Modeling Operator Behavior in the Safety Analysis of Collaborative Robotic Applications

Mehrnoosh Askarpour[1(✉)], Dino Mandrioli[1], Matteo Rossi[1], and Federico Vicentini[2]

[1] DEIB, Politecnico di Milano, Milan, Italy
{mehrnoosh.askarpour,dino.mandrioli,matteo.rossi}@polimi.it
[2] CNR, ITIA, Milan, Italy
federico.vicentini@cnr.itia.it

Abstract. Human-Robot Collaboration is increasingly prominent in people's lives and in the industrial domain, for example in manufacturing applications. The close proximity and frequent physical contacts between humans and robots in such applications make guaranteeing suitable levels of safety for human operators of the utmost importance. Formal verification techniques can help in this regard through the exhaustive exploration of system models, which can identify unwanted situations early in the development process. This work extends our SAFER-HRC methodology with a rich non-deterministic formal model of operator behaviors, which captures the hazardous situations resulting from human errors. The model allows safety engineers to refine their designs until all plausible erroneous behaviors are considered and mitigated.

Keywords: Cognitive models · Formal verification · Task-analytic models · Human errors · Safety analysis · Human-robot collaboration

1 Introduction

Collaborative robot applications necessitate close proximity and possible physical contacts between operators and robots, due to the intrinsic nature of the activities they execute together. Thus, a central requirement in the design of this kind of applications is an assessment that identifies hazards and eliminates or mitigate risks. While informal assessment techniques such as HAZOP [28] might overlook some hazards; formal verification techniques are more reliable as they exhaustively check whether a system—modeled through a mathematical notation—satisfies required properties (e.g., safety properties), or has incompletenesses and inconsistencies [5]. However, modeling collaborative applications requires to consider the human behavior and its non-determinism caused by autonomous judgments and improvising actions [43].

In this paper we extend the SAFER-HRC methodology introduced in [3,4] for the assessment of the safety of Human-Robot Collaborative (HRC) applications, by improving the formal model of operators on which the methodology relies.

S. Tonetta et al. (Eds.): SAFECOMP 2017, LNCS 10488, pp. 89–104, 2017.
DOI: 10.1007/978-3-319-66266-4_6

Unlike classic hazard identification approaches such as FTA [27] and FMECA [26] which cannot deal with unpredictable human behavior, the proposed model takes into account both normative human behavior and a number of possible deviations. Thus, previously unrecognized hazardous situations are detected.

Two common approaches to model human operators in system models are: (i) task-analytic models, which represent the observable manifestation of operators' behavior; and (ii) cognitive models, which instead describe the cognitive process behind the operator's observable behavior [11]. In the first approach tasks are represented as hierarchies of actions whose execution sequence is modeled by if-then logic rules, and the operator behavior is part of the overall model of the system. The latter techniques, instead, capture the knowledge used by the operator to execute the task. The human cognitive state is usually described by a set of variables that change with respect to the other elements in the system, following a set of logical production rules. Cognitive models, unlike task-analytic approach, highlight the erroneous behaviors of the operator—i.e., human activities that do not achieve their goals [39]—and provide clues about why such behavior arose, and flaws in the design that allow their occurrence [16]. These clues can be used to refine the system design to reduce the likelihood of operator errors. Pairing a hierarchical task model with a human cognitive architecture has been used to determine the role of operators in the system performance or failure [40]. Cognitive models can be integrated in a larger formal model and evaluated as a part of the system [8].

In [3,4] we formalized collaborative systems to verify the physical safety of human operators. We used a task-analytic approach that models collaborative systems in terms of three main modules—Operator, Robot and Layout—and breaks down tasks into atomic actions. In this work, we combine the previous model with a cognitive model capturing erroneous human behavior driven from the operator's perception of the environment and mental decisions. Thus, we can describe likely errors due to certain characteristics that are frequently found in human operators. To this end, the cognitive-driven reasons and phenotypes of errors are treated as black boxes, and their consequences leading to hazardous situations are analyzed in terms of human safety. The proposed methodology captures human errors and inspects harmful situations caused by those errors.

This paper is structured as follows: Sect. 2 discusses related works on the formalization of human behaviors, especially errors. Section 3 introduces SAFER-HRC and its extension with a model of operators' erroneous behaviors. Section 4 shows how the improved model helps detect new, previously unrecognized hazardous situations. Section 5 discusses some open issues and concludes.

2 Related Work on Human Behavior and Errors

Cognitive architectures such as SOAR [34], ACT-R [2,42], OCM [14,35] and PUM [45] have been widely-used to model the normative and erroneous human behavior. SOAR and ACT-R cannot be used in formal verification approaches since they lack a formal semantics, whereas OCM is known to be suitable only for

designing air traffic control applications. PUM, instead, has been used in a wider range of applications [12,13,18,44], where the model of the operator includes a set of goals and a set of actions to achieve them. The operator's knowledge of how to execute actions is modeled as a set of rules. PUM models differentiate between mental and physical actions: first the operator decides to execute an action; then, the action eventually starts. PUM models can be used in different domains, such as Collaborative Robotics. Nevertheless they should be able to replicate human erroneous behavior, so that a verification process can detect the hazardous situations raised by them. Currently, model-based techniques that include human errors focus on interface devices and not physical collaboration and contacts [10]. For example, Physiograms [19] model interfaces of physical devices and [9,36] study the impacts of miscommunication in human-human collaboration while interacting with critical systems. [7,38] explore human deviation from correct instructions using ConcurTaskTrees [37]. [41] upgrades the SAL cognitive model with systematic errors taken from empirical data. Nevertheless, using data related to specific case studies can lead to the loss of generality. [17,19,33] are other examples that combine cognitive and formal models to capture erroneous human behaviors in interactions with devices.

No generally accepted classification of human errors is available. [43] divides errors into two main groups: location- and orientation-related. The former happen when an action is related to multiple locations, or if there are two actions which take place in different locations within the task. The latter happen, for example, when the operator needs to hold a workpiece while it is being screwdriven on a pallet: even if the location is correct, the action might not conclude due to the wrong orientation of the workpiece. [25], instead, classifies the simple phenotypes of human errors into: repetition of an action, reversing the order of actions, omission of actions, late actions, early actions, replacement of an action by another, insertion of an additional action from elsewhere in the task, and intrusion of an additional unrelated action. [22] manually introduces these phenotypes into task specifications, which suffers from many false negatives. The author identifies complex phenotypes created by the combination of simple ones: (i) undershooting, which occurs when an action stops too early; (ii) side tracking, where a segment of unrelated action is carried out, then the correct sequence is resumed; (iii) capturing, where an unrelated action sequence is carried out instead of the expected one; (iv) branching, where the wrong sequence of actions is chosen; (v) overshooting, where the action carries on past its correct end point by not recognizing its post-conditions. Complex phenotypes are combinations of simple ones, as shown in Fig. 1 (where "wrong place" refers to the action's temporal position in the execution sequence, not to a location in the layout). [16] proposes to include in the model strong enough design rules so that cognitively plausible erroneous behavior is not allowed, or is at least is unlikely. Further, authors in [15] propose a framework for reasoning about human-in-the-loop, to identify potential causes for human failure in human-automation interface applications. They have also explored the role of personal variables such as knowledge, experience, the ability to complete recommended actions (self-efficiency), effectiveness of those actions (response-efficiency) and operator trust in them.

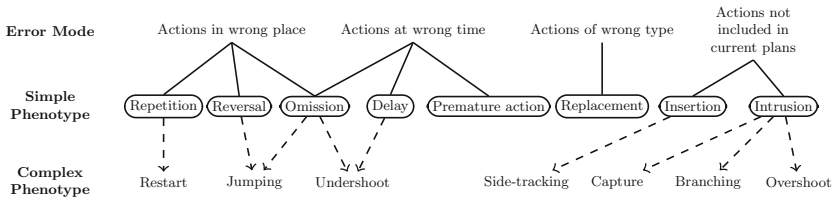

Fig. 1. A taxonomy of erroneous actions phenotypes, taken from [22]

To conclude this section, we remark that some works use probabilistic models to model human operators [20,23,32]. In this vein, [21] uses probability distributions to express human behavior, where distribution functions capture factors such as fatigue and proficiency. However, no validated data from live experiments with real users have been made available, and the article mentions that fictional data were used in order to simulate a distribution over erroneous human behavior. In this work, we do not use probabilistic models to capture human behaviors mainly for two reasons. The first one is a lack of data concerning realistic values for such probabilities. The second is that in our approach the operator model is part of a bigger one that is used to perform the safety analysis of systems. The main aim of the overall model is to identify hazardous situations (including those caused by human errors) and to define suitable Risk Reduction Measures (RRMs, see Sect. 3) to reduce their resulting risk. The goal is to eventually introduce relevant RRMs for every erroneous situation that the safety engineers deem relevant, regardless of their probability value. Nevertheless, once the causes of human errors are identified and categorized (as in Sect. 3), one could evaluate the corresponding probabilities of the different error causes. This can help refine the design by defining more efficient and relevant RRMs.

3 Operator Model

The SAFER-HRC methodology [3,4] hinges on a formal model of HRC applications that relies on a discrete notion of time, and on finite discretizations of the notion of space and of the scalar properties of systems—e.g., velocity. Let us first provide some background on the techniques on which the model is based.

Preliminaries. The SAFER-HRC formal model is expressed through the TRIO metric temporal logic, which features an underlying linear temporal structure and a quantitative notion of time [24]. TRIO formulae are built out of the usual first-order connectives, operators, and quantifiers, as well as a single basic modal operator, called Dist, that implicitly relates the *current time*, to another time instant: given a time-dependent formula ϕ and a (arithmetic) term t indicating a time distance (either positive or negative), formula $\text{Dist}(\phi, t)$ specifies that ϕ holds at exactly t time units from the current one. While TRIO can exploit both discrete and dense time domains, in this work we assume the standard

Table 1. List of derived TRIO operators

Operator	Definition	Meaning
Futr(ϕ, d)	$d > 0 \wedge \text{Dist}(\phi, d)$	ϕ occurs exactly at d time units in the future
Past(ϕ, d)	$d > 0 \wedge \text{Dist}(\phi, -d)$	ϕ occurred exactly at d time units in the past
AlwF(ϕ)	$\forall t(t > 0 \Rightarrow \text{Dist}(\phi, t))$	ϕ holds always in the future
Until(ϕ, ψ)	$\exists t(\text{Futr}(\psi, t) \wedge \forall t'(0 < t' < t \Rightarrow \text{Dist}(\phi, t')))$	ϕ will hold until ψ occurs
SomF(ϕ)	$\exists t(t > 0 \wedge \text{Dist}(\phi, t))$	ϕ occurs sometimes in the future
SomP(ϕ)	$\exists t(t > 0 \wedge \text{Dist}(\phi, -t))$	ϕ occurred sometimes in the past

model of the nonnegative integers $\mathbb{N}$ as discrete time domain. TRIO defines a number of *derived* temporal operators from the basic Dist, through propositional composition and first-order logic quantification. Table 1 defines some of the most significant ones, including those used in this work. The satisfiability of TRIO formulae is in general undecidable. SAFER-HRC uses a decidable subset of the language, which can be handled by automated verification tools, such as the Zot bounded satisfiability checker [1]. Zot is used to check the model of the system against desired safety properties. If the property is not satisfied, Zot provides a counterexample witnessing a system execution that violates the property.

Basic Model. The proposed model for HRC applications includes three main modules: operator (O), robot (R), and layout (L). ISO15066 [31] contains biomechanical studies that divide the human body into 26 sections according to their pain tolerance and being injury prone. In consistency to this standard, O describes all of the identified body sections, the relative constraints concerning their movements, and their position and velocity at each time instant. R divides robots into their components (arms and end-effectors), and captures the velocity, position and force of each part at each time instant. The values for velocity and force range over the quantized set $\{\text{none}, \text{low}, \text{mid}, \text{high}\}$. These values will later be used to calculate the risk value of the system. The layout L of the workspace is partitioned into a finite number of regions, each with a defined shape, material and an attribute stating the presence of obstacles ($\{\text{occluded}, \text{clear}, \text{free}, \text{warning}\}$). At each time instant, the position of each element of R and O corresponds to the region in which it is located.

SAFER-HRC aims to identify hazardous situations—as described in [29]—by providing their formalization, compute a quantized risk ($\in \{0, 1, 2\}$), and define corresponding Risk Reduction Measures (RRMs) in conformance with [30]. Each HRC task is broken down into a set of elementary actions, which are the smallest possible functional units and are executed either by operators (the performer of the action is the operator: $a_{i,actor} = \text{op}$) or by robots (the performer is the robot: $a_{i,actor} = \text{ro}$). Each action is associated with a pair $\langle \text{preC}, \text{postC} \rangle$ of pre- and post-conditions, which are formalized through logical constraints capturing the

action's temporal relations with other actions (e.g., precedence). In each instant, the state of an action is described by one of the following atomic formulae:

- $a_{i,sts}$ = ns (not started): state in which not all pre-conditions are true, yet.
- $a_{i,sts}$ = wt (waiting): a state of human actions only, in which all pre-conditions are true, but the action has not yet started. It allows for the introduction of delays or hesitations in human actions.
- $a_{i,sts}$ = exe (executing): running state, triggered by the validity of all pre-conditions.
- $a_{i,sts}$ = sfex (safe executing): special running state, triggered by the activation of at least one RRM. This means that there are currently detected hazards in the system but their consequences are being mitigated by RRMs without interrupting the execution of the action.
- $a_{i,sts}$ = hd (hold): exception state, entered upon an explicit suspension of execution due to a request from the operator. When the state is active, the execution is momentarily paused, although some RRMs may be enabled.
- $a_{i,sts}$ = dn (done): regular termination state, triggered by the validity of all post-conditions.
- $a_{i,sts}$ = exit: whenever the desired safety properties are violated in any state value, the action quits (together with all other actions). The transition may be triggered by situations (e.g., hazards and risks) detected in other actions.

The proposed model also includes the formalization of two recognized types of physical hazards, according to [31]: (i) Transient (Tr) ones, which are fast, impact-like contacts, where body parts are hit and then recoil because of the kinetic energy transferred to the body; and (ii) Quasi-static (Qs) ones, which are sustained contacts of body parts against a constraining object with continuous energy flow from the robot. SAFER-HRC introduces two types of RRMs to treat the detected hazards: Speed and Separation Monitoring (SSM) and Power and Force Limitation (PFL) to avoid physical contacts or limit their consequences, respectively. The former type maintains the robot speed constantly low when the robot works at a distance less than a predefined threshold from the operator; the latter type, instead, limits the value of the robot force.

The original version of the O module focused more on modeling the normative and expected behavior of the operator. In this paper, our aim is to model the operator behavior more realistically, and to capture also situations such as unintended behavior—errors or misuses. The operator model has been extended by including reasonably predictable human errors, to detect hazards that they cause which have been overlooked in the basic model.

Extended Operator Model: Formal Cognitive Model. We now present an extension of module O of the model proposed in [3]. The new model allows for the generation of traces that include human errors and encompasses functional and behavioral human modeling. Since the model is used to generate errors and to analyse the effects of errors on human safety, it focuses on the consequences of human errors rather than their causes. We explain below how phenotypes of

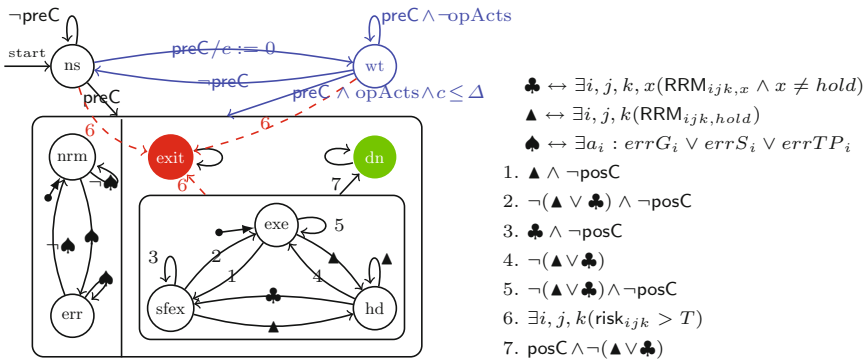

Fig. 2. Parallel composition of execution and erroneous state of each action

erroneous human behavior are taken into account, but we leave the study of genotypes for future work. The model relies on the following assumptions.

(i) Operator actions are fully non-deterministic and are rationally possible to be taken by the operator as their pre-conditions turn true. Every action has a timeout Δ within which the operator needs to decide whether to start the execution of the action after its pre-conditions hold.

(ii) Each action has a corresponding preceding mental decision $opActs_i$, which works as its trigger. When $opActs_i$ becomes true, it means that in the next instant the operator starts executing action a_i. An action which is waiting starts executing right after when the operator makes her mental decision to start acting. Each action has also a corresponding mental decision about stopping (ending, pausing) its execution, captured by predicate $opStops_i$. These two predicates capture the perception of the operator from processing the state of her environment, and the decisions (starting/stopping) she will make according to those perceptions. The decision of the operator to start and end an action is due to what she may see, touch or feel. Instead of modeling each of these causes separately, we model the decision itself directly.

(iii) Additional states for operator actions are introduced describing if the execution state is normative (nrm) or erroneous (err). Figure 2 shows the statechart capturing the evolution of a single action, which can be in one of the possible execution states $\{\text{exe}, \text{sfex}, \text{hd}, \text{exit}\}$ **and** at the same time in one of $\{\text{err}, \text{nrm}\}$.

(iv) The number of possible human errors is bounded, to avoid making the model too complex and also to avoid generating a lot of false positive situations.

Formalizing Human Errors. Given the classification of [43] and the phenotypes introduced in [25], we categorize human errors in three main types:

1. Time-related errors ($errTP$), which occur when the operator does not follow the correct temporal ordering of actions. Eventually these errors lead to an instance of one of the phenotypes introduced in Fig. 1.

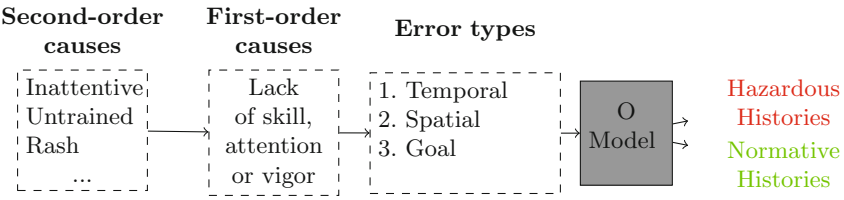

Fig. 3. Types of error that occur due to first-order causes related to the operator's state during execution

2. Space-related errors ($errS$), that happen when the operator is in the wrong L region or places the instruments in the wrong regions.
3. Goal-related errors ($errG$), that arise when the operator is in the right spot to execute an action and starts the execution with no time-related error, but she does not follow instructions correctly and the action is performed poorly. This happens more frequently when the operator is untrained and unskillful.

An error can appear due to two different layers of motives: first-order and second-order causes. Errors originate directly from first-order causes, which are the lack of at least one of the following factors: skill (knowledge + experience + trust in design), attention or vigor. First-order causes themselves originate from the state of the operator, who can be fatigued, inattentive, unaware of the instructions, rash, etc. The goal of this work is to enrich the list of detected hazards, and to this end considering only first-order causes is enough. Second-order causes, on the other hand, are less relevant during the process of identifying new instances of hazardous situations and assessing their corresponding risks. The relations between errors and their first- and second-order causes are shown in Fig. 3. In the rest of this section we illustrate how the formal model takes human errors into account and represents the situations that can arise because of them. In fact, human errors must be formalized and included in the model to systematically generate and verify them during application of the SAFER-HRC methodology.

1. Time-related Errors happen when the operator manipulates the expected temporal order of actions and creates an unwanted situation.

$$errTP_i \Leftrightarrow Repetition_i \vee Omission_i \vee Late_i \vee Early_i \vee Intrusion_i \tag{1}$$

Our proposed model formalizes each phenotype introduced in [25] and listed in Fig. 1 through the formulae below. Implicitly, since we are dealing with operator errors, constraint $a_{i,actor} = \text{op}$ is assumed in all the following formulae.

The model captured in module O does not force the operator to start another action or remain idle after executing an action. Consider for example the case (see Sect. 4) where the operator should prepare the jigs and fixtures before the robot starts moving towards the pallet. If the operator repeats the preparation for other jigs or continues to play with jigs after they are already in place, there could be a collision between the operator's hand and the robot end-effector. The formula below formalizes an erroneous **repetition** of an action i by stating that

this occurs when the action is currently done and either it was just completed without the operator acknowledging this (i.e., $\text{Past}(\neg a_{i,sts} = \text{dn} \wedge \neg opStops_i, 1)$ holds), or the operator decides to perform it again (i.e., $opActs_i$ holds).

$$Repetition_i \Leftrightarrow a_{i,sts} = \text{dn} \wedge (\text{Past}(a_{i,sts} \neq \text{dn} \wedge \neg opStops_i, 1) \vee opActs_i) \tag{2}$$

The cases of **reversed** and **replaced** actions in the temporal model are actually covered by the formalization of early/late actions presented below. To allow for the wrong sequencing of operator actions, the corresponding pre-conditions are looser than those for actions executed by robots. In fact, whereas the pre-condition for a robot action typically includes a list of other actions that must have been completed, this does not occur for operator actions, which can be executed as soon as the operator is in the right area and has the required tools.

An action a_i is **omitted** if the operator never executes it, thus predicate $opActs_i$ is never true ($\text{AlwF}(\neg opActs_i)$ holds). Consequently, the status of a_i will always remain ns or wt in the future (i.e., $\text{AlwF}(a_{i,sts} = \text{wt} \vee a_{i,sts} = \text{ns})$ holds), which prevents the execution of robot actions whose pre-conditions require the termination of a_i. Subformula $\text{SomP}(\text{Lasted}(a_{i,sts} = \text{wt}, \Delta))$ states that for action i to be considered omitted it must have previously been in the waiting state for at least Δ time units—otherwise it might simply be the case that not enough time has been given the operator to start it.

$$\begin{aligned} Omission_i \Leftrightarrow & \text{AlwF}(\neg opActs_i \wedge (a_{i,sts} = \text{wt} \vee a_{i,sts} = \text{ns})) \wedge \\ & \text{SomP}(\text{Lasted}(a_{i,sts} = \text{wt}, \Delta)) \end{aligned} \tag{3}$$

An action i is **delayed** (i.e., predicate $Late_i$ holds) if the operator starts executing it (i.e., $opActs_i$ holds), and the timeout Δ to start the action after it was enabled has already expired sometimes in the past.

$$Late_i \Leftrightarrow opActs_i \wedge \text{SomP}(\text{Lasted}(a_{i,sts} = \text{wt}, \Delta)) \tag{4}$$

Action i is **prematurely executed** (i.e., $Early_i$ holds) when its pre-conditions are not yet satisfied (it is still "not started"), but the operator has already decided to execute it ($opActs_i$ holds). In fact, the operator's act does not change the status of a_i, which remains "not started".

$$Early_i \Leftrightarrow a_{i,sts} = \text{ns} \wedge \text{Past}(opActs_i, 1) \tag{5}$$

Intrusion and **insertion** errors occur when the operator confuses the task to be executed with another one. Both these situations are captured by predicate $Intrusion_i$, which is formalized by the formula below. More precisely, if T is the task being executed, $Intrusion_i$ holds when there is an action of T that is in the waiting state, but the operator starts executing an action j that is not in T:

$$Intrusion_i \Leftrightarrow a_{i,sts} = \text{wt} \wedge \exists j \notin T : \text{Past}\big(opActs_j, 1\big) \tag{6}$$

2. Space-related errors are raised due to movements of the operator in the layout which are not over-constrained in the model. An action is prone to space-related errors when the operator violates its location-base requirements during

its execution (i.e., $safeL_i$ is false while the action is executing), or if the action goes into a "hold" state without the operator having asked to stop the action (i.e., $opStops_i$ is false, which means that the holding state has been entered for reasons related to the behavior of the operator). Examples of operator behaviors that lead to such situations are: leaving her required position or safe spot, getting closer to robot than the distance indicated in the instructions, approaching the robot when an alarm signals to stay away.

$$\begin{aligned} errS_i \Rightarrow & \big(\neg safeL_i \wedge (a_{i,sts} = \text{exe} \vee a_{i,sts} = \text{sfex})\big) \vee \\ & \big(a_{i,sts} = \text{hd} \wedge (\text{Past}(a_{i,sts} \neq \text{hd} \wedge \neg opStops_i), 1)\big) \end{aligned} \tag{7}$$

3. Goal-related Errors deal with actions which are not executed consistently with the instructions of the task. For example if the operator does not place the fixtures properly or tightens the part on the pallet while screw-driving. The presence of goal-related errors is represented by predicate $errG_i$, which is non-deterministically assigned values during the exploration of the system traces. Notice that, in practice, goal-related errors can be detected, for example, through the use of cameras installed in the work-cell; hence, predicate $errG_i$ can be seen as capturing the information provided by such cameras.

The addition of the formalization of these phenotypes to the model allows us to check whether there are hazardous situations that cannot be mitigated by currently introduced RRMs, thus if the base model failed to capture hazards that arise due to human errors. As mentioned in Sect. 3, the model introduces constraints on the number of human errors during execution of a task. In fact, it is reasonable that an operator does not make too many errors during a single execution of an application; on the other hand, this number can be configured in the model, although the higher the number, the greater the complexity of the model and the required analysis time. The following formula—which has been simplified for the sake of brevity, and where $\text{past}(count_i, 1)$ is the function returning the value of $count_i$ at the previous instant—describes the increment of the counter of errors made during action i:

$$\begin{aligned} count_i = \text{past}(count_i, 1) + 1 \Leftrightarrow & (errG_i \wedge \text{Past}(\neg errG_i, 1)) \vee (errS_i \wedge \text{Past}(\neg errS_i, 1)) \\ & \vee (errTP_i \wedge \text{Past}(\neg errTP_i, 1)) \end{aligned} \tag{8}$$

The total counter *count* is simply the sum of all $count_i$, and we impose that it never exceeds a (configurable) threshold N: $\text{AlwF}(count \leq N)$. Notice that, in a similar vein, in the formal model we impose a constraint that $errG_i$ cannot occur more often than every 5 time units.

The next section shows how experiments carried out with the enhanced model can highlight hazardous cases that originate from human errors.

4 Case Study

The case study on which we applied the enhanced SAFER-HRC methodology features a hybrid human-robot assembly task in the preparation of a machine

tool pallet—i.e., setting jigs and mounting/dismounting workpieces into fixtures before/after machining. In Flexible Manufacturing Systems, load/unload stations are the only parts handled manually in a largely automatic procedure. A collaborative robot, capable of relocating inside the production plant, can be used for a number of tasks—e.g., carrying containers with workpieces or finished items, supporting workpieces during assembly. The overall goal of the HRC application is to provide all services that improve the ergonomics of manual operations, release the operators from repetitive/heavy/dull tasks, provide quantitative logs of operations, reduce errors. The operator can choose to achieve the task in different ways: either she performs the actions related to the pick-and-place subtask while the robot screw-drives (alternative V1), or vice-versa (alternative V2). The work-cell layout $\mathcal{L}$ is divided according to a polar grid, partitioning the angular range of the robot shoulder joint and the outreach of the manipulator from its base (see Fig. 4).

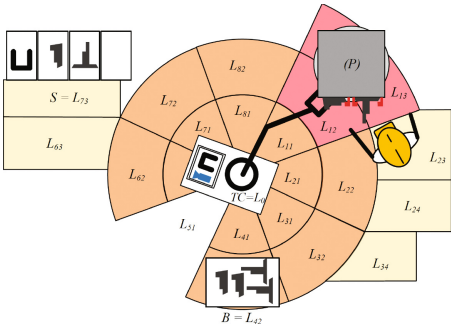

Fig. 4. Top-view representation of the discretized layout

Formal verification was carried out through the Zot [1] tool, which exhaustively explores the state-space of traces of the model up to a bounded length [24]. All verification experiments were performed using the `bvzot` plugin [6] on a 2.6 GHz Intel® core™ i5 machine. The bounded search depth for Zot was set to 30, and the verification execution was a matter of few seconds. We verified that the risk does not exceed a desired threshold (2, in this case), unless there is a RRM which mitigates it right at the next time instant. The property is captured by the following formula:

$$\forall i,j,k\Big(\mathrm{Alw}\big((\mathsf{risk_{ijk}} < 2) \ \vee \ (\mathsf{risk}_{ijk} = 2 \wedge \exists y(\mathsf{RRM}_{ijk,y}) \wedge \mathrm{Futr}(\mathsf{risk}_{ijk} < 2, 1))\big)\Big) \tag{9}$$

Given the extensions introduced in the operator model, we aim to guarantee that the property is verified also when human errors are systematically considered at design time. In particular, we observed new hazardous situations that the old model was not able to capture or mitigate. Here we report on a pair of them: the first one describes a new quasi-static hazard instance that persists even though

Table 2. Output of the verification tool: state of the model at each time instant

(a) Risk Analysis Table: hazards present (Hzds), location of the occurrence (L_k), moving and direction of operator and robot parts (end-effector, first and second links), relative speed, force and separations and current risk value

t	Executing	Hzds	L_k	Se	Still/Move	Direction (Reach/Leave)	v	f	Sep (ee,op)	Sep (R1,op)	Sep (R2,op)	Risk
15	actn−V ... 8 − 1 ...	1.Qs of Waist area by R1	2	4	Op−EE−R2−R1	EE−R2−R1						
		2.Qs of Arm area by R1	2	4	m −m −s −m	l −l −l	high	high	close	close	close	2

t	Executing	Hzds	L_k	Se	Still/Move	Direction (Reach/Leave)	v	f	Sep (ee,op)	Sep (R1,op)	Sep (R2,op)	Risk
16	actn−V ... 8 − 1 ...	1.Qs of Arm area by R1	1	3	Op−EE−R2−R1	EE−R2−R1						
		2.Qs of Waist area by R2	1	3	s −m −m −m	r −r −r	high	low	close	close	close	2

(b) Error Detection Table

t	Actn−V	status	Erroneous State	error type	Hzds	Risk	RRMs
15	7 − 1 ... 8 − 1 ...	... done exrm	... err norm	S − G − TP ... 0 , 0 , (re) 0 , 0 , (−) ...	Qs of Arm area by R1 Qs of Waist area by R1	2	3 6

t	Actn−V	status	Erroneous State	error type	Hzds	Risk	RRMs
16	7 − 1 ... 8 − 1 ...	... done exrm	... err norm	S − G − TP ... 0 , 0 , (re) 0 , 0 , (−) ...	Qs of Arm area by R1 Qs of Waist area by R2	2	3 6

RRMs are active in the system. The second one demonstrates a situation for which there is not actually a reasonable RRM that can fully mitigate it.

Persistent Hazard. Table 2 shows an example of the output of the Zot formal verification. Table 2(a) is used by safety analyzers to examine the risk level at each time instant. In this example, the chosen alternative is V1 and actions from V2 are considered as Intrusion errors. The table shoes that there are two time instants in a row with risk value equal to two, which violates the desired safety property. Table 2(b) shows where human errors happen; in particular, it shows that, although there are active RRMs at times 14 and 15, the risk level is still 2 and the error present is $Repetition_7$. Thus, we associate a stronger RRM to the hazard "Qs on Arm area by R1", which not only limits the relative force—which was enough in absence of human errors—but also the relative velocity value.

Unreasonably Uncommon Behavior. The formal model presented in this paper has not been designed to directly address irrational human behaviors or intentional misuses. Nevertheless, it is able to detect some situations that can be classified as "unreasonably uncommon", as they should be very unlikely to occur. For example, consider the case of an operator that purposely throws

Table 3. New hazardous situations detected by the tool (a) Situation where the operator is rapidly closing on the robot, which is still (b) Error cause: the operator is mistakenly executing an action belonging to another task, which requires her to move where the robot is (`in` refers to intrusion error)

(a) Situation where the operator is rapidly closing on the robot, which is still

t	Executing	Hzds	L_k	Se	Still/Move	Direction (Reach/Leave)	v	f	Sep (ee,op)	Sep (R1,op)	Sep (R2,op)	Risk
5	actn-V	–	–	–	Op-EE-R2-R1	EE-R2-R1	high	mid	close	close	close	0
					m −s −s −s	r −r −r						

(b) Error cause: the operator is mistakenly executing an action belonging to another task, which requires her to move where the robot is (`in` refers to intrusion error)

t	Actn-V	status	Erroneous State	error type	Hzds	Risk	RRMs
5	1 − 1	done	norm	S − G − TP			
	2 − 1	done	err	0 , 0 , (−)			
	3 − 1	wt	err	0 , 0 , (re)	0		
	...	...	...	0 , 0 , (in)			

herself under the robot sharp end-effector. Defining a RRM for this case can be considered useless, because a determined operator would still not follow it. The experiment of Table 3 shows a case where the operator is unexpectedly moving towards the robot. The reason could be intentional harm—caused by an instance of $errG$— or that the operator is doing something wrong that leads to an unwanted situation—presence of instances of $errTP$, $errS$ or $errG$. However, no hazard has been identified by the tool because we kept very unlikely scenarios out of the model when formalizing hazards and risks (essentially, the risk for these kinds of situations is considered low), and in any other case a still robot in the homing position (L_0) is not considered a source of harm and danger. In fact, the situation in this case has been detected by perusing a trace produced by the Zot tool in "simulation mode", that did not highlight a high risk.

5 Discussion and Conclusion

In this work, the SAFER-HRC methodology is extended to capture and consider also the erroneous behavior of human operators. Using the improved model, we re-checked desired safety properties of previously analyzed HRC applications; the new checks highlighted some instances of hazards that had been overlooked in previous runs of the methodology due to a lack of precise human modeling.

The improved accuracy of the model opens the possibility to refine previously-introduced RRMs in order to provide a trade-off between safety and efficiency. Previously, very general RRMs had to be introduced, such as "reduce speed down to a certain value", or "reduce applied force down to zero". However, the newly introduced details concerning the reasons behind and the exact configuration of hazards allow us to define more specific and hazard-dependent RRMs and avoid the use of over-conservative RRMs when a less strict RRM can provide safety.

As a future step, we plan to associate probability distributions with relations between hazards and human errors to see how errors can increase the occurrence of hazards. In this way we might be able to provide more efficient treatments for hazards without compromising the functionality of the system. We are also concluding a prototype tool—a plug-in for Papyrus Eclipse Environment—which resolves the difficulty of dealing with logic formulae to model the applications for safety experts. The tool automatically transforms UML diagrams to Logical formulae and invokes Zot to verify the safety property so that the design and verification of HRC application is made easier, faster and more automated.

Acknowledgment. We thank the anonymous reviewers for their comments and suggestions, which helped us improve the paper.

References

1. The Zot bounded satisfiability checker. http://github.com/fm-polimi/zot
2. Anderson, J.R.: ACT: a simple theory of complex cognition. Am. Psychol. **51**, 355–365 (1996)
3. Askarpour, M.: Risk assessment in collaborative robotics. In: Proceedings of FM-DS (2016)
4. Askarpour, M., Mandrioli, D., Rossi, M., Vicentini, F.: SAFER-HRC: safety analysis through formal verification in human-robot collaboration. In: Skavhaug, A., Guiochet, J., Bitsch, F. (eds.) SAFECOMP 2016. LNCS, vol. 9922, pp. 283–295. Springer, Cham (2016). doi:10.1007/978-3-319-45477-1_22
5. Baier, C., Katoen, J.P.: Principles of Model Checking (2008)
6. Baresi, L., Pourhashem Kallehbasti, M.M., Rossi, M.: Efficient scalable verification of LTL specifications. In: Proceedings of ICSE (2015)
7. Basnyat, S., Palanque, P.: A task pattern approach to incorporate user deviation in task models. In: Proceedings of ADVISES (2005)
8. Bolton, M.L.: Automatic validation and failure diagnosis of human-device interfaces using task analytic models and model checking. Comput. Math. Organ. Theory **19**, 288–312 (2013)
9. Bolton, M.L.: Model checking human-human communication protocols using task models and miscommunication generation. J. Aerospace Inf. Syst. **12**, 476–489 (2015)
10. Bolton, M.L., Bass, E.J., Siminiceanu, R.I.: Generating phenotypical erroneous human behavior to evaluate human-automation interaction using model checking. Int. J. Hum.-Comput. Stud. **70**(11), 888–906 (2012)
11. Bolton, M.L., Bass, E.J., Siminiceanu, R.I.: Using formal verification to evaluate human-automation interaction: a review. IEEE Trans. SMC Syst. **43**(3), 488–503 (2013)
12. Butterworth, R., Blandford, A., Duke, D.: The role of formal proof in modelling interactive behaviour. In: Markopoulos, P., Johnson, P. (eds.) Proceedings of DSV-IS, pp. 87–101. Springer, Vienna (1998). doi:10.1007/978-3-7091-3693-5_7
13. Butterworth, R., Blandford, A., Duke, D.: Demonstrating the cognitive plausibility of interactive system specifications. Form. Asp. Comp. **12**, 237–259 (2000)
14. Cerone, A., Lindsay, P.A., Connelly, S.: Formal analysis of human-computer interaction using model-checking. In: Proceedings of SEFM (2005)

15. Cranor, L.F.: A framework for reasoning about the human in the loop. In: Proceedings of UPSEC (2008)
16. Curzon, P., Blandford, A.: From a formal user model to design rules. In: Forbrig, P., Limbourg, Q., Vanderdonckt, J., Urban, B. (eds.) DSV-IS 2002. LNCS, vol. 2545, pp. 1–15. Springer, Heidelberg (2002). doi:10.1007/3-540-36235-5_1
17. Curzon, P., Blandford, A.: Formally justifying user-centred design rules: a case study on post-completion errors. In: Boiten, E.A., Derrick, J., Smith, G. (eds.) IFM 2004. LNCS, vol. 2999, pp. 461–480. Springer, Heidelberg (2004). doi:10.1007/978-3-540-24756-2_25
18. Curzon, P., Rukšėnas, R., Blandford, A.: An approach to formal verification of human-computer interaction. Form. Asp. Comput. **19**(4), 513–550 (2007)
19. Dix, A.J., Ghazali, M., Gill, S., Hare, J., Ramduny-Ellis, D.: Physigrams: modelling devices for natural interaction. Form. Asp. Comput. **21**, 613 (2009)
20. Feng, L., Humphrey, L., Lee, I., Topcu, U.: Human-interpretable diagnostic information for robotic planning systems. In: Proceedings of IROS (2016)
21. Feng, L., Wiltsche, C., Humphrey, L., Topcu, U.: Synthesis of human-in-the-loop control protocols for autonomous systems. IEEE T-ASE **13**(2), 450–462 (2016)
22. Fields, R.E.: Analysis of erroneous actions in the design of critical systems. Ph.D. thesis, University of York (2001)
23. Fu, J., Topcu, U.: Synthesis of joint control and active sensing strategies under temporal logic constraints. IEEE Trans. Automat. Contr. (2016)
24. Furia, C.A., Mandrioli, D., Morzenti, A., Rossi, M.: Modeling Time in Computing (2012)
25. Hollnagel, E.: Cognitive reliability and error analysis method (CREAM) (1998)
26. International Electrotechnical Commission: IEC 60812: 2006: Analysis techniques for system reliability - Procedure for failure mode and effects analysis
27. International Electrotechnical Commission: IEC 60812: 2006: Fault tree analysis
28. International Electrotechnical Commission: IEC 61882: Hazard and operability studies (HAZOP studies) - Application guides
29. International Standard Organisation: ISO12100: 2010, Safety of machinery - General principles for design - Risk assessment and risk reduction
30. International Standard Organisation: ISO14121-2: 2007, Safety of machinery - Risk assessment - Part 2
31. International Standard Organisation: ISO/TS15066: 2015, Robots and robotic devices - Collaborative robots
32. Junges, S., Jansen, N., Katoen, J., Topcu, U.: Probabilistic model checking for complex cognitive tasks - A case study in human-robot interaction. CoRR (2016)
33. Kim, N., Rothrock, L., Joo, J., Wysk, R.A.: An affordance-based formalism for modeling human-involvement in complex systems for prospective control. In: Proceedings of WSC (2010)
34. Laird, J.E.: The Soar Cognitive Architecture. MIT Press, Cambridge (2012)
35. Lindsay, P.A., Connelly, S.: Modelling erroneous operator behaviours for an air-traffic control task. In: Proceedings of AUIC (2002)
36. Pan, D., Bolton, M.L.: Properties for formally assessing the performance level of human-human collaborative procedures with miscommunications and erroneous human behavior. Int. J. Ind. Ergonom. (2016)
37. Paterno, F., Mancini, C., Meniconi, S.: ConcurTaskTrees: a diagrammatic notation for specifying task models. In: Howard, S., Hammond, J., Lindgaard, G. (eds.) INTERACT 1997. IFIP AICT, pp. 362–369. Springer, Boston, MA (1997). doi:10.1007/978-0-387-35175-9_58

38. Paternò, F., Santoro, C.: Preventing user errors by systematic analysis of deviations from the system task model. Int. J. Hum.-Comput. Stud. **56**, 225–245 (2002)
39. Reason, J.: Human Error. Cambridge University Press, Cambridge (1990)
40. Ritter, F.E., Rooy, D.V., Amant, R.S., Simpson, K.: Providing user models direct access to interfaces: an exploratory study of a simple interface with implications for HRI and HCI. IEEE Trans. SMC Syst. (2006)
41. Ruksenas, R., Back, J., Curzon, P., Blandford, A.: Verification-guided modelling of salience and cognitive load. Form. Asp. Comput. **21**, 541 (2009)
42. Salvucci, D.D., Lee, F.J.: Simple cognitive modeling in a complex cognitive architecture. In: Proceedings of CHI (2003)
43. Shin, D., Wysk, R.A., Rothrock, L.: Formal model of human material-handling tasks for control of manufacturing systems. IEEE Trans. SMC Syst. **36**(4), 685–696 (2006)
44. Werther, B., Schnieder, E.: Formal cognitive resource model: modeling of human behavior in complex work environments. In: Proceedings of CIMCA-IAWTIC (2005)
45. Young, R.M., Green, T.R.G., Simon, T.J.: Programmable user models for predictive evaluation of interface designs. In: Proceedings of CHI (1989)