaviour, and the safety measures deployed to prevent or mitigate hazards. A principle of the framework is building confidence into the system by testing it, searching for safety-relevant configurations, to provide feedback regarding safety aspects into the design.

The first step is to perform a safety analysis of the system (1) to understand the occurrences of hazards in the system, the sequence of actions involved, and the conditions leading to such events. The system design further informs the development of a Digital Twin (2) which constitutes a baseline for our simulation-based approach. The Digital Twin also allows the evaluation of new techniques before their deployment in the actual cell. The artifacts of the safety analysis provide information on safety-relevant situations in the system and the conditions for their detection (3). The system design further constrains acceptable configurations of the system. It defines elements open to variations and their degree of freedom, thus outlining the domain of our search (4). Safety events captured running the simulation, from a generated configuration, provide feedback on situations of interest (5) during the search. Finally, the coverage of generated and observed situations during analysis (6) provides some confidence in the safety of the system, or highlights shortcomings that need to be addressed.

As an example, consider a configuration with a near miss: *“the arm was moving while its path was obstructed”.* The configuration holds the pre-conditions for the accident: *“collision between a moving arm and an operator”*. The situation generator can guide testing towards similar configurations to evaluate the occurrence of safety risks, or additional sensing and control that can be incorporated in the cell design to prevent such occurrences.

Safety Analysis

The Safety Analysis aims to understand the safety of the system, by identifying potential hazards, and the situations or causes leading to these events such that they can be managed. To that purpose, the analysis needs to be aware of the components and entities interacting in the system as well as its operating environment. Without loss of generality, we present the Systems Theoretic Process Analysis (STPA) technique as the underlying safety analysis. The results of the application of STPA to our use case and our experience have been documented in [9].

STPA [8] originates from systems approaches to safety engineering. Accidents are assumed to arise from insufficient feedback or inadequate control in the system, as modelled by a control structure. The STPA analysis process is defined as follows:

- Identify accidents and losses scenarios: these encompass a range of undesired events such as damage to property, injury to humans, or environmental pollution, e.g. an operator is within close proximity of the spot welder in the cell while it is active.
- Construct the system control structure: the control structure captures the entities in the system and the flow of actions and information between them, e.g. the cobot controller may instruct the arm to move to a specific position while the arm provides feedback on its current joint positions.
- Identify unsafe control actions: unsafe control actions correspond to the execution of actions in undesirable configurations of the environment and system, e.g. the controller issues a move order (action) when the arm is under maintenance (configuration).
- Identify causal factors and control flaws: this steps considers how unsafe actions arise as a result of inadequate control, e.g. the cobot controller might instruct an arm to move in presence of an obstruction if it is above the range sensors.

The method supports incremental refinements of the analysis results, allowing the granularity of the results to be controlled as the design of the system gets refined, or as additional control is included. This suits our analysis framework as we consider the system from a high-level perspective to cope with variability. Safety analysis artifacts persist across minor design variations, such as the ones we consider inside the CSI: Cobot project.

Our method relies on the artifacts produced by the STPA technique, notably hazards and unsafe control actions. Hazards identify the occurrence of events which by definition challenge the safety of the system. Unsafe control actions (UCAs), while not hazardous themselves, are undesirable. Each UCAs is composed of an action and a situation, the execution of the former or lack thereof in said situation may give rise to a hazard. Monitoring for UCAs in simulation-based testing can help identify hotspots in the configuration space, and focus the analysis effort on those configuration regions more likely to result in hazards. In turn, considering both the action and situation components of a UCA individually can guide the search towards configurations likely to trigger UCA.

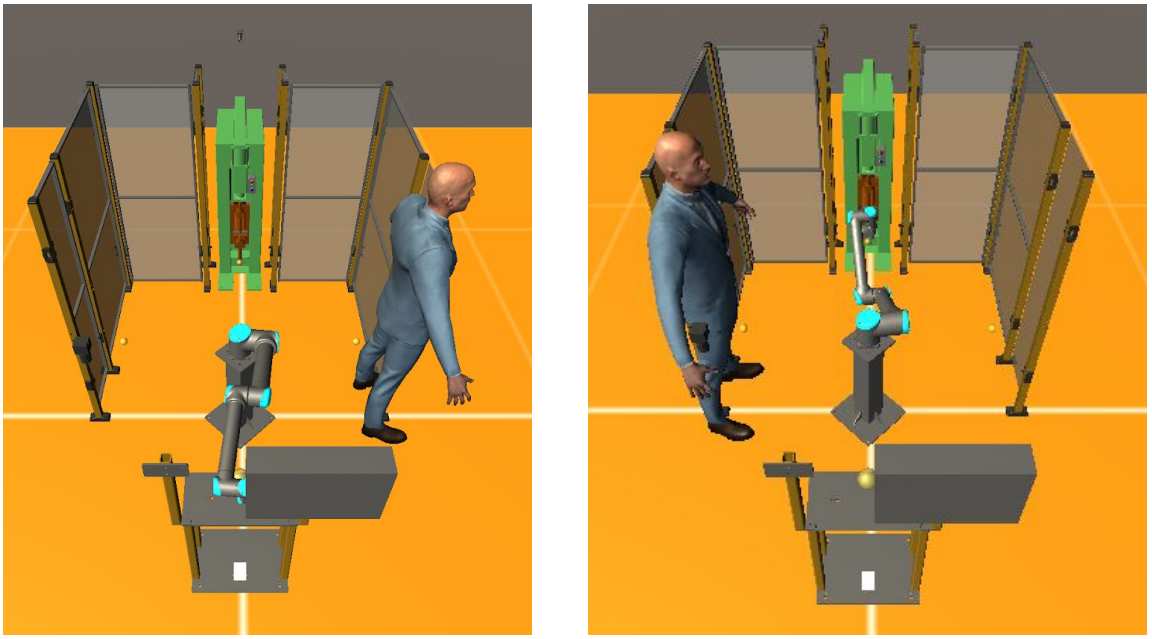

Figure 2. Example configurations of the considered use-case with an operator (blue, resp. right and left) in the cell on the path of the cobot (grey, centre) to the spot welder (green, top).

As an example consider the cell as depicted in Figure 2. Collisions between an operator and the cobot arm are a potential hazard (Hazard H3 [9]). Such an event only occurs if the operator is in the cell within reach of the arm while the arm or operator is moving, as captured by UCA-9-P2 ("*The Cobot moves position while its path is obstructed*" [9]). Identifying configurations where the situation aspect of UCA-9-P2 occurs, i.e. "there is an obstruction in the cell", can guide the search towards triggering the UCA itself, and in turn the hazard. Figure 3 illustrates the overlap between those conditions, H3, UCA-9-P2, cell obstruction and arm moving.

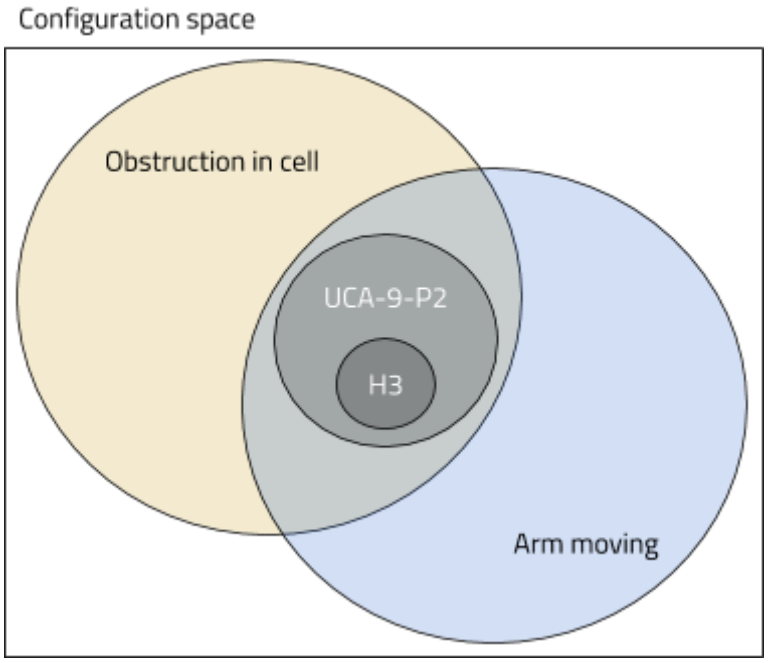

Figure 3. Overlap between the collision hazard H3, related UCA-9-P2, obstruction in cell, and arm moving situations in the system configuration space.

Safety Monitoring

Safety Monitoring aims to identify, from the event traces generated by a simulation run, the set of safety-relevant situations that occurred during the run. The tool monitors the run for hazard occurrences if any, or interesting, safety-related situations (UCA or their components). Runtime Verification methods [10] provide a vast array of tools and techniques to identify the violation of specific properties in a system, either through offline [11] or online monitoring [12].

The situations we monitor for are formalised from STPA artifacts, namely Hazard and UCA. The textual description of artifacts needs to be translated into a formal specification for monitoring which might introduce errors in the process. We rely on a validation step to assess the suitability of our formal artifact specification. Each formalised artifact is tested against a set of traces, each defining a simple sequence of events, with expected outcomes, i.e. the safety condition occurs or not. Considering the example of UCA-9-P2 in Figure 3, the test only defines when the arm is moving and when there is an obstruction in the cell with different tests capturing different orderings and overlaps of the two predicates.

The simulation needs to expose the required information for monitoring. The safety analysis thus informs the design of the simulation not only in terms of the components and actions that should be modelled, but also regarding some of the events that need to be tracked. The safety monitors can be incorporated in the simulation tool itself, or applied to the output of the tool. We rely on a separate, offline monitoring approach to define our safety monitoring.

This allows the parallel development of the simulation environment, and the definition and validation of monitors.

There is a distinction between the actual state of the system (physical or simulated) as perceived by its components. The safety monitors might rely on information that is unavailable or incorrect in the physical system due to sensing gaps or faults. This contextual notion of ground truth is another key to defining requirements on the simulation. Furthermore some conditions may need to be simplified for evaluation and monitoring, preferably subsuming the target condition to prevent false negatives at the cost of false positives. For example, instead of a complex physics simulation to model whether an assembly is correctly secured by the end effector of an arm, we can rely on a stochastic model to simulate assembly slippage.

We use Linear Temporal Logical (LTL) [13] to model safety artifacts. An LTL formulae captures a condition on the future of a path, combining predicates using logic operators, e.g. the operator is grabbing a component **and** the cobot is moving, and temporal ones, e.g. the cobot does not move **until** the cobot is configured. The monitors exist as objects in the Python language. This provides for our validation requirements through the Python testing facilities. Furthermore, we can easily identify during integration which events required for the evaluation of a monitor are missing from a given simulation trace. Missing events can stem as an example from inappropriate configuration, e.g. if no space is identified as the work cell for detecting obstructions or if entities names have changed.

Situation generation

Automated testing techniques for autonomous robots need to address a number of challenges. The tools need to evaluate the confidence in the safety of the system achieved by a generated test campaign, to assess whether or not additional testing effort is required at the run level (for a given configuration) or at the campaign level (generating more configurations). Macro and micro situation components need to be identified to capture respectively components which can be configured to impact the system, and components that the system should monitor for confidence evaluation. We first introduce the notion of situation coverage, before discussing its implication in the proposed framework, and the initial situation space definition for our use case.

Situation Coverage

The coverage achieved by a test campaign is an important indicator of the confidence in the System under Test (SuT) behaviour. High coverage speaks for the quality of the testing strategy and provides an indication that sufficient testing has been performed. Metrics such as MCDC [16] focus on code coverage, ensuring as an example the absence of dead code. However structural metrics are not adapted to evaluating functional requirements even more so the emergent behaviours that may arise from autonomous robots [17].

Situation coverage [7] was proposed to address these issues and introduces a coverage criterion adapted to the testing of autonomous robots. The underlying principle is to identify

components of the environment the SuT might encounter, identify their domain, and ensure they are evaluated during testing, on their own or in combination with other situation components. Testing of an autonomous vehicle would be required to navigate different intersections (components) with various shapes (domain). This would be combined with the type of vehicles it might encounter at the intersection, their direction of travel, etc.. Initial evaluation of the method showed promising results, but with a strong requirement on a sufficiently representative environment [19].

Macro situation components capture the SuT configuration domain, reasonable operating conditions for the system. They define the axes (parameters) and boundaries (range of values) explored during testing. The identification of situation components is obviously informed by the system design, but the safety analysis also provides information about parameters which might jeopardize the safety of the system and should be evaluated during testing. Configuration points which appear insignificant may still be included for blocking purposes, to provide information on the sources of variability in the system.

Micro situation components, or in-run components, encompass small scale situations the system encounters while it runs. They provide feedback on the situations the system has encountered, or its reactions to different scenarios. Micro situation components are identified from more functional aspects of the system, capturing the set of events it should cope with, e.g. a human enters the cell while an assembly is being welded. The definition of such components thus depends on the set of situations of interest in the SuT.

Test generation

The main purpose of the test generation step is to trigger situations of interest, especially corner cases in the system, to assess the safety of the system. The configurations evaluated through the simulator and generated by the framework should remain within reasonable operational parameters of the system. As focus is given to safety relevant situations, the distribution of observed situations may not follow the distribution of cases in the deployed system.

The distinction between macro and micro components defines two levels of coverage requirements. At the macro-level, the components define a requirement on the inputs or combinations thereof exercised during testing. Micro situation components define the set of situations that should always, or never, be observed during the test campaign. Note that in either case, coverage values below 100% are hard to interpret without a confidence metric, as the uncovered cases might withhold safety issues.

Our approach precludes the application of model-based techniques to generate tests [20]. There is no information on relations between the inputs and outputs of the system, that is no a priori knowledge on which situations a specific configuration triggers. Simulation is used to evaluate this impact. We instead rely on combinatorial or Design of Experiments (DoE) methods [21] to build the initial set of configurations explored by the framework. From a set of components and their levels, i.e. the definition of the configuration space, such methods generate a set of complete or partial covering experiments. Non-combinatorial designs in

particular aim to produce experiments with sufficient data to understand the impact of the different components and their n-wise combinations on the system's response, with a reduced number of configurations..

We rely on the safety analysis artifacts to assess the coverage achieved by our test campaign at the micro-level. The occurrence of hazards or UCAs during testing has a negative impact on the system, with repeated occurrence of the same UCA or various UCA being triggered resulting in a lower confidence in the safety of the system. This approach carries the result of the safety analysis and its artifacts into the testing framework.

UCA and their individual components also provide information to guide the search towards regions of interest, as exemplified in Figure 3. Sensitivity analysis and principal component analysis can further provide useful feedback on the macro components which contribute to the unsafe or undesirable response of the system. Note that the framework is in its early stages of evaluation, and we are still refining the definition of confidence, coverage, and the situation generation approach to ensure both quantifiable and achievable metrics.

Situation space preliminary definition

Macro-level situations define the scope of possible configurations for the system under test. They capture the nominal operational parameters of the system as well as a tolerance within those parameters. As an example, the system operates without obstruction in the cell but should be able to react if one occurs. These situations should not solely be driven by the requirements or software paths. Instead, one needs to enumerate the components of the situations, the different actors, the different missions, the entities involved, possible interactions, and other environmental factors.

We initially rely on a manual assessment of the situation space in the considered use case. The assessment is informed by the safety analysis, and by discussions with the various project partners. Consider a camera-based sensor. The position and orientation of the sensor are inherent configuration parameters which contribute to defining the portion of the workspace accessible to the sensor. Environmental factors such as occlusions, lighting conditions, or the colours of the different entities are part of the underlying configuration points which might impact the behaviour of the sensor.

The situation space is divided into regions of increasing size outlined in Table 1. To evaluate the resilience of the system, as we gain confidence that the system is safe in a given region, testing can proceed to the next region, i.e. a wider situation space. We first consider all operational parameters to be within nominal values, all entities behave and are set up as expected (Nominal). We then introduce variations in the environment, ones the system (correctly set up) should be able to cope with such as occlusions and obstructions. Variations in the system set up, incorrect sensors placement or calibration, are then introduced. The final step is the injection of faults in the system, sensors becoming unresponsive, false data, delayed information, lack of response to stimuli, or impulsive action.

Component	Nominal	Behaviour-I	Behaviour-II
Wall	Wall enclose 3 sides of workspace	Wall enclose 0..3 sides of workspace	-
	Wall correctly positioned around workspace	-	-
Range sensor	No occlusions	Potential occlusions	-
	Viewpoint covers the space in front of the tool	-	-
	Sensor at shin height	-	Sensor at different heights
Camera sensor	Viewpoint may be restricted to workspace portion with at least bench, cobot, and tool	-	Over/under/partially covering workspace
	Viewpoint may cover zones outside workspace	-	-
	No occlusions	Potential occlusions	-
	No obstructions	Potential Obstructions	-
Controller	Executes required sequence	-	-
Cobot	In reach of welder	-	May be out of reach
	In reach of assembly bench	-	May be out of reach
	No initial obstructions	Potential obstructions	-
Operator	Executes required sequence	-	May execute wrong sequence
	No entry in workspace, move around workspace	May enter workspace	-
	Single operator	-	1+ operator(s)
Assembly	Always presented correctly	May be presented incorrectly	-

	No defect	-	May have defects
	Correct assembly type	-	May be incorrect type

Table 1. Preliminary of the situation space for the safety evaluation of the considered use case.

2 Prototype simulation

The simulator captures a model of the system and its behaviour under controlled configurations. Our framework can quickly evaluate the system's response to numerous situations, including potentially hazardous ones, without any risks to its actual actors. The Digital Twin [14] developed in the CSI: Cobot project bridges the gap between the simulation and the real world, with physical components represented in and impacted by the simulation as they operate, or vice versa. It provides an environment to assess the impact of new tools in the system before their actual deployment, and it is therefore well suited to our approach..

The use of a Digital Twin further provides for a gradual safety assessment of the system. Pure simulation runs, using only virtual entities, can be used to assess the occurrences of unsafe situations on the system. Twinned tests, mixing virtual humans and actual controllers or actual humans in virtual environments, then validate the observations of the analysis once sufficient confidence is achieved from the initial simulations. Complex behaviours for a simulation, such as precise operator gestures, can thus be accounted for by projecting an actual operator in a mixed-reality environment.

The simulation is considered as a black box in our framework when it comes to running configurations and observing safety events. The framework makes no assumption on how the simulation behaves under a given configuration, but it needs to include those design elements identified by the safety analysis. This includes both requirements for safety monitoring and configuration points. In other words, the simulation exposes a ground truth, within the context of the simulated environment. The ground truth while safety-focused can be used as a baseline for the evaluation of other techniques, e.g. by comparison of the predicted obstruction by a sensor to the obstruction as modelled in the simulation. A separate validation step or safety assessment might help identify risks related to discrepancies with the world in the simulation tool and their mitigation.

Processing framework

The processing framework for our analysis focuses on interactions with the Digital Twin (illustrated by Step 5 in Figure 1), ensuring valid configuration files are generated and observations can be processed to identify situations of interest. This is a crucial component to guide the search through the system configuration space towards safety events. In this section we focus on technical aspects of the integration with the Digital Twin, the rationale behind the configuration generation and safety monitoring are discussed in the following sections.

The identification of safety-relevant events in our framework uses linear temporal logic formulas, combining predicates with logic and temporal operators, each capturing a small component of the situation. The processing step captures the value of these situation components from the Digital Twin message record as a set of time-series. Communication in the Digital Twin indeed occurs through messages exchanged between the different entities. This includes both functional messages, e.g. orders issued by the cobot controller to the arm, and safety ones, e.g. triggers on collision between entities. The messages are stored as typed, nested key-value containers. The container hierarchy, the keys traversed to reach a value, provide information about the variables captured in each message. We automatically extract such variables and their values from each message. Using the message timestamp, our framework can then generate corresponding time-series for the variables.

As an example consider the Joint Status message in Figure 4 from an arm which captures the status for a number of joints (under key "**joint**"). The status of joint **n** is accessed through **message["joint"][n]** and the emitting entity ("**ur10**") under **message["entity"]**. The framework automatically extracts the variables "**ur10/joints/n**", using the entity to discriminate between different joints (Figure 4). The processing is thus independent of the message type hierarchy defined in the Digital Twin, supporting new message types without modifications. If aliasing is required or other post-processing, i.e. different message types referring to the same variable use different conventions to identify it, the framework supports ad-hoc remapping of values.

JointStatus
{
time: **t**,
id: "UR10",
joint: [
{pos: 0},
{pos: 2}
]
}
	At time **t**:
	ur10/joint/0/pos = 0
	ur10/joint/1/pos = 2

Figure 4. Example of message collected from the Digital Twin (left) and the corresponding variables (right) used for safety monitoring.

The Digital Twin provides hooks to configure existing entities in the simulated environment. As for the safety monitoring, the Digital Twin needs to expose sufficient control over the elements captured in the safety analysis and situation space definition. We extended the Digital Twin to ensure that all entity properties can be controlled through a configuration file. As more entity types and entities are added into the environment, their properties are automatically exposed through said configuration file. Upon startup, the simulated scene is set up according to baked-in default values with the configured values overriding the default ones if defined. The simulation then starts stimulating entity behaviours, e.g. the cobot controller issues commands to guide the cobot on a specific path. The overrides approach to

configuration allows the search to define and focus on relevant properties of the configuration space. Initial evaluations can also focus on a more limited set of macro-components, knowing sensible defaults are used otherwise, before increasing the domain of the search.

3 Preliminary evaluation

The ongoing development of the Digital Twin and our analysis framework, as this report is published, precludes a complete evaluation of our approach to the assessment of the safety of our industrial study. In particular, sensing and control are limited and permit neither a full implementation of the considered use case nor a full assessment of all formalised safety artifacts. We focus on Hazard H3 and the related UCA-9-P2 introduced in Section 1:

- H3: "*Equipment or Component subject to unnecessary stress";*
- UCA-9-P2 "*The Cobot moves position while its path is obstructed*".

We first discuss in the section the setup of our simulation and some abstractions to allow for a preliminary evaluation of our framework and toolchain. In a second step, we evaluate the occurrences of hazards in the scene using configurations generated through different methods.

Scene Setup

Our scene describes the default setup of our industrial use-case, depicted in Figure 5. All major components are in place, the welder, the cobot arm, the shared bench, and the surrounding cage. The highlighted zone in the middle defines the cell region. The arm is programmed to follow a path between the work bench defined as a set of waypoints from the bench at the bottom, to the welder at the top, and then back to its original position. There is no guarantee on the intermediate positions of the arm on that path, except for waypoints on the left and right of the cell.

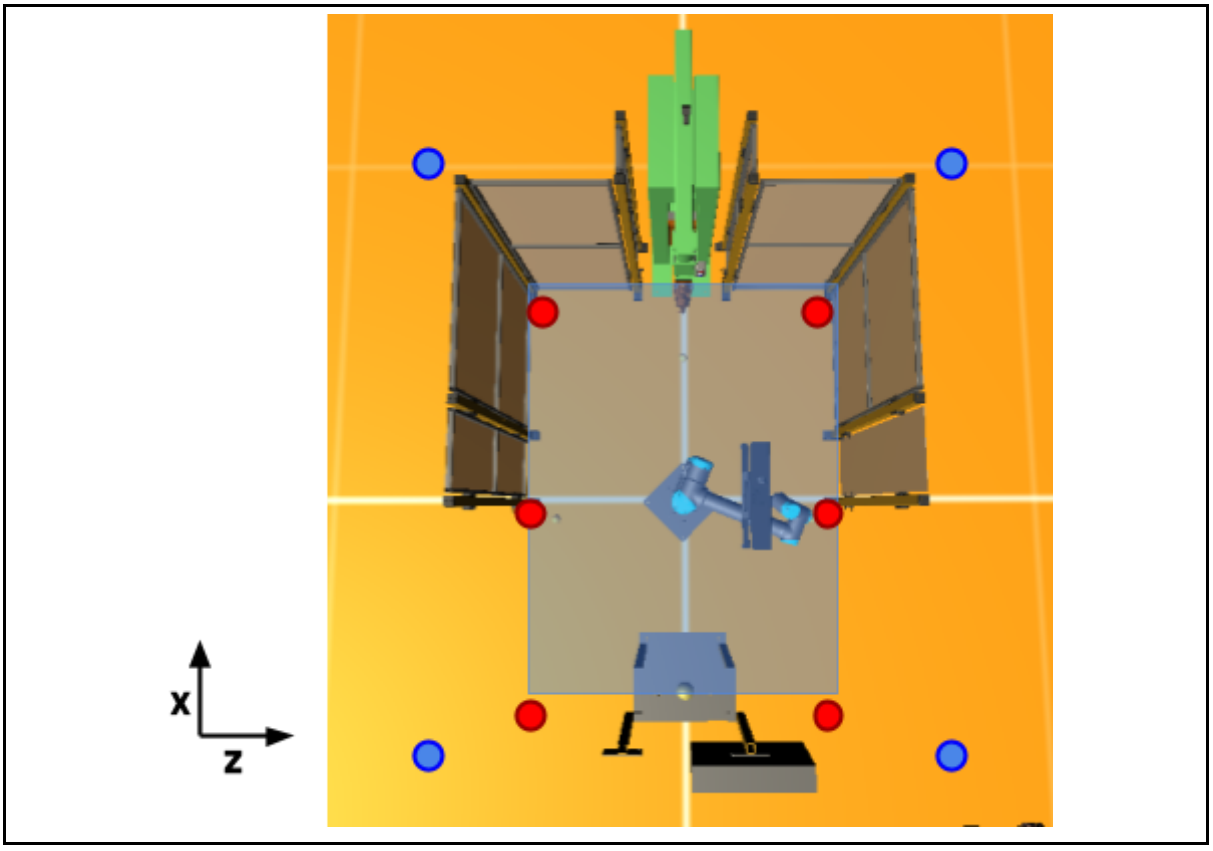

Figure 5. Default scene configuration used for the preliminary evaluation viewed from the top with the work cell region outlined (right)

The operator should keep out of the cell while either the arm or the welder are active less he puts himself at risk of injury. While sensors are modelled in the scene, i.e. cameras and range sensors, in the absence of a safety controller the arm will continue on its path irrespective of obstructions. The occurrence of UCA-9-P2 or H3 is thus tied to the position of the operator respectively in relation to the cell, and the arm computed path.

We further abstract the occurrence of hazards in the evaluation, widening the condition as discussed in Section 1. The simulation does not yet include a physics model and forces on collisions between two entities cannot be simulated. H3 is simplified to assess the occurrence of a collision between the arm and an operator irrespective of their relative speeds. Similarly, the kinematics manager cannot guarantee the trajectory followed by the arm between two waypoints. The operator is assumed to be on the path of the arm as soon as it enters the cell. Three variables are extracted from simulation runs:

- The logical positions of the operator, in the cell, at the workbench, or at the welder;
- Whether the arm is moving or not at any given time;
- Occurrences of collisions between the arm and the operator.

Results

We use the toolchain to evaluate our assumptions on the feasibility of our approaches and the related assumptions. Work in the project focused on building the Digital Twin and the

required tooling to allow for safety assessments using the proposed framework, i.e. simulations under controlled configurations and the identification of safety events. The results on this section thus focus on state of the art heuristics for the generation of tests, applied to a limited input domain: the spatial coordinates of the operator in the simulation.

Relation between UCA and Hazard

We first evaluate our hypothesis of a relation between the occurrences of UCAs and Hazards, that is the hypothesis that the occurrence of a UCA, as captured by the safety analysis, is necessary for the occurrence of a hazard, as depicted in Figure 3. Focus of testing effort on configurations where UCA (or components thereof) occur increases the likelihood of discovering latent hazards.

We generate 100 configurations of the environment, each with the operator in a random position within the region defined by the outermost spots (blue) on Figure 5. Each configuration is run through the simulation for a full cycle, the arm traversing all its waypoints. Processing the simulation output classifies runs into ones where no safety situation occurred, either the UCA or Hazard occur independently, or both occur. The distribution of runs across these categories is presented in Figure 6.

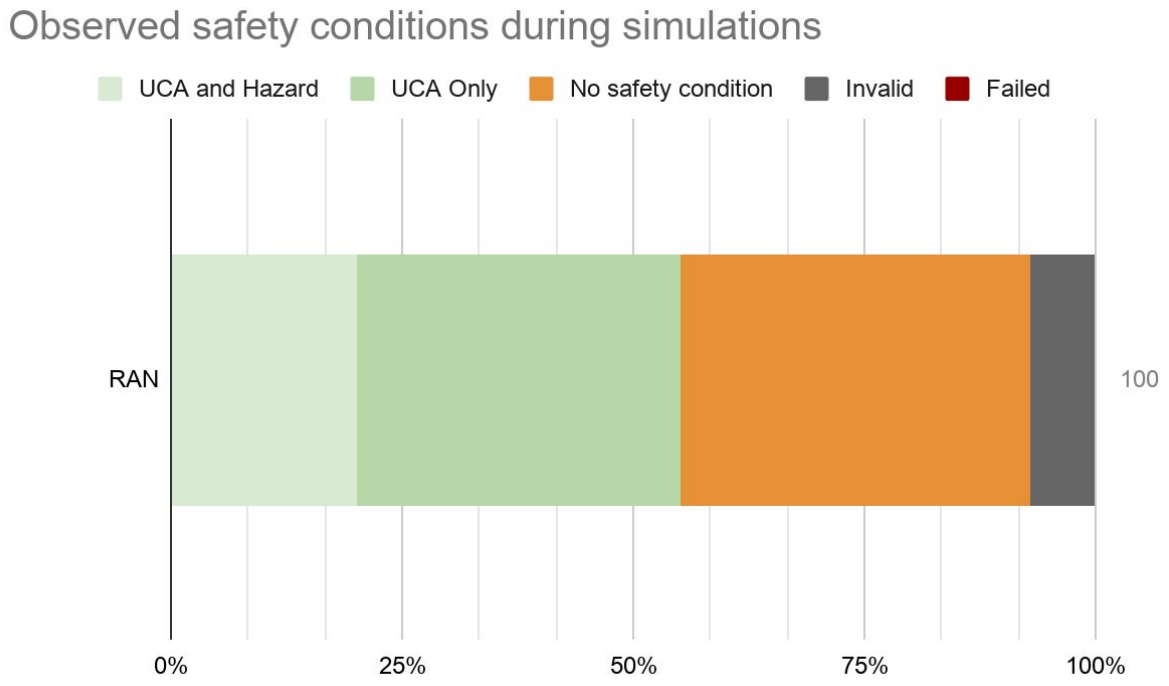

Figure 6. Distribution of safety situations detected across 100 runs with random positioning of the operator.

During our experimentations, we observed no situation where a collision occurred (Hazard) without the assorted UCA being observed as well. While both situations are defined using different events, operator position and arm moving vs. actual collision, there is indeed a causality relation between them. This relation is captured as part of the safety analysis. The

UCA was observed in around 55% of the simulation runs, either on its own or in conjunction with the hazard.

Some configurations (7) were deemed invalid with the operator placed on top of the arm. The issue lies not with the formatting of the generated configuration, but its semantic in the context of the modelled environment. We expect such occurrences to be more frequent as the scene is refined with additional entities or safety events. Invalid configurations need to be accounted for during the search and the configuration space accordingly trimmed when relevant.

While the scene and configuration space are simplified abstractions of the considered use case, this suggests that focused testing around regions where UCAs occur results in a higher likelihood of safety hazards (35%) versus unguided approaches (20%). This should only occur after an initial, unbiased set of tests.

Scoping the search space

The initial configurations explored by the framework are key to scoping the search space, and performing a first selection of regions of interest where safety issues may reside. A number of experiment design strategies [22] have been proposed to that purpose. They inform on the relation between the different components of a system and their impact on its response, without exploring all combinations of values.

We compare three such strategies to understand how they might impact the identification of safety conditions: Full Factorial, Generalized Subset Design and Random. A full factorial design explores all possible combinations of components' values. The input domain for our search has been discretized into intervals 50cm apart to ensure it is finite. Generalized Subset Design (GSD) [3] is a fractional experimental design which supports components with more than two levels. It is parametrized by a reduction factor, such that GSD(**x**) generates one $\mathbf{x}^{th}$ of the experiments of a full factorial design. Random simply picks random values for each component and it is parametrized by the number of generated configurations, e.g. RAN(100) uses 100 runs.

Figure 7 presents the distribution of safety situations across all runs (horizontal axis) for each of the considered strategies (vertical axis). The strategies are ordered by the number of generated configurations, from the most to the least exhaustive ones. To allow for comparison, the RAN strategy was set to generate the same number of runs as the different GSD configurations.

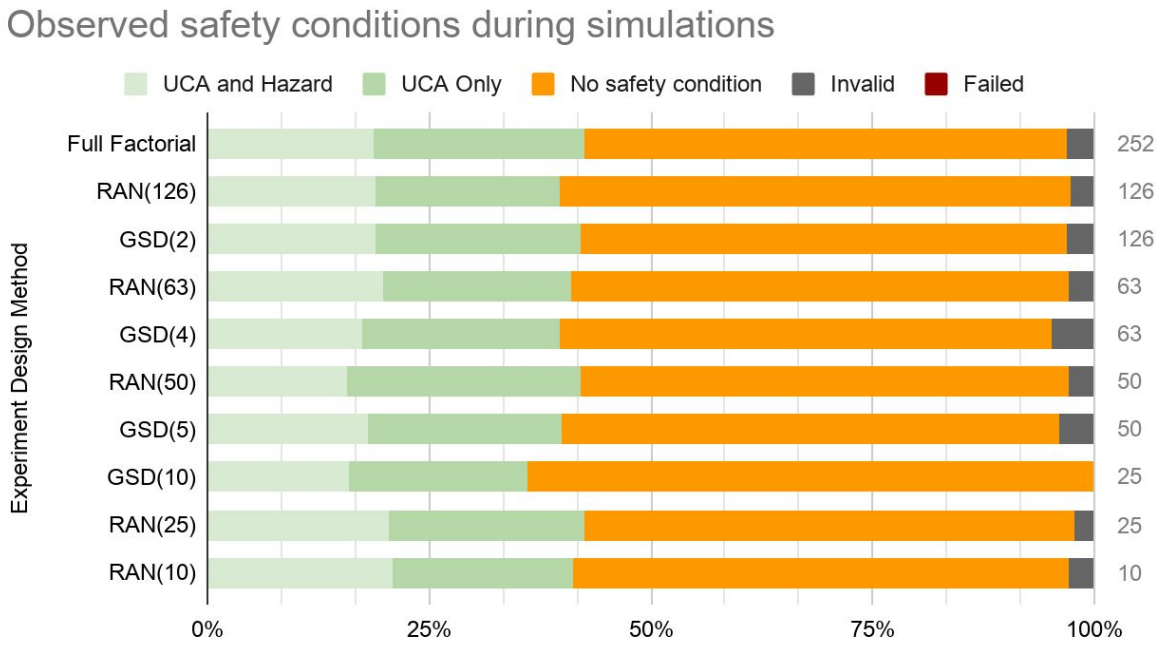

Figure 7. Comparison of the experimental design strategies on the distribution of safety situations detected across multiple runs.

From our initial observations, there is no significant difference between the considered strategies. The RAN strategy with only 10 runs produces similar results to the full exploration of the input space. Increasing the number of runs tends to reduce the variability in the results of the RAN strategy.

The lack of significant differences between strategies is the result of a small input domain (252 configurations) and response (3 significant levels). This is exemplified by the difference between the discrete input domain in this comparison and the continuous one considered in Figure 6. The discretization further resulted in a larger portion of the input space sitting on the outskirts of the cell, the region where hazard might occur.

Restricting the configuration space

Knowledge about the domain can further help improve the scope of the search. Considering we focus on safety events occurring around the cell, we restrict our domain to the innermost region (red spots) depicted in Figure 5. We compare the same strategies: Full factorial, GSD, and RAN. The results are presented in Figure 8, with a matching number of runs for the GSD and RAN strategies.

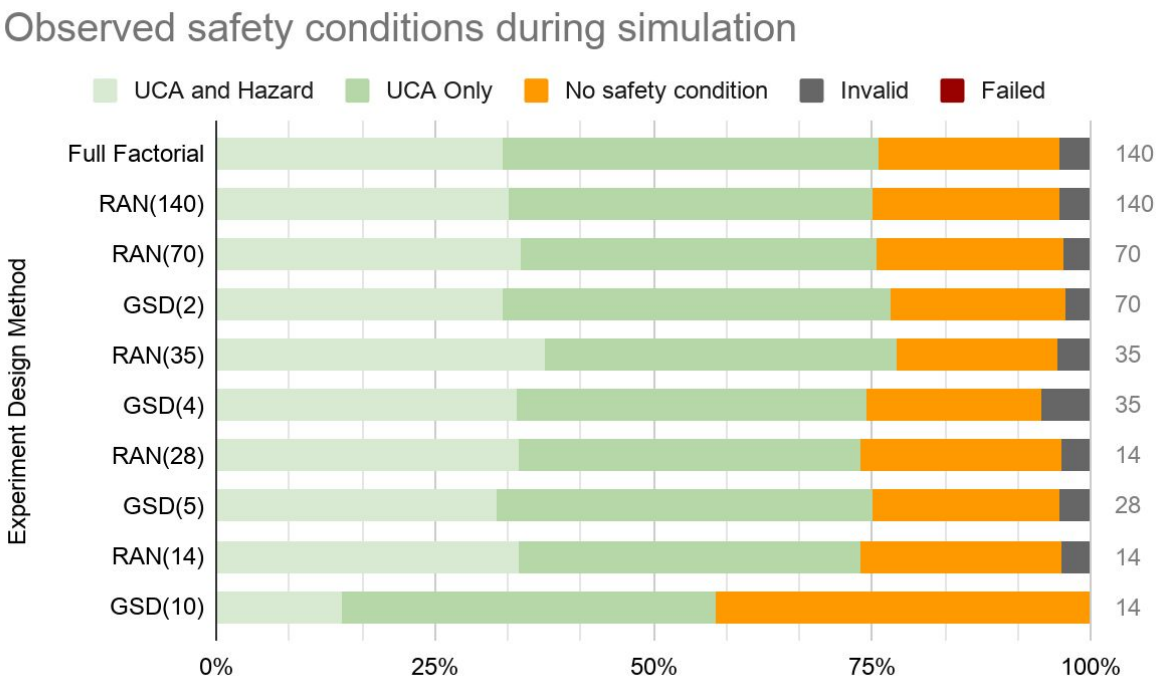

Figure 8. Comparison of the experimental design strategies under a restricted configuration space.

Compared to same randomised search over a wider domain (Figure 7), a more informed and restricted configuration space results in a greatly increased likelihood of observing safety conditions, including hazards. The overall number of configurations is almost divided by 2. However, there is still no significant distinction between the different strategies.

Conclusion

The safety analysis of a system provides a wealth of information on its hazardous behaviours, including undesirable situations which may lead to losses. We propose a safety-focused testing framework based on the artifacts produced by the analysis to understand which configurations of a system might lead to a hazard. Such configurations should be the focus of further testing. Our work focused on the development of and integration with the Digital Twin supporting the framework. Our initial evaluation explored a number of experiment designs over a small configuration space to identify the occurrences of safety events. While no significant differences were identified between the designs due to a limited scope, the results support our hypothesis. That is regions of the configuration space where accidents occur are included into regions of less hazardous situations. We plan to further improve the set of conditions which can be monitored by the Digital Twin, and explore alternative solutions to cover the configuration space and provide confidence in the overall safety of the system.

Bibliography

[1] ISO/DIS 10218-1 Robotics — Safety requirements for robot systems in an industrial environment — Part 1: Robots

[2] CSI: Cobot Project, https://www.york.ac.uk/assuring-autonomy/projects/csi-cobot/

[3] CARLA: Open-source simulator for autonomous driving research. https://carla.org/

[4] Bobka, Paul, et al. "Simulation platform to investigate safe operation of human-robot collaboration systems." *Procedia CIRP* 44 (2016): 187-192.

[5] Afzal, Afsoon, et al. "A study on challenges of testing robotic systems." *2020 IEEE 13th International Conference on Software Testing, Validation and Verification (ICST)*. IEEE, 2020.

[6] Huck, Tom P., Christoph Ledermann, and Torsten Kröger. "Simulation-based Testing for Early Safety-Validation of Robot Systems." *arXiv preprint arXiv:2011.10294* (2020).

[7] Alexander, R., H. Hawkins, and D. Rae. *Situation Coverage-A Coverage Criterion for Testing Autonomous Robots. Department of Computer Science, University of York*. Technical Report, 2015.

[8] Leveson, Nancy G., and John P. Thomas. "STPA handbook." *Cambridge, MA, USA* (2018).

[9] AAIP Body of Knowledge: 1.2.1 Considering human/machine interactions

[10] Falcone, Yliès, et al. "A taxonomy for classifying runtime verification tools." *International Conference on Runtime Verification*. Springer, Cham, 2018.

[11] D. Basin, F. Klaedtke, E. Zalinescu: "The MonPoly monitoring tool", RV-CuBES 2017.

[12] Hallé, Sylvain, and Raphael Khoury. "Event Stream Processing with BeepBeep 3." *RV-CuBES*. 2017.

[13] Emerson, E. Allen. "Temporal and modal logic." *Formal Models and Semantics*. Elsevier, 1990. 995-1072.

[14] https://github.com/douthwja01/CSI-cobotics

[16] Ghani, Kamran, and John A. Clark. "Automatic test data generation for multiple condition and MCDC coverage." *2009 Fourth International Conference on Software Engineering Advances*. IEEE, 2009.

[17] Helle, Philipp, Wladimir Schamai, and Carsten Strobel. "Testing of autonomous systems–Challenges and current state-of-the-art." *INCOSE International Symposium*. Vol. 26. No. 1. 2016.

[19] Hawkins, Heather, and Rob Alexander. "Situation Coverage Testing for a Simulated Autonomous Car--an Initial Case Study." *arXiv preprint arXiv:1911.06501* (2019).

[20] Gleirscher, Mario. "Hazard-based selection of test cases." *Proceedings of the 6th International Workshop on Automation of Software Test*. 2011.

[21] Grindal, Mats, Jeff Offutt, and Sten F. Andler. "Combination testing strategies: a survey." *Software Testing, Verification and Reliability* 15.3 (2005): 167-199.

[22] Fisher, Ronald Aylmer. "The design of experiments." *The design of experiments.* 7th Ed (1960).

[23] Surowiec, Izabella, et al. "Generalized subset designs in analytical chemistry." *Analytical Chemistry* 89.12 (2017): 6491-6497.

[24] Villani, Valeria, et al. "Survey on human–robot collaboration in industrial settings: Safety, intuitive interfaces and applications." *Mechatronics* 55 (2018): 248-266.
[25] Bauer, Andrea, Dirk Wollherr, and Martin Buss. "Human–robot collaboration: a survey." International Journal of Humanoid Robotics 5.01 (2008): 47-66.
[26] Zou, Xueyi, Rob Alexander, and John McDermid. "Testing method for multi-uav conflict resolution using agent-based simulation and multi-objective search." *Journal of Aerospace Information Systems* 13.5 (2016): 191-203.
[27] Norden, Justin, Matthew O'Kelly, and Aman Sinha. "Efficient black-box assessment of autonomous vehicle safety." *arXiv preprint arXiv:1912.03618* (2019).
[28] Guiochet, Jérémie, Mathilde Machin, and Hélène Waeselynck. "Safety-critical advanced robots: A survey." *Robotics and Autonomous Systems* 94 (2017): 43-52.