Proceedings Paper:
Stefanakos, Ioannis, Calinescu, Radu, Douthwaite, James A. et al. (2022) Safety Controller Synthesis for a Mobile Manufacturing Cobot. In: Software Engineering and Formal Methods 2022:Proceedings. Software Engineering and Formal Methods, 28-30 Sep 2022, Humboldt University. Lecture Notes in Computer Science. Springer, DEU, pp. 271-287.

https://doi.org/10.1007/978-3-031-17108-6_17

Safety Controller Synthesis for a Mobile Manufacturing Cobot

Ioannis Stefanakos[1 [0000−0003−3741−252X]], Radu Calinescu[1 [0000−0002−2678−9260]], James Douthwaite[2 [0000−0002−7149−0372]], Jonathan Aitken[2 [0000−0003−4204−4020]], and James Law[2 [0000−0002−4966−3556]]

[1] University of York, York, UK
{ioannis.stefanakos,radu.calinescu}@york.ac.uk
[2] University of Sheffield, Sheffield, UK
{j.douthwaite,jonathan.aitken,j.law}@sheffield.ac.uk

Abstract. We present a case study in which probabilistic model checking has been used to synthesise the correct-by-construction safety controller for a mobile collaborative robot (*cobot*) deployed in a prototype manufacturing cell alongside a human operator. The case study used an ICONSYS iAM-R mobile cobot responsible for the execution of a complex machining process comprising tasks requiring the use of multiple machines at different locations within the cell. Within this process, the role of the safety controller was to ensure that the cobot carried out its tasks and movements between task locations without harming the human operator responsible for its supervision and for performing additional tasks. The paper describes our generalisable approach to synthesising the mobile cobot safety controller, and its evaluation using a digital twin of our experimental manufacturing cell at the University of Sheffield Advanced Manufacturing Research Centre in the UK.

Keywords: safety controller · probabilistic model checking · collaborative robot · discrete-event controller synthesis · Markov chain

1 Introduction

Collaborative robotics "have the potential to revolutionise manufacturing" [4,21] and are expected to play a key role in Industry 4.0 [8,23,24]. However, the use of collaborative robots (*cobots*) in industry comes with significant safety concerns. Industrial cobots have been designed to directly operate and share workspaces with human operators, leading to the introduction of additional safety risks and the need to mitigate these [16,20,22].

Ensuring the safety of human operators (e.g., [3,25]) and improving the risk assessment during collaborative tasks with cobots (e.g., [18,19]) has been the primary focus of research in the area of human-robot collaboration in manufacturing environments. However, the effectiveness of this collaboration is still limited, e.g., due to restrictive cobot movement and frequent emergency stops resulting from safety concerns. The rapid technological advancements in robotics

introduce increasingly complex robotic systems that confine the ability to assess and mitigate risks. At the same time, to fully benefit from the use of cobots in industry there must be a trade-off between risk and performance. Thus, it is necessary to develop techniques that address these constraints and test their application in case studies with real collaborative robotic processes from the manufacturing domain.

In this paper, we present a case study involving the use of probabilistic modelling and verification for the synthesis of a safety controller for a collaborative robot used in a industrial research lab (Fig. 1). We carried out this case study at the University of Sheffield Advanced Manufacturing Research Centre (AMRC) [3] in the UK.

Fig. 1. Collaborative robot in action inside the AMRC Gear Center

The main contributions of our case study paper are:

- a new method for augmenting the activity diagram describing the ideal manufacturing process carried out in collaboration by the robot and a human operator with risk mitigation constructs based on the cobot safety operation modes recommended by the international standard ISO/TS 15066 [13];
- a new technique for mapping the mitigation-extended activity diagram of the industrial process to a stochastic model encoded in the high-level probabilistic modelling language of the established model checker PRISM [14];
- the use of probabilistic model checking to synthesise safety controllers that satisfy risk and cost-related constraints for the industrial process;
- the presentation of the end-to-end process we used to synthesise a safety controller for a real mobile cobot (i.e., an ICONSYS iAM-R[4]) deployed in an experimental manufacturing cell.

By considering a mobile cobot with its additional safety concerns due to the cobot moving between task locations, and by providing a detailed description of our safety controller synthesis, these contributions go beyond existing research papers on the safe use of cobots in manufacturing (e.g., [2,10,26]).

The rest of the paper is organised as follows. Section 2 introduces a manufacturing process comprising a number tasks, allocated between a cobot and a

[3] https://www.amrc.co.uk/

[4] The iAM-R is a mobile collaborative robot built on the MiR200 mobile robot base, and carrying a 3kg, 5kg, or 10kg 6-axis Universal Robot collaborative manipulator (the 10kg version being the focus this case study). The two are combined with an Iconsys modular interface, which provides programmable control over the platform. https://iam-r.iconsys.co.uk/

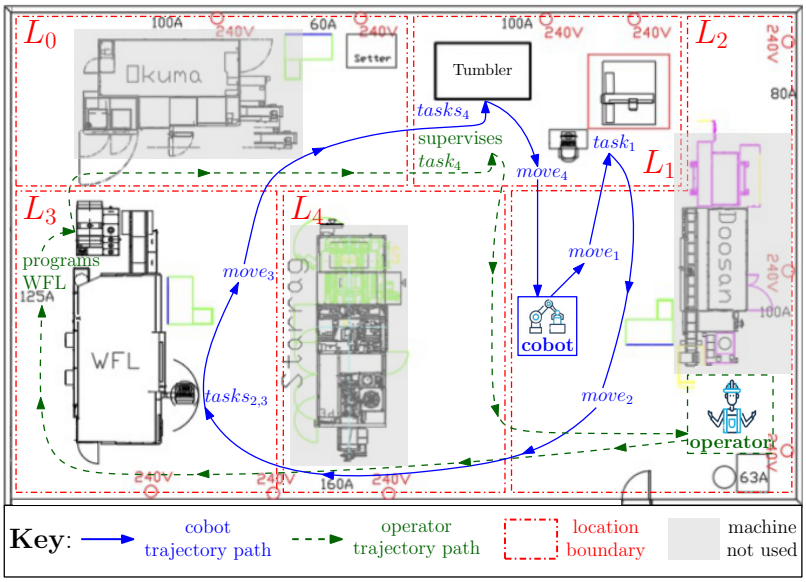

Fig. 2. Floorplan of the AMRC Gear Center facility

human operator. Section 3 presents our employed approach for the synthesis of safety controllers, ensuring the safety of the human operator in the introduced manufacturing process. Section 4 describes a two-pronged evaluation methodology for the synthesised safety-controllers, and Section 5 compares our solution to existing approaches. Finally, Section 6 summarises the benefits and limitations of our approach, and suggests directions for future work.

2 Manufacturing process

Fig 2 depicts the shop floor for the AMRC manufacturing process used in our case study. As shown in this diagram, the shop floor consists of a main area annotated with information about the location of tasks that need to be carried out both by the cobot and the operator, and their movement between locations.

The boundary of each location is specified by the red dashed lines, and the location identifier is given in the form of L_n where n is the location's number. The trajectory path of the cobot is depicted by the continuous blue line that also captures information about the various movements between locations and tasks that the cobot needs to perform. The movement and task identifiers appear in the form of $move_k$ and $task_l$ where k and l are numbers associated with each of the cobot's planned movements and tasks in an increasing order, respectively. The operator's trajectory path is depicted by the dashed green line, and brief descriptions of the operator's tasks can also be seen on the diagram along this

path. Several Computer Numerical Control (CNC) machines [5] can be observed on the shop floor. Two machines are used by the process: an WFL M30G 5-axis turn-mill with hobbing, shaping, and reciprocating broaching capability; and a Sharmic VRM225 abrasive tumbler for automated deburring and polishing. The shaded machines are unused, and represent fixed obstacles the cobot must avoid.

The starting location for both the cobot and the operator is L_2. When the process starts, the cobot travels to location L_1 to pick a component ($task_1$), and then moves to L_3 by traveling through L_4, as does the operator, to perform $task_2$ and $task_3$. These tasks involve the cobot in loading a billet to the WFL machine, and retrieving the finished component. At the same time, the human operator is responsible for any programming required at the WFL machine. When the tasks are finished, both the cobot and the operator move to location L_1 by travelling though L_0 to resume the next steps of the process. At L_1 the cobot performs $task_4$ and $task_5$ under the operator's supervision. These tasks involve the cobot depositing the finished component to the tumbler machine for deburring, and collecting it upon completion. In case something goes wrong, the operator will intervene to fix the issue. Following the previous tasks' successful completion, the cobot will carry out the final $task_6$, returning the polished component before travelling back to the starting location L_0, followed by the operator.

The process carried out by the cobot and a human operator is summarised by the "ideal" activity diagram from Fig. 3.[6] The diagram is consisted of two main branches associated with the cobot's and operator's movements and tasks on the left and right side, respectively. The fork notations lead to synchronisation points (*sync*) where the cobot and the operator need to confirm the completion of the previous tasks before they proceed to the next. Some tasks are performed independently by the cobot and some by the operator. These are standardised tasks that are always eventually performed even if there is a delay in place. On the contrary, there is another group of cobot's tasks that necessitates the operator's supervision as failure in such a task requires the operator's intervention to fix the issue, allowing the cobot to resume carrying out or re-initiating the task ($task_l\_retry$). The probabilities of success ($p_{t_l success}$) and failure ($p_{t_l failure}$) of these tasks and the resulting paths are denoted by decision nodes in the diagram. Finally, after the successful completion of all tasks (i.e., the cobot and the operator travel to their starting locations) the process can either be repeated or terminated.

Synthesising the safety controller as described in the remainder of the paper requires knowledge about the levels of risk and probabilities of hazards occurring during cobot movement and task execution in the proximity of the human operator. These quantities can be obtained experimentally by using sensors (i.e cameras, or wearables) combined with software tools that track human movements. This ensures the capability of the work cell to recognise situations that

[5] CNC machining: computerized manufacturing process in which pre-programmed software and code controls the movement of production equipment.

[6] This is an "ideal" activity diagram (and the starting point for our work) because it does not consider the hazards/risks associated with the process.

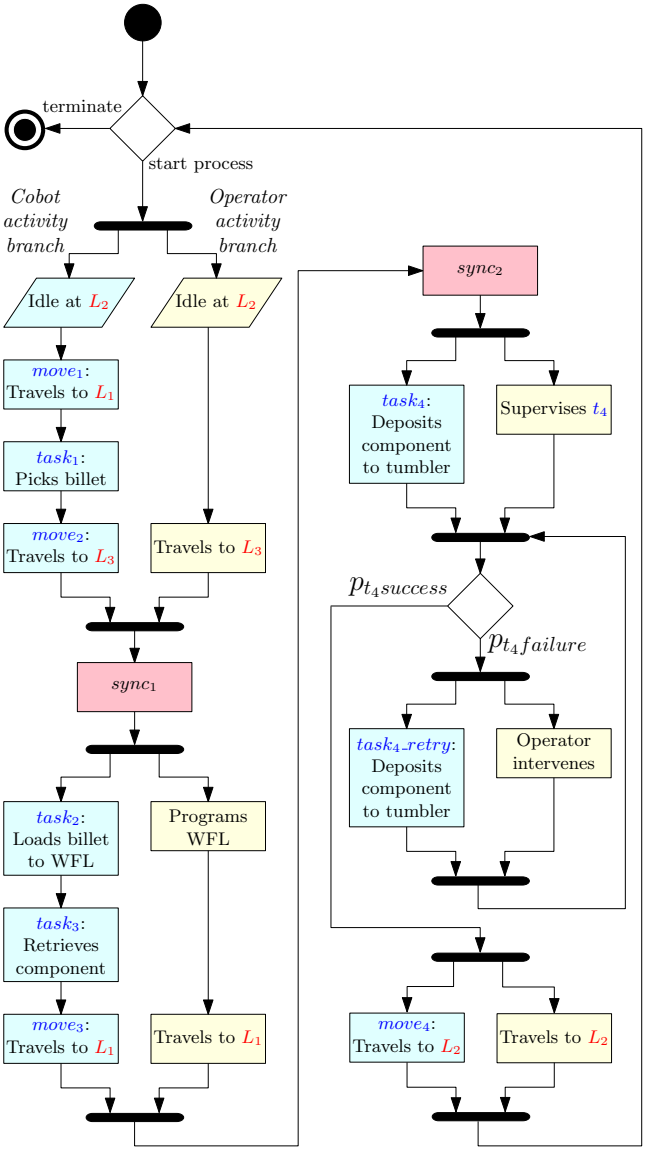

Fig. 3. Activity diagram of the collaborative process

can lead to undesired risks. The Assuring Autonomy International Programme (AAIP) project RECOLL [7] is an example of recent work that collected a considerable amount of data related to a group of operators. The analysis of the obtained data helped to determine the probability of a hazard's occurrence and

[7] https://www.york.ac.uk/assuring-autonomy/demonstrators/flexible-manufacturing/

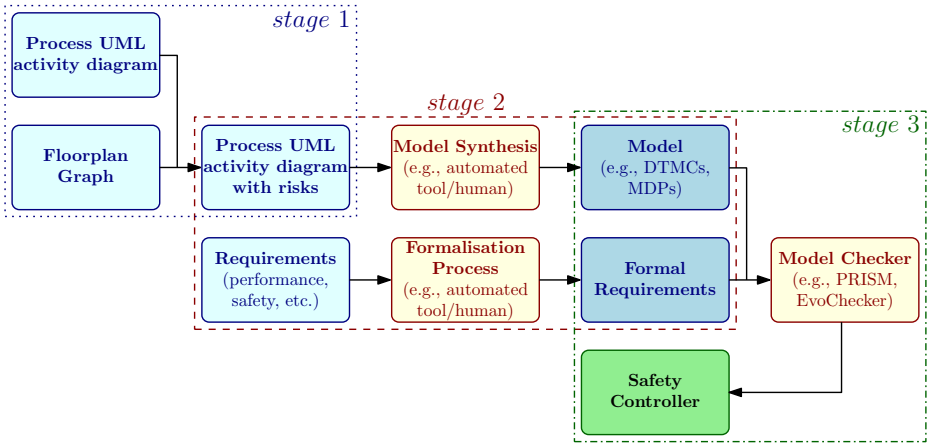

Fig. 4. High-level diagram of the employed approach.

its correlation with an incorrect task execution by the operator and/or an incorrect task planning by the system itself.

Given these quantities, which for our case study their values are assumed, we focused on the synthesis of a safety controller considering several mitigation actions for hazards associated with cobot tasks and movements. Note that not all tasks and movements that appear in the process have a high-risk factor. In our case study, only $move_2$, $move_3$, $move_4$ and $task_4$ in combination with operator's movements and tasks have been identified as actions that could lead to hazards.

A detailed description of the hazards, mitigation strategies and their associated risk and probability values can be found in the following section.

3 Safety controller synthesis

3.1 Overview of the approach

As shown in Fig. 4, our three-staged approach carries out the synthesis of safety controllers, and comprises a set of inputs, processing activities, and outputs used for the synthesis of safety controllers. During the first stage, an UML activity diagram and a floorplan graph are used to identify potential hazards in the process. This leads to the annotation of the UML diagram with mitigation strategies for the identified hazards (e.g., slowing down the cobot's speed to avoid collision with the human operator). Stage two of the approach synthesises a discrete-time Markov chain (DTMC) model based on the annotated UML diagram of the previous step, and formalises the requirements of the process. Finally, stage three employs probabilistic model checking and applies an exhaustive search over the discretised parameter space of the model to synthesise combinations of values for the DTMC parameters that correspond to requirements-compliant safety controllers for the industrial process.

3.2 Stage 1: Hazard identification

In this first stage of the approach, the UML activity diagram from Fig. 3 and the floorplan from Fig. 2 are combined in order to identify hazards in the process. This is currently a manual activity in which the user needs to assess where hazards may appear. The output of this combined analysis is an UML activity diagram annotated with probabilities of hazards occurring when the cobot is performing a high-risk movement or task (i.e., an action that may affect the human operator who is or will be in close proximity to the cobot), and mitigation actions which are also selected based on probabilities. An example of such diagram can be seen in Fig. 5, depicting the last part of our case study's process, where $task_4 retry$ and $move_4$ are replaced by the dashed rectangles that contain the hazard mitigation actions.

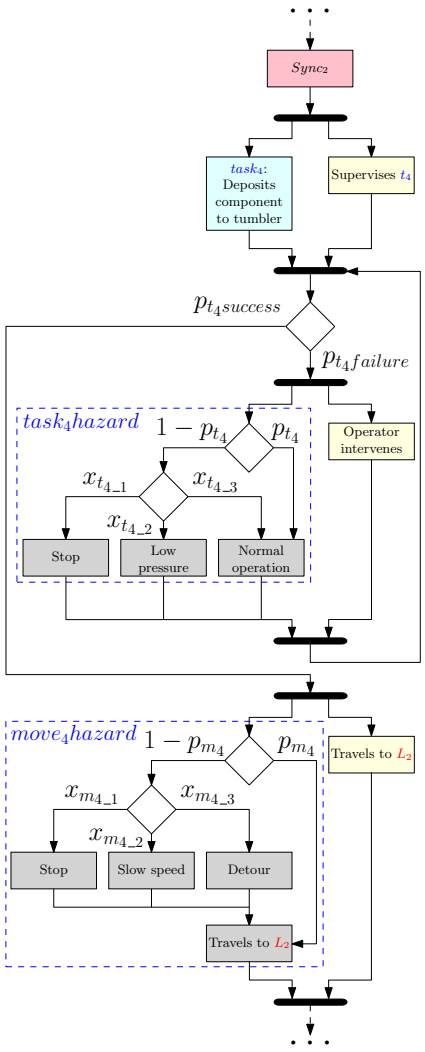

Fig. 5. Activity diagram of the collaborative process with risks

These actions are defined following the specifications of the standard ISO/TS 15066 [13] which describes four main techniques for collaborative operation: a) safety-rated monitored stop, b) hand-guiding, c) speed and separation monitoring, and d) power and force limiting. All these apply in our collaborative process, except the hand-guiding.

In case of cobot's failure to complete $task_4$ (Fig. 5), the operator must intervene to correct the issue, so that the cobot can resume performing the task. This introduces a high-risk situation as the operator will be in close proximity to the cobot, and any unpredicted movement could potentially result in injury. To prevent this from happening, a series of hazard identification and mitigation steps are introduced under $task_4 hazard$. Specifically, there is a p_{t_4} probability that a hazard will not occur, leading to the cobot's normal operation and a $1 - p_{t_4}$ probability that a hazard will occur, leading to the following mitigation actions. With probability of $x_{t_{4\_1}}$ the cobot will pause its operation, with $x_{t_{4\_2}}$ will decrease the applied pressure, and with

$x_{t_{4\_3}}$ will resume its normal operation as the risk has not exceeded the given threshold.

Following the successful completion of $task_4$, the cobot will try to move to its initial location ($move_4$) to terminate or repeat the process, as does the operator. A potential delay by the operator could cause a collision accident, which can be avoided by the correct prevention mechanisms defined under $move_4 hazard$. These indicate that with a probability of p_{m_4} a hazard will not occur, leading to the cobot's planned movement towards location L_2, and with a $1 - p_{m_4}$ probability that a hazard will occur, triggering the following mitigation actions. With probability of $x_{m_{4\_1}}$ the cobot will pause its movement, with $x_{m_{4\_2}}$ will slow down, and with $x_{m_{4\_3}}$ will find the next available route towards L_2.

3.3 Stage 2: Stochastic modelling

During the second stage of the employed approach, we built a DTMC derived from the annotated UML diagram of Fig. 5. We specified this DTMC in the high-level modelling language of the probabilistic model checker PRISM [14], which represents a given system as the parallel composition of a set of *modules*. The state of a *module* is defined by a set of finite-range local variables, and its state transitions are encoded by probabilistic guarded commands that modify these variables, and have the general form:

$$[\,]\; g \;\rightarrow\; \lambda_1 \;:\; u_1 \;+\; \lambda_2 \;:\; u_2 \;+\; \ldots \;+\; \lambda_n \;:\; u_n; \tag{1}$$

where *guard* g is a boolean expression over all model variables. If the guard evaluates to true, the arithmetic expression λ_i, $1 \leq i \leq n$ gives the probability with which the u_i change of the module variables occurs. Commands can be (optionally) labelled with *actions* that are placed between the square brackets, e.g., [*sync*], causing all modules comprising commands with the same action to synchronise (i.e., perform one of these commands simultaneously). For more details regarding the PRISM modelling language, we refer the reader to the PRISM manual, available at `http://www.prismmodelchecker.org/manual`.

Part of the synthesised DTMC, modelling the collaborative process, can be seen in Fig. 6. Line 4 declares the probability of a hazard occurring during $move_2$, and lines 5–7 the probabilities of each of the mitigation actions. The process module (lines 11–23) contains the main functionality of the model where each step of the process is defined. The reward structures, capturing information about states in the model (e.g., a risk of 0 is associated with the mitigation action of state c=4), are located between lines 27–42. The complete model of our case study's process can be found in our GitHub repository[8].

The process requirements are formally expressed in probabilistic computation tree logic (PCTL) [12] extended with rewards [1]. The syntax of PCTL is as follows:

[8] `https://github.com/CSI-Cobot/CSI-artefacts`

```
dtmc

const double p_m2 ;  //hazard probability during move_2
const double x_m2_1 ;    //probability of stopping
const double x_m2_2 ;    //probability of reducing speed
const double x_m2_3 ;    //probability of following an alternative route
...
module process
  c : [0..30] init 0;

  [] c=0 -> 1:(c'=1);   //cobot idle at L0
  [] c=1 -> 1:(c'=2);   //cobot performs task_1 at L1
  [] c=2 -> p_m2 :(c'=3) + (1- p_m2 ):(c'=7); // hazard prob. for move_2
  [] c=3 -> x_m2_1 :(c'=4) + x_m2_2 :(c'=5) + x_m2_3 :(c'=6); //mitigation
  [] c=4 -> 1:(c'=7);   // cobot stops
  ...
  [] c=28 -> 1:(c'=30); // cobot slows speed
  [] c=29 -> 1:(c'=30); // cobot identifies alternative route
  [] c=30 -> 1:(c'=0);  // cobot travels to L2
endmodule

rewards "delay"
  c=4  : 20;    // base - stop
  c=5  : 13;    // base - reduce speed
  c=6  : 15;    // base - detour
  ...
  c=22 : 4; // arm  - low pressure
endrewards

rewards "risk"
  c=4  : 0; // base - stop
  c=5  : 2; // base - reduce speed
  c=6  : 1; // base - detour
  ...
  c=22 : 1; // arm  - low pressure
endrewards
```

Fig. 6. DTMC model representation of the process with hazard mitigation actions.

$$\begin{aligned} \Phi \ &::= true \mid a \mid \neg\, \Phi \mid \Phi \ \wedge \ \Phi \mid P_{\bowtie p}[\phi] \\ \phi \ &::= \ X\ \Phi \mid \Phi\ U^{\leq k}\ \Phi \end{aligned} \tag{2}$$

and cost/reward state formulae are defined by the grammar:

$$R_{\bowtie r}[C^{\leq k}] \mid R_{\bowtie r}[I^{=k}] \mid R_{\bowtie r}[F\ \Phi] \tag{3}$$

where $k \in \mathbb{N} \cup \{\infty\}$, $p \in [0,1]$ and $r \in R_{\geq 0}$ are probability/reward bounds (that can be replaced with '=?' if the computation of the actual probability or reward is required), and Φ and ϕ are state and path formulae, respectively. The definition of PCTL semantics is beyond the scope of this paper; details are available from [5,15].

We use the above PCTL syntax to specify the properties of the DTMC model. In particular, we extract information about the delay introduced in the process by the mitigation actions, and the risk of a hazard occurring:

$$\begin{aligned} R^{delay}_{=?}[\ F\ \text{“}done\text{”}\ ] \\ R^{risk}_{=?}[\ F\ \text{“}done\text{”}\ ] \end{aligned} \tag{4}$$

where “*done*” is a label referring to the end of the process.

3.4 Stage 3: Synthesis

In the third and final stage, an exhaustive search is performed over the discretised parameter space of the output DTMC model, combined with the set of formal requirements to obtain the Pareto-optimal controller configurations. The parameters of the DTMC model are consisted by a) the probabilities obtained from the manufacturing process that cannot be modified (e.g., the probability that a hazard will occur during the $move_3$ action of the cobot), and b) the probabilities associated with mitigation actions for which suitable values must be obtained in order to have an optimal trade-off between risk and performance (e.g., the probability $x_{t_{4\_1}}$ that a stop will be triggered to avoid a potential hazard during $task_4$).

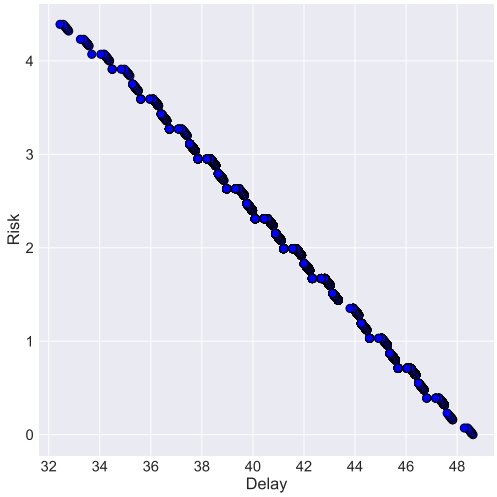

Fig. 7. Pareto-optimal controller configurations

Regarding the second group, we go through all the combinations of controller parameters {x_m**i**_1, x_m**i**_2, x_m**i**_3}, similarly for {x_t**i**_1, x_t**i**_2, x_t**i**_3}, between {0.0, 0.0, 1.0} and {1.0, 0.0, 0.0} with step 0.2, where **i** refers to the number of movement or task associated with the mitigation action. This results to a total of 456,976 combinations of parameter values, i.e. model instantiations, whose analysis using the model checker PRISM required just under one hour on a 2 GHz Quad-Core Intel Core i5 MacBook Pro computer with 32 GB of memory, and which are used in identifying optimal trade-offs between the requirements of the process. Section 4.1 provides an example on how these optimal trade-offs can be obtained.

4 Evaluation

To evaluate the safety-controller synthesis approach used in our case study, we employed a two-pronged evaluation methodology. First, we used our approach to synthesise a set of safety controller instantiations that meet strict safety constraints and achieve optimal trade-offs between the efficiency of the manufacturing process and the level of residual risk. Second, we developed a digital twin of the manufacturing process and used it to trial one of these safety controller instantiations. These results from these stages of our two-pronged evaluation are described in Sections 4.1 and 4.2.

Fig. 8. The process DT constructed as a test-bed within the DTF. The DT presents a faithful reconstruction of the real-world process used for the evaluation of the proposed safety-controller.

4.1 Generation of safety controller instantiations

Fig. 7 depicts the Pareto-front of optimal controller configurations. Using the data collected during stage 3 of the approach (Section 3.4) we are able to identify which of these model instantiations provide the most optimal trade-offs between risk and performance. This information is of great use in scenarios where it is necessary to improve the overall performance of the process, while ensuring that risks are below the specified threshold levels. We obtained these results by analysing the output data from PRISM's experiments using python scripts. This process can also be automated by using tools such as EvoChecker [9] that provides more efficient search algorithms.

4.2 Evaluation on a digital twin

Digital twin description The digital twin used to evaluate the safety controller was developed within the Digital Twin Framework (DTF), in associated works [6], as part of the Confident Safety Integration for Collaborative Robotics (CSI:Cobot) project. The DTF is versatile sandbox for the development and testing of safety-critical systems [9].

The DTF provides the kinematic, communication and data infrastructure necessary to deploy digital twins on real world systems. Using its integrated APIs for MATLAB ® and the Robotic Operating System (ROS), commands issued inside the DTF may be exchanged with the physical system and demonstrated as a real-time response.

[9] For further information on the CSI:project please visit the project's website at: `https://www.sheffield.ac.uk/sheffieldrobotics/about/csi-cobots/csi-project`, and our associated repository: `https://github.com/CSI-Cobot/CSI-artefacts`

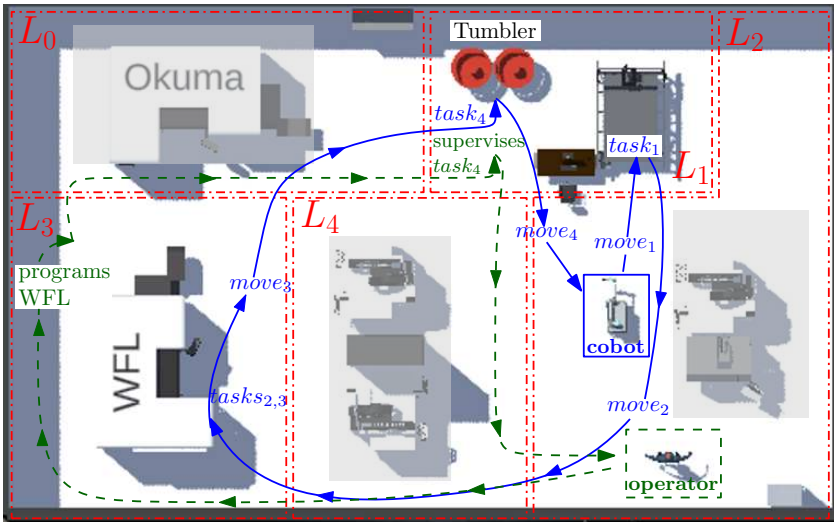

Fig. 9. Transferal of the floor plan from Fig. 2 to the digital environment.

Process Twin In this study, the DTF was used create a faithful recreation of the process as (seen in Fig. 8), from the real-world process description as shown in Fig. 9. In associated works [6,10], we describe the inclusion of entity-modules and behaviour-modules as a system of *actors* and *abstract-actors* respectively. The distinction being those that are *embodied* in the real-world and those that are not. We define A to be this set of actors, but also as the set of communication nodes. Actor $n \in A$ is then able to communicate with other actors through a static interface $I_n = (I_n, O_n)$. Here $I_n = [i_n^1, i_n^2, ..., i_n^k]$ and $O_n = [o_n^1, o_n^2, ..., o_n^h]$ define n's set of input (subscription) and output (publication) channels respectively. Each actor n is modelled a distinct state machine that listens on channels I_n and responds on channels O_n where $I_n, O_n \subseteq X$ where X is the set of all available channels.

The process controller is modelled as *abstract-actor* $n_{pc} \in A$, implementing the interface $I_{n_{pc}} = (I_{n_{pc}}, O_{n_{pc}})$. n_{pc} communicates with the other digital-twin systems via this interface in order to interact with the *iAMR* mobile robot, the human *operator* and other sensors in the environment. The nominal process procedure (seen in Fig. 3) is defined by the state machine of n_{pc} that responds to process updates from process members $n_{i\in 1:k} \subset A$ received on channels $I_{n_{pc}}$ with responses commands $O_{n_{pc}}$.

Safety controller Similar to the process controller, the safety controller n_{sc} is introduced as a behaviour module and *abstract-actor*. Here $n_{sc} \in A$, and $I_{sc} = (I_{n_{sc}}, O_{n_{sc}})$ defines its interface. n_{sc} may communicate with both the actor and process controller such that $n_{i\in 1:k} \in A_{sc} \subset A$ and $n_{pc} \in A_{sc}$. To allow n_{sc} to intervene with the nominal process managed by n_{pc}, n_{sc} is introduced as

an independent state-machine able to observe channels $I_{n_{sc}}$ and enact changes to the process such that $I_{n_{pc}} \subseteq O_{n_{sc}}$.

For example, upon reception of sensor data received on $I_{n_{sc}}$ indicating that the iAMR and operator are in the same process region. n_{sc} issues a request to n_{pc} to request a change in safety mode for the robot in that region. This command is received on $I_{n_{pc}}$ and issued on $O_{n_{pc}}$, as a request to process member $n_{k=2}$, for a new safety mode-*reduced speed*. $n_{k=2}$ defines the iAMR mobile robot, which in turn enacts a response to this request by changing its active safety mode to *reduced speed*.

Evidencing Safety Whilst active, the DTF provides connection to the robots via other APIs (such as ROS) and data storage (via SQL) amongst other fundamental DT services necessary for this study. Information broadcast on channels $X = (I_{n=1:k}, O_{n=1:k})$ are timestamped and recorded in a process database. This allows key process and safety signals to be recovered and analysed alongside additional ground-truth data (i.e. absolute positions and decision historicity).

In the following example, the machine-tending DT is used to demonstrate the application of the safety controller to a realistic process interface. The process is implemented as an*abstract-actor* and state-machine whose process is defined by the work-activities (shown in Fig. 3). Work-requests are issued by the process controller, to human and robot *actors* (digital twins) as each step *WState* is completed. The example centers around the collaborative preparation of a component, which must be moved between machines (CNC & WFL) by the robot before being deposited in a tumbler for deburring. An overview of the complete process can be seen in Fig. 9.

The safety controller is implemented similarly, and observes the current process step which is reported as the *Action* of the human and robot by the process controller. The safety controller however, utilises a camera sensor module in order to observe both the *region* (as *L0, l1,.. L4 etc.*) of the robot and human operator and their approximate separation distance in order to evaluate a mitigation action proportional to the hazard (see Section 3.2). The state of the process controller, safety controller and all communications are logged upon execution of the example scenario and recovered from the process database post-execution.

Table 1 presents the scenario event-sequence reported by the DTF. The process is shown to begin with both the human operator and robot in their start positions (see Fig. 9). The human operator initially proceeds to work at the WFL before completing a sequence of work actions. At 19.45.41s the operator's work is completed and proceeds to the collaborative work cell *AtTumbler*. Whilst this occurs, the robot moves from its *AtStart* location to collect a work component from the *AtCnC* location and continues to the *AtWFL* where the component is processed. The component is then collected and taken to the collaborative *AtTumbler* location. The robot arrives first and deposits the component in the tumbler. The human operator later joins at 19.46.00s, interrupting the cobot by violating the safety controller's *close* condition, which should cause a safety-stop.

At 19.46.14, before the robot begins to work, the safety controller issues a *SafetyReq(Stopped)*. A response from the robot *SafetyRes* is immediately issued

Table 1. The process event time-series during the example case-study.

Time	Type	Robot Location	Robot Action	Robot WState	Human Location	Human Action	Human WState
19:41:51	Process	Start	Idle	Incomplete	Start	Moving	Incomplete
19:42:00	Process	AtCNC	Done	Incomplete	Start	Moving	Incomplete
19:42:08	Process	AtCNC	Done	Incomplete	AtWFL	Done	Incomplete
19:42:15	Process	AtCNC	Idle	Incomplete	AtWFL	Done	Incomplete
19:42:23	Process	AtCNC	Idle	Incomplete	AtWFL	Idle	Incomplete
19:42:23	Process	AtCNC	Idle	Incomplete	AtWFL	Working	Incomplete
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮
19:44:46	Process	AtWFL	Moving	Complete	AtWFL	Done	Incomplete
19:45:01	Process	AtWFL	Moving	Complete	AtWFL	Idle	Incomplete
19:45:01	Process	AtWFL	Moving	Complete	AtWFL	Working	Incomplete
19:45:15	Process	AtTumbler	Done	Incomplete	AtWFL	Working	Incomplete
19:45:30	Process	AtTumbler	Idle	Incomplete	AtWFL	Working	Incomplete
19:45:41	Process	AtTumbler	Idle	Incomplete	AtWFL	Done	Incomplete
19:45:41	Process	AtTumbler	Idle	Incomplete	AtWFL	Done	Complete
19:46:00	Process	AtTumbler	Idle	Incomplete	AtTumbler	Done	Incomplete
19:46:14	SafetyReq		Stopped				
19:46:14	SafetyRes		Stopped				
19:46:14	Process	AtTumbler	Idle	Complete	AtTumbler	Done	Incomplete
19:46:15	Process	AtTumbler	Stopped	Complete	AtTumbler	Idle	Incomplete
19:46:15	Process	AtTumbler	Stopped	Complete	AtTumbler	Working	Incomplete
19:46:30	Process	AtTumbler	Stopped	Complete	AtTumbler	Done	Incomplete
19:46:30	Process	AtTumbler	Stopped	Complete	AtTumbler	Done	Complete
19:46:37	SafetyReq		Nominal				
19:46:37	SafetyRes		Nominal				
19:46:47	Process	AtTumbler	Moving	Complete	Start	Done	Incomplete
19:46:56	Process	Start	Done	Incomplete	Start	Done	Incomplete

and at 19.46.15s the robot enacts *Action(Stopped)*. The human operator continues to inspect the component in *close* proximity of the robot, until at 19.46.30s, the his work (*WState*) is completed and he moves away from *AtTumbler*. Once the operator complies with the *close* condition again, the safety controller successfully issues a new *SafetyReq(Nominal)* to resume normal operation. The robot and operator then proceed to return to the start locations and the example process is completed. A complete video of the sequence can be found on the CSI:Cobot repository [10].

[10] Additional study data and materials can be found on the CSI:Cobot repository: `https://github.com/CSI-Cobot/CSI-artefacts`.

5 Related work

Mobile collaborative robots have only emerged as a viable technology in recent years (e.g., [7,17]) and, to the best of our knowledge, no formal approaches have been used so far for the synthesis of their safety controllers.

For static cobots, an approach for safety-controller synthesis has been developed in the earlier stages of our CSI:Cobot project. Like the solution presented in our paper, this approach [10,11] uses stochastic models to capture the interactions between the cobot and the human operator, and probabilistic model checking to analyse these models. However, unlike the approach we employed for the case study presented in this paper, the previous CSI:Cobot solution from [10,11] does not consider the floorplan of the shop floor, nor the risks associated with the cobot travelling between different shop floor locations and the mitigations for these risks. Furthermore, the techniques used to augment the activity diagram of the collaborative manufacturing process risks and mitigations, and to derive the process DTMC from the augmented activity diagram represent new contributions of this paper, as does the ICONSYS iAM-R case study.

These differences also distinguish our work from other approaches to using formal verification for the safety analysis of cobot-human interaction, e.g., [2,25].

6 Conclusion

We presented a case study in which probabilistic model checking was used to synthesise Pareto-optimal safety controller configurations for a mobile cobot from an experimental manufacturing cell. In future work, we intend to deploy the safety controller tested in the digital twin on the actual iAM-R mobile cobot for validation in our experimental manufacturing cell.

We envisage that the multi-stage approach employed for this purpose can be generalised to a broad range of mobile-cobot scenarios, and that many of its activities can be automated. Both of these and assessing the scalability of the approach represent additional directions of future work for our project. Finally, when automating the synthesis of the Pareto-optimal controller configurations, we plan to replace the exhaustive search through the discretised controller configuration space with the much more efficient metaheuristic search provided by the EvoChecker probabilistic model synthesis tool [9].

Acknowledgements: This research has received funding from the Assuring Autonomy International Programme (AAIP grant CSI: Cobot), a partnership between Lloyd's Register Foundation and the University of York, and from the UKRI project EP/V026747/1 "Trustworthy Autonomous Systems Node in Resilience". We are grateful to our industrial collaborator for the gained insights into manufacturing cobots and to the AMRC for allowing us to use the iAM-R mobile collaborative robot, to implement the physical robotic process and evaluate the synthesised safety controller into their facilities.

References

1. Andova, S., Hermanns, H., Katoen, J.: Discrete-time rewards model-checked. In: Formal Modeling and Analysis of Timed Systems. Lecture Notes in Computer Science, vol. 2791, pp. 88–104. Springer Berlin Heidelberg (2003). https://doi.org/10.1007/978-3-540-40903-8_8
2. Askarpour, M., Mandrioli, D., Rossi, M., Vicentini, F.: SAFER-HRC: Safety analysis through formal verification in human-robot collaboration. In: Computer Safety, Reliability, and Security. pp. 283–295. Springer International Publishing (2016). https://doi.org/10.1007/978-3-319-45477-1_22
3. Bi, Z., Luo, C., Miao, Z., Zhang, B., Zhang, W., Wang, L.: Safety assurance mechanisms of collaborative robotic systems in manufacturing. Robotics and Computer-Integrated Manufacturing **67**, 102022 (2021). https://doi.org/10.1016/j.rcim.2020.102022
4. Cherubini, A., Passama, R., Crosnier, A., Lasnier, A., Fraisse, P.: Collaborative manufacturing with physical human–robot interaction. Robotics and Computer-Integrated Manufacturing **40**, 1–13 (2016). https://doi.org/10.1016/j.rcim.2015.12.007
5. Ciesinski, F., Größer, M.: On Probabilistic Computation Tree Logic, pp. 147–188. Springer Berlin Heidelberg (2004). https://doi.org/10.1007/978-3-540-24611-4_5
6. Douthwaite, J., Lesage, B., Gleirscher, M., Calinescu, R., Aitken, J.M., Alexander, R., Law, J.: A modular digital twinning framework for safety assurance of collaborative robotics. Frontiers in Robotics and AI **8** (2021). https://doi.org/10.3389/frobt.2021.758099
7. D'Souza, F., Costa, J., Pires, J.N.: Development of a solution for adding a collaborative robot to an industrial agv. Industrial Robot **47**(5), 723–735 (2020)
8. El Zaatari, S., Marei, M., Li, W., Usman, Z.: Cobot programming for collaborative industrial tasks: An overview. Robotics and Autonomous Systems **116**, 162–180 (2019). https://doi.org/10.1016/j.robot.2019.03.003
9. Gerasimou, S., Tamburrelli, G., Calinescu, R.: Search-based synthesis of probabilistic models for quality-of-service software engineering. In: 30th IEEE/ACM International Conference on Automated Software Engineering. pp. 319–330 (2015). https://doi.org/10.1109/ASE.2015.22
10. Gleirscher, M., Calinescu, R., Douthwaite, J., Lesage, B., Paterson, C., Aitken, J.M., Alexander, R., Law, J.: Verified synthesis of optimal safety controllers for human-robot collaboration. Science of Computer Programming **218**, 102809 (2022). https://doi.org/10.1016/j.scico.2022.102809
11. Gleirscher, M., Calinescu, R.: Safety controller synthesis for collaborative robots. In: 25th International Conference on Engineering of Complex Computer Systems (ICECCS). pp. 83–92 (2020)
12. Hansson, H., Jonsson, B.: A logic for reasoning about time and reliability. Formal Aspects Computing **6**(5), 512–535 (1994). https://doi.org/10.1007/BF01211866
13. ISO/TS 15066: Robots and robotic devices – Collaborative robots. Standard, Robotic Industries Association (RIA) (2016), `https://www.iso.org/standard/62996.html`
14. Kwiatkowska, M., Norman, G., Parker, D.: Prism 4.0: Verification of probabilistic real-time systems. In: Computer Aided Verification. pp. 585–591. Springer Berlin Heidelberg (2011). https://doi.org/10.1007/978-3-642-22110-1_47
15. Kwiatkowska, M.Z., Norman, G., Parker, D.: Stochastic model checking. In: Formal Methods for Performance Evaluation, 7th International School on Formal Methods for the Design of Computer, Communication, and Software Systems.

Lecture Notes in Computer Science, vol. 4486, pp. 220–270. Springer (2007). https://doi.org/10.1007/978-3-540-72522-0_6
16. Lee, K., Shin, J., Lim, J.Y.: Critical hazard factors in the risk assessments of industrial robots: Causal analysis and case studies. Safety and Health at Work **12**(4), 496–504 (2021). https://doi.org/10.1016/j.shaw.2021.07.010
17. Levratti, A., Riggio, G., Fantuzzi, C., De Vuono, A., Secchi, C.: TIREBOT: A collaborative robot for the tire workshop. Robotics and Computer-Integrated Manufacturing **57**, 129–137 (2019)
18. Liu, Z., Wang, X., Cai, Y., Xu, W., Liu, Q., Zhou, Z., Pham, D.T.: Dynamic risk assessment and active response strategy for industrial human-robot collaboration. Computers & Industrial Engineering **141**, 106302 (2020). https://doi.org/10.1016/j.cie.2020.106302
19. Marvel, J.A., Falco, J., Marstio, I.: Characterizing task-based human–robot collaboration safety in manufacturing. IEEE Transactions on Systems, Man, and Cybernetics: Systems **45**(2), 260–275 (2015). https://doi.org/10.1109/TSMC.2014.2337275
20. Matthias, B., Kock, S., Jerregard, H., Kallman, M., Lundberg, I., Mellander, R.: Safety of collaborative industrial robots: Certification possibilities for a collaborative assembly robot concept. In: 2011 IEEE International Symposium on Assembly and Manufacturing (ISAM). pp. 1–6 (2011). https://doi.org/10.1109/ISAM.2011.5942307
21. Maurice, P., Padois, V., Measson, Y., Bidaud, P.: Human-oriented design of collaborative robots. International Journal of Industrial Ergonomics **57**, 88–102 (2017). https://doi.org/10.1016/j.ergon.2016.11.011
22. Murashov, V., Hearl, F., Howard, J.: Working safely with robot workers: Recommendations for the new workplace. Journal of Occupational and Environmental Hygiene **13**(3), D61–D71 (2016). https://doi.org/10.1080/15459624.2015.1116700
23. Rüßmann, M., Lorenz, M., Gerbert, P., Waldner, M., Justus, J., Engel, P., Harnisch, M.: Industry 4.0: The future of productivity and growth in manufacturing industries. Boston consulting group **9**(1), 54–89 (2015)
24. Sherwani, F., Asad, M.M., Ibrahim, B.: Collaborative robots and industrial revolution 4.0 (IR 4.0). In: 2020 International Conference on Emerging Trends in Smart Technologies. pp. 1–5 (2020). https://doi.org/10.1109/ICETST49965.2020.9080724
25. Vicentini, F., Askarpour, M., Rossi, M.G., Mandrioli, D.: Safety assessment of collaborative robotics through automated formal verification. IEEE Transactions on Robotics **36**(1), 42–61 (2020). https://doi.org/10.1109/TRO.2019.2937471
26. Zanchettin, A.M., Rocco, P.: Path-consistent safety in mixed human-robot collaborative manufacturing environments. In: IEEE/RSJ International Conference on Intelligent Robots and Systems. pp. 1131–1136 (2013). https://doi.org/10.1109/IROS.2013.6696492