SaRA: A Tool for Safe Human-Robot Coexistence and Collaboration through Reachability Analysis

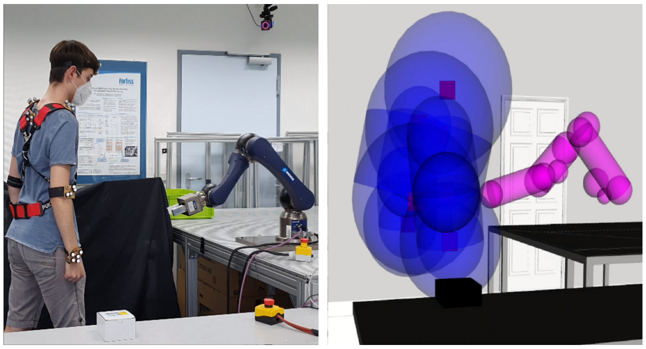
Sven R. Schepp, Jakob Thumm, Stefan B. Liu, and Matthias Althoff

***Abstract*— Current safety mechanisms implementing industry standards for human-robot coexistence separate humans and robots through caging. Other approaches allowing humans to enter the workspace of manipulators do not provide formal safety guarantees. Thus, this study aims to facilitate the widespread adoption of collaborative robots by presenting SaRA, an extensible tool that performs set-based reachability analysis and formally guarantees safety. Our experimental results show that the set-based prediction of a human can be computed in a few microseconds, using SaRA, allowing for real-time consideration of many surrounding humans in an environment.**

I. Introduction

The coexistence and collaboration of humans and robots has become increasingly important with the adoption of robots for surgical procedures [1], [2], manufacturing [3], [4], agriculture [5], or construction work [6]. Human safety must be fully guaranteed during any interaction to allow humans to enter robotic domains and for the widespread integration of robots in human environments [7]. In current industrial plants, human safety is guaranteed by rigid cages, or light curtains enclosing a robot's task space. Hereby, the robot motion is immediately stopped when a human enters the enclosure. Although this approach is safe in principle, no shared object manipulation or even the mere presence of humans in the vicinity of moving manipulators is allowed. Consequently, these approaches do not enable any human-robot collaboration and only very restricted coexistence [3] [8]. To facilitate closer human-robot collaboration, recent works have proposed control algorithms that limit the force or velocity of robots in the vicinity of humans [3]. Other approaches have adapted robot movements based on the distance between robots and operators [9] or use hand-guiding [10]. However, they do not utilize formal methods, so they cannot provide formal safety guarantees for humans in the workspace.

We therefore present our C++-based tool SaRA for Safe Human-Robot Coexistence and Collaboration through Reachability Analysis that formally determines the entire possible space that the human can occupy in a given time horizon in form of a reachable set as described in [7], [11]. A visualization of reachable sets is shown in Fig. 1. These sets can be used to formally guarantee safe human-robot interaction, e.g., as specified in ISO 15066 [12], as we have demonstrated in our previous work:


(a)

(b)

Fig. 1: (a) Human with motion tracking gear is performing a task next to the controlled robot. (b) Snapshot of the corresponding reachable sets of the human (blue) and the robot (pink); the reachable set of the human grows within one verification cycle as shown later.

- In [7], the human reachable sets are used to guarantee that a robot must be at rest before it is touched by humans. This is achieved through fail-safe planning, where a fail-safe maneuver is constantly updated, such that predicted robot occupancies do not intersect with the predicted human reachable sets, before the robot reaches a resting position. This approach conforms to speed and separation monitoring as defined in ISO 15066 [12].
- In [13], the human reachable sets are further used in a fail-safe planning approach, to reduce the robot speed such that collision force limits can be guaranteed. That approach conforms to power and force limiting as defined in ISO 15066 [12].

A. Contribution

Our open-source tool SaRA performs reachability analysis to provide formally guaranteed safety in human-robot coexistence and collaboration. Our stand-alone tool consists of a header-only C++ library (`ReachLib`) and a ROS-package[1] (`reachable_occupancy`). `ReachLib` represents human reachable sets as a set of capsules (as described in Sec. II-A). The library provides methods that perform set-based reachability analysis of humans over a given time horizon in only a few microseconds and can be used to determine intersections between reachable sets of humans and robots. `ReachLib` can be integrated within any C++ compatible environment. The ROS-package `reachable_occupancy` provides visualizations of the reachable sets calculated by `ReachLib` in RViz.

The authors are with the Department of Informatics, Technical University of Munich, 85748 Garching, Germany. `schepp@in.tum.de`, `jakob.thumm@tum.de`, `stefan.liu@tum.de`, `althoff@tum.de`

[1] Our package is available at: `https://github.com/Sven-Schepp/SaRA`.

B. Related work

We first review approaches to safe human-robot coexistence and collaboration that do not provide formal safety guarantees, subsequently, relevant formally safe approaches are discussed.

a) Non-formal methods: A general overview of non-formal safety mechanisms for human-robot collaboration, including predictive and reactive control methods, is provided in [3]. The works in [9] and [14] use minimum distance metrics. However, the authors did not explain how to formally derive the necessary clearance given the uncertainty of human and robot movements. Reaction-based safety strategies to reduce collision impacts have been presented in [15]. Predictive approaches aim to improve safety by anticipating human motion during deployment in highly dynamic environments using vision based systems, as described in [16]. The work in [17] presented an in-depth review of color and depth-based methods, including various active vision systems, for collision avoidance, the determination of distance between obstacles, or human intent recognition. An approach combining intent-driven and robust models has been presented in [18]. Furthermore, safety can also be provided intrinsically using soft robots [19], [20].

b) Formal methods: In our previous work in [7], [11], and [21] we have presented an approach that guarantees human safety using reachability analysis of robots and the upper body of humans. This approach constantly updates a fail-safe maneuver which formally guarantees the absence of collisions (see Sec. II-C). Similar methods for providing safe motions of mobile robots and autonomous vehicles have been presented in [22], [23], and [24]. The latter two approaches also considered occluded regions in their path planning by assuming them as initial positions of obstacles (unseen or phantom objects) approaching at maximum velocity. The work in [25] used a capsule representation for robot reachability analysis to achieve a real-time capable receding horizon trajectory planner that considers only static obstacles. With our software framework, their approach could be extended to also enable safe trajectory planning in human environments. To this point, there is no tool for computing over-approximative reachable sets of the entire human body considering individual joints.

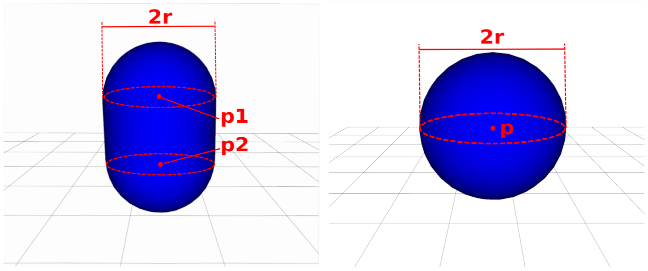
C. Article structure

We present three models for reachability analysis and a description of our self-verification procedure in Sec. II. Sec. III describes the structure and interfaces of our library, which is then evaluated in Sec. IV. Finally, we provide conclusions, including future work, in Sec. V.

II. Models

A. Reachability analysis

The reachability analysis is based on our previous work in [7], [11]. We calculate sets that enclose all possible future occupancies of a human over a time interval $[t_0, t_f]$. To formally introduce human reachable sets, let us first define $\boldsymbol{x}_0$ as the initial human state, $\boldsymbol{u}(\cdot)$ as a possible input trajectory, and $\chi(t; \boldsymbol{x}_0; \boldsymbol{u}(\cdot))$ as a human trajectory at time t given $\boldsymbol{x}_0$ and $\boldsymbol{u}(\cdot)$. The reachable set $\mathcal{R}(t)$ of a human at time t for a set of initial states $\mathcal{X}_0$ and a set of possible inputs $\mathcal{U}$ is computed as in [7, p. 5] by



(a)

(b)

Fig. 2: Visualizations of a capsule (a) and a ball (b).

$$\mathcal{R}(t) = \{\chi(t; \boldsymbol{x}_0, \boldsymbol{u}(\cdot)) \mid \boldsymbol{x}_0 \in \mathcal{X}_0, \forall t : \boldsymbol{u}(t) \in \mathcal{U}\}\,.$$

Reachable sets enclosing all possible trajectories over the interval $[t_0, t_f]$ are given by $\mathcal{R}([t_0, t_f]) = \bigcup\limits_{t \in [t_0, t_f]} \mathcal{R}(t)$. $\mathcal{R}([t_0, t_f])$ is calculated in Cartesian task space, where we choose w.l.o.g. $t_0 = 0$ and $t_f = t_{\text{brake}} + t_{\text{delay}}$, where t_{brake} represents the time a robot needs to come to a full stop after a system-intrinsic delay t_{delay}. Since we only verify collision avoidance, our reachable sets are only computed for the position domain. In this work, we represent reachable sets by a union of capsules to realize efficient intersection checking during verification, as described in Sec. IV. Capsules can be defined as the Minkowski sum[2] of a convex hull ($conv$) of two points $\mathcal{P} = \{P_1, P_2\}$ and a ball ($\mathcal{B}(r)$) of radius r: $\mathcal{C} = conv(P_1, P_2) \oplus \mathcal{B}(r)$; see Fig. 2.

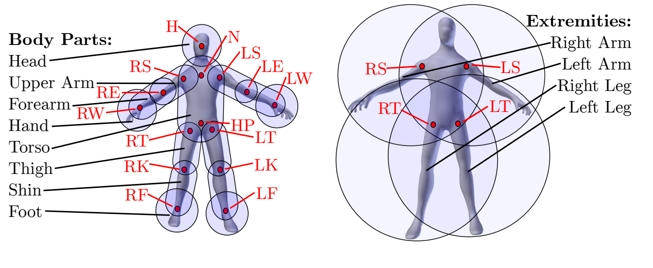
B. Model description

We present models of first-order and second-order considering position, velocity, and acceleration constraints. Unifying these models would be unnecessarily complex for reachability analysis. In contrast, we compute over-approximative reachable sets for each model: $\mathcal{R}_{\text{POS}}$ for position constraints, $\mathcal{R}_{\text{VEL}}$ for velocity constraints, and $\mathcal{R}_{\text{ACC}}$ for acceleration constraints. If for each time interval, any one of these sets does not intersect the occupancy of the robot, the robot movement is verified as safe. $\mathcal{R}_{\text{VEL}}$ and $\mathcal{R}_{\text{ACC}}$ consist of 14 body parts as shown in Fig. 3a, where the ends of the kinematic chain (hands, feet, and the head) are not enclosed by a capsule, but by the special case of a ball. The reachable set of a body part given the first-order model that considers the velocity constraints of each joint is computed as in [7, p. 6] by

$$\mathcal{R}_{\text{VEL}}(t) = \boldsymbol{y}(0) \oplus \mathcal{B}(\delta y) \oplus \mathcal{B}(v_{\max} \cdot t),$$

where $v_{\max}$ represents the maximum velocity of a body part, $\boldsymbol{y}(0)$ the initial position, and δy the measurement uncertainties of the position. The second-order model requires the

[2] $\mathcal{U} \oplus \mathcal{V} = \{\boldsymbol{u} + \boldsymbol{v} | \boldsymbol{u} \in \mathcal{U}, \boldsymbol{v} \in \mathcal{V}\}$.



(a)

(b)

Fig. 3: (a) Depiction of the full-body human reachable set with capsules in blue and joint positions in red as calculated for $\mathcal{R}_{\text{VEL}}$ and $\mathcal{R}_{\text{ACC}}$. (b) Balls as calculated for $\mathcal{R}_{\text{POS}}$; for improved visibility the balls are moved slightly off center.

current velocity $\dot{\boldsymbol{y}}$ and is computed as in [7, p. 6], using

$$\mathcal{R}_{\text{ACC}}(t) = \boldsymbol{y}(0) \oplus \mathcal{B}(\delta y) \oplus \dot{\boldsymbol{y}} \cdot t \oplus \mathcal{B}(\delta \dot{y} \cdot t) \oplus \mathcal{B}\left(\frac{a_{\text{max}}}{2} \cdot t^2\right)$$

where a_{max} represents the maximum acceleration of a body part and $\delta \dot{y}$ describes the measurement uncertainties of the velocity. $\mathcal{R}_{\text{VEL}}(t)$ and $\mathcal{R}_{\text{ACC}}(t)$, as previously presented, must be computed for all body parts as shown in Fig. 3a. As an example for the reachable set of a leg, we introduce $\mathcal{R}_{\text{T}}(t)$, $\mathcal{R}_K(t)$, and $\mathcal{R}_F(t)$ as the reachable sets of the thigh, knee, and foot joint. The radii of the enclosing capsules of the thigh, shin, and foot are respectively given by r_{thigh}, r_{shin}, and r_{foot}, resulting in the capsules

$$\begin{aligned}
\mathcal{R}_{\text{thigh}}(t) &= conv(\mathcal{R}_{\text{T}}(t), \mathcal{R}_{\text{K}}(t)) \oplus \mathcal{B}(r_{\text{thigh}}(t)), \\
\mathcal{R}_{\text{shin}}(t) &= conv(\mathcal{R}_{\text{K}}(t), \mathcal{R}_{\text{F}}(t)) \oplus \mathcal{B}(r_{\text{shin}}(t)), \\
\mathcal{R}_{\text{foot}}(t) &= \mathcal{R}_{\text{T}}(t) \oplus \mathcal{B}(r_{\text{foot}}(t)).
\end{aligned}$$

Finally, the reachable set of the leg is computed as $\mathcal{R}_{\text{leg}}(t) = \mathcal{R}_{\text{thigh}}(t) \cup \mathcal{R}_{\text{shin}}(t) \cup \mathcal{R}_{\text{foot}}(t)$. Then, the full-body reachable sets $\mathcal{R}_{\text{VEL}}(t)$ and $\mathcal{R}_{\text{ACC}}(t)$ are accordingly given by the union of the reachable sets of all body parts.

$\mathcal{R}_{\text{POS}}(t)$ consist of four capsules, which are defined by the two shoulder and both thigh positions, as shown in Fig. 3b. These describe the reachable sets of the right arm $\mathcal{R}_{\text{r,arm}}(t)$, left arm $\mathcal{R}_{\text{l,arm}}(t)$, right leg $\mathcal{R}_{\text{r,leg}}(t)$, and left leg $\mathcal{R}_{\text{l,leg}}(t)$. The model for computing $\mathcal{R}_{\text{POS}}(t)$ considers positional limits p_{max} of a human for the length of fully stretched arms and legs. The centers of the balls in Fig. 3b expand with the maximum velocity of the shoulder joint $v_{\text{S, max}}$ and thigh joints $v_{\text{T, max}}$, respectively. The reachable set of one leg is then computed as in [7, p. 6] by

$$\begin{aligned}
\mathcal{R}_{\text{POS, leg}}(t) = \boldsymbol{y}_T(0) \oplus \mathcal{B}(v_{\text{T,max}} \cdot t + \|\boldsymbol{y}_T(0) - \boldsymbol{y}_K(0)\| \\
+ \|\boldsymbol{y}_K(0) - \boldsymbol{y}_F(0)\| + \delta y + r_{\text{foot}}),
\end{aligned}$$

where $\boldsymbol{y}_T$, $\boldsymbol{y}_K$, and $\boldsymbol{y}_F$ represent the current positions of the joints of the thigh, knee, and foot. The position-constrained reachable sets are then computed as in [7, p. 6] by

$$\mathcal{R}_{\text{POS}}(t) = \mathcal{R}_{\text{r,arm}}(t) \cup \mathcal{R}_{\text{l,arm}}(t) \cup \mathcal{R}_{\text{r,leg}}(t) \cup \mathcal{R}_{\text{l,leg}}(t).$$

C. *Self-verification*

For self-verification, reachable sets of humans are subsequently checked for intersections with the robot's reachable set $\mathcal{R}_{\text{ROB}}$. Over-approximative capsules enclosing the robot's motion can be calculated with the approach presented in [26, Sec. IV A]. The robot's current maneuver is proven safe should one of the reachable sets $\mathcal{R}_{\text{POS}}$, $\mathcal{R}_{\text{VEL}}$, or $\mathcal{R}_{\text{ACC}}$ not intersect with $\mathcal{R}_{\text{ROB}}$ for all $t \in [t_0, t_f]$. The self-verification procedure implemented in our robot's controller is described in detail in [11] and [21, Sec. 2] and will be integrated into SaRA in the future.

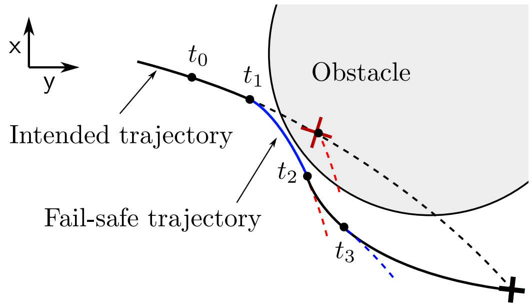


Fig. 4: Planning of intended trajectories (black) and fail-safe trajectories (blue, red). The robot starts at t_0 with the trajectory from t_0 to t_1 verified as safe and a fail-safe trajectory starting at time t_1. The intended trajectory from t_1 to t_2 (dashed black) is not verified as safe, thus the fail-safe trajectory (blue) is executed. Within $[t_1, t_2]$, another intended (solid black) and fail-safe (dashed red) trajectory are calculated for the combined time interval $[t_2, t_3 + t_f]$. The trajectory from t_2 to t_3 (solid black) including its fail-safe trajectory (dashed blue) is verified as safe and executed. The robot finally returns back to its originally intended trajectory.

By induction, we can formally guarantee safety for an infinite time horizon. Starting from a safe state at time t_i (see Fig. 4), the intended trajectory is calculated for $t \in [t_{i+1}, t_{i+2}]$, as well as a fail-safe trajectory for $t \in [t_{i+2}, t_{i+2} + t_{f,i+2}]$, where $t_{f,i}$ is the value of t_f starting at time t_i. If the intended trajectory and its subsequent fail-safe trajectory for the combined interval $[t_{i+1}, t_{i+2} + t_{f,i+2}]$ cannot be verified as safe, the previous fail-safe trajectory calculated for $[t_{i+1}, t_{i+1} + t_{f,i+1}]$ is executed instead. Therefore, the robot will stay on the fail-safe trajectory until an intended trajectory is verified as safe to return to its originally intended trajectory. Our approach thereby requires current measurements of the Cartesian position[3] of human joints at every time step i. SaRA is agnostic with respect to the method of obtaining joint positions; in our previous work we have tested our self-verification approach using a motion capture system [7], light curtains [21, Sec. 4.1], a Kinect camera, and depth sensors [21, Sec. 4.2].

III. Implementation

SaRA consists of a hierarchical structure with three abstraction layers designed to facilitate the extension and addition of further set calculation methods. By introducing an intermediate layer, we can model a reachable set as a union of multiple smaller reachable sets, e.g., humans can be defined as a union of body parts. This approach facilitates the implementation of models with variable granularity since a reachable set can be defined by any number of partial

[3] The second-order model additionally requires Cartesian velocities.

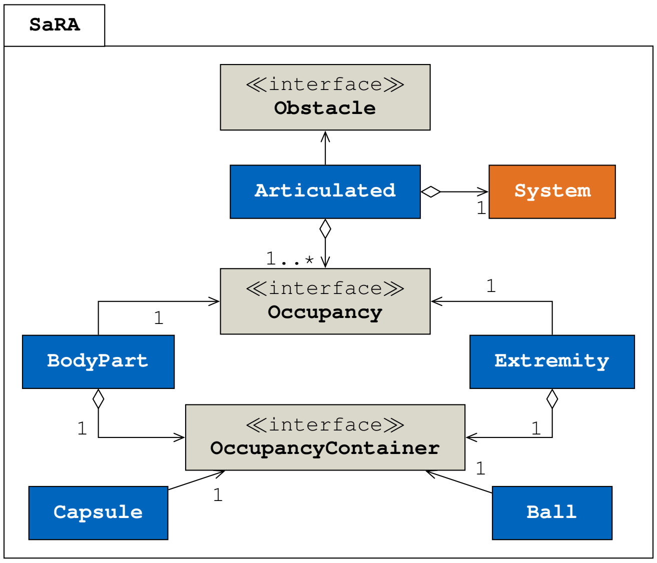


Fig. 5: UML representation of the `ReachLib` library. The blue classes are specifically designed for human reachable sets and can be changed or expanded upon. The interfaces describe the general structure of our library such that new models can be added.

reachable sets. A UML description of the general library structure is shown in Fig. 5.

At the highest level, we have the generic interface `Obstacle`, describing properties that any model must contain. Every `Obstacle` must include at least one reachable set. Additionally, parameters for measurement uncertainties and delay have to be provided as described in Sec. IV-A. Furthermore, a method for determining intersections with obstacle occupancies must be defined. The human reachable sets detailed in Sec. II represent a type of `Obstacle`. $\mathcal{R}_{\text{POS}}$, $\mathcal{R}_{\text{VEL}}$, and $\mathcal{R}_{\text{ACC}}$ are reachable sets of `Articulated` models representing body parts. Further, additional models of type `Articulated` can be defined by the user, such as intent-driven approaches as described in [18].

The second abstraction layer, defined by the `Occupancy` interface, describes the general properties of reachable sets. Every reachable set, must therefore, be named and include a form of `OccupancyContainer` by which a reachable set is geometrically described. All types of `Occupancy` and `OccupancyContainer` must further provide an *update* method that updates the geometry of the reachable set given current measurements. Implementations of the `Occupancy` class are also required to provide a method for intersection checking. In the case of representing a human, `BodyPart` and `Extremity` are considered implementations of the generic `Occupancy` class. They represent the reachable sets of the `Articulated` models and contain capsules individually updated at every step.

The last layer, `OccupancyContainer`, defines the geometric representations of a reachable set in Cartesian space. The `Capsule` and `Ball` classes define geometric representations of the reachable sets of `BodyPart` and `Extremity`, respectively. These could be exchanged for other set representations, such as cylinders, polyhedra, or zonotopes in the future. Thus, existing models can be reused and equipped with different geometric representations of reachable sets.

TABLE I:
Human parameters for reachable set calculation from [7].

Body Part	a_{max} $[\text{m}\,\text{s}^{-2}]$	v_{max} $[\text{m}\,\text{s}^{-1}]$	r_{b} [m]
Head	25.0	2.0	0.3
Torso	20.0	2.0	0.3
Upper/Lower Arm	50.0	2.0	0.1
Hand	50.0	2.0	0.205

The models $\mathcal{R}_{\text{POS}}$, $\mathcal{R}_{\text{VEL}}$, and $\mathcal{R}_{\text{ACC}}$ are provided with the data described in Tab. I during the initial setup, where r_b represents the measured radius of the corresponding body part. These data are taken from our previous work in [7]. The added measurement uncertainties and delay of all components in our setup are defined in the `System` class. The reachable sets can then be acquired in the form of a list of type `OccupancyContainer` contained within each `Articulated` model's instance[4] following an update step.

IV. Experiments

A. Setting

Real-world experiments were conducted to explore the capabilities of SaRA. The setup includes a *Schunk LWA 4P* manipulator with six degrees of freedom continuously moving along a pre-planned trajectory. Human movements are tracked using a motion capture system[5]. The Cartesian positions of joints are calculated from marker positions. The parameters used for reachable set calculation are listed in Tab. I. We use a time-varying braking time t_{brake}, based on the current velocity and acceleration of the robot. The braking time can alternatively be set as the constant maximum braking time of the manipulator, which, however, will lead to more conservative behavior. The maximum t_{brake} during our experiments was $\text{t}_{\text{brake\_max}} = 0.528\,\text{s}$ and the average was $\text{t}_{\text{brake\_avg}} = 0.187\,\text{s}$.

Our experiments only include upper-body reachable sets of humans, since our setup, shown in Fig. 6, cannot cause collisions between the robot and the human's lower body. The captured points are therefore: H, N, LS, LE, LW, RS, RE, and RW, as given in Fig. 3a.

We executed the following motions:

- A slow approach towards the robot to test common upper-body motions.
- Punches towards the end effector to test hand safety during fast motions.
- A headbutt towards the manipulator to test head safety.
- A box is picked up and placed next to the robot by the human as an example for common industrial use cases.

[4] A more detailed explanation and integration description is given in our manual at: `https://github.com/Sven-Schepp/SaRA`.

[5] The motion capture hardware and software is described at: `https://www.vicon.com/hardware/cameras/vero/`.

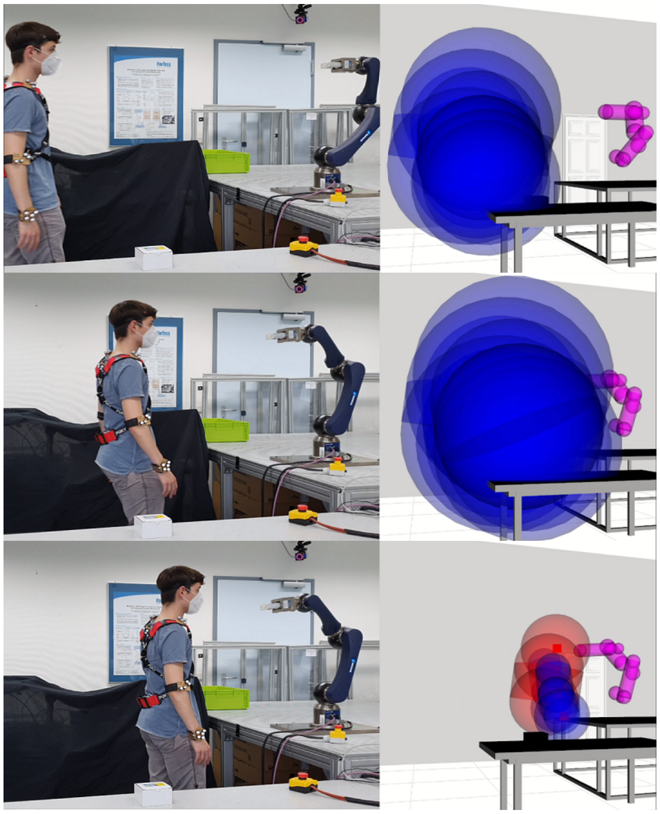


Fig. 6: Example frames from the approaching experiment on the left (from top to bottom) with respective $\mathcal{R}_{\text{VEL}}$ on the right. Here, the necessary time horizons are dynamically computed and depend on the duration of the fail-safe maneuver. In the above frames, they are $t_f = 0.244$, $t_f = 0.416$,and $t_f = 0.010$, respectively.

We provide a video[6] as supplementary material to this paper in which all experiments are demonstrated. Snapshots of the human approaching the robot are shown in Fig. 6, where the reachable sets of the human change from blue to red once an intersection with the reachable sets of the robot (pink) is detected. The size of the human reachable sets are large in the beginning since both human and robot are in motion. They grow continuously while both human and robot are accelerating. Finally, intersections between the reachable sets of the head and the robot $\mathcal{R}_{\text{ROB}}$, as well as between the reachable sets of the torso and $\mathcal{R}_{\text{ROB}}$ are detected, resulting in a full stop of the manipulator. The main factor determining the size of reachable sets is t_{brake}, in addition to the velocity of the human for $\mathcal{R}_{\text{ACC}}$.

We investigate the effects of t_{brake} on the size of reachable sets by calculating them for the recording `02_01` of the Carnegie Mellon University MoCap[7] library. Thereby, larger values for $t_f = t_{brake} + t_{delay}$ resulted in larger reachable sets as shown in Fig. 7.

The values for v_{max} and a_{max} are taken from [7]. The measurement uncertainties of the position and velocity are assumed as $\delta y = 0.004m$ and $\delta \dot{y} = 0.04\frac{m}{s}$, respectively, and the delay is assumed as $t_{delay} = 0.010s$ [11].

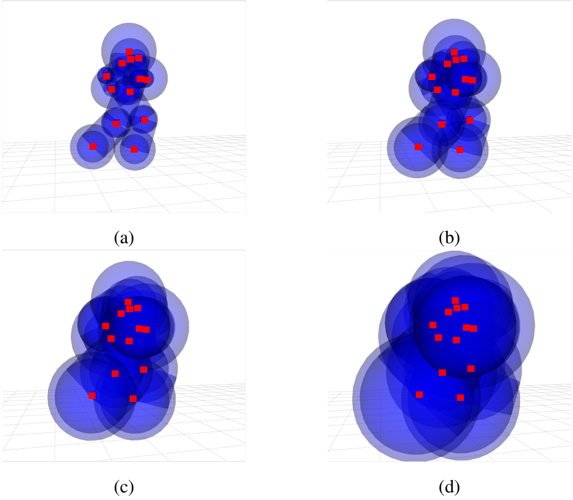


Fig. 7: Full-body reachable sets are calculated for $t_f = 0.16\,\text{s}$ in (d), with figures (a)–(c) displaying sets calculated for intermediate times. The joint positions are indicated by red squares.

B. Results

In all experiments, SaRA successfully detects all potential collisions and stops the robot beforehand. All marker positions are contained within the reachable sets of $\mathcal{R}_{\text{POS}}$, $\mathcal{R}_{\text{VEL}}$, and $\mathcal{R}_{\text{ACC}}$. We analyze the computation time[8] of all sets individually and combined within all four experiments. We only show the results for the headbutt experiment here, since it had the highest average calculation times[9]. Tab. II shows the time SaRA requires to compute $\mathcal{R}_{\text{POS}}$, $\mathcal{R}_{\text{VEL}}$, and $\mathcal{R}_{\text{ACC}}$. The results in Tab. III describe the time SaRA requires to both calculate the reachable sets and perform intersection checking with the robot's reachable sets.

All models are calculated in a few microseconds, even when intersection operations are considered. The individual models are computed in less than $62\,\mu\text{s}$. If intersections are included, all individual models and intersections are calculated in less than $67\,\mu\text{s}$. Further, when combined, all models are calculated in less than $90\,\mu\text{s}$ on average. This increases to a maximum of $98\,\mu\text{s}$, if intersections of all sets with the robot's reachable sets are additionally considered. However, these values depend on how often the reachable sets intersect, since as soon as one model is found to have no intersections with the robot's reachable sets, none of the remaining models need to be checked.

C. Discussion

Since the sampling time of standard robot controllers ranges from 500 to 4000 µs, the computation times of the reachable sets are not time critical and we can even consider

[6] The video is available at: `https://www.youtube.com/watch?v=QDYq_FQL1Ds`.

[7] The human motion capture data used in this experiment was obtained from `http://mocap.cs.cmu.edu/`. The database was created with funding from NSF EIA-0196217.

[8] All calculations were performed on an i7-9750H (16 GB RAM), where only those operations required for reachable set computation and intersection checking were included during timing.

[9] All other results can be found here: `https://github.com/Sven-Schepp/SaRA`.

TABLE II:
Headbutt experiment: calculation time no intersections.

Model	avg. [µs]	std. [µs]	max [µs]	min [µs]
$\mathcal{R}_{\text{POS}}$	2.1	2.0	46	1
$\mathcal{R}_{\text{VEL}}$	4.7	2.7	62	3
$\mathcal{R}_{\text{ACC}}$	6.3	3.1	48	4
All Models	17.6	7.3	82	2

TABLE III:
Headbutt experiment: calculation time with intersections.

Model	avg. [µs]	std. [µs]	max [µs]	min [µs]
$\mathcal{R}_{\text{POS}}$	3.2	2.8	29	2
$\mathcal{R}_{\text{VEL}}$	10.8	5.5	66	7
$\mathcal{R}_{\text{ACC}}$	15.5	5.7	67	11
All Models	22.2	8.7	98	16

multiple humans. The question, which model should be computed, however, depends on the inputs, which can be provided to SaRA:

- If only the position of the human joints are given, then only $\mathcal{R}_{\text{POS}}$ and $\mathcal{R}_{\text{VEL}}$ can be computed.
- If also the velocity of the human joints are given, then $\mathcal{R}_{\text{ACC}}$ can be additionally computed, making our self-verification less conservative.

V. Conclusion

Our real-world experiments show that SaRA effectively performs reachability analysis for guaranteed safety in human-robot coexistence and collaboration. SaRA produces human reachable sets and performs intersection checking in only a few microseconds. We believe that integrating SaRA in manufacturing environments can significantly improve production efficiency by removing caging. We regard SaRA as a fundamental step toward allowing humans and robots to coexist and collaborate in any environment. Our next goals entail real-world use of SaRA in a manufacturing plant for collaborative studies with trained workers and the consideration of occlusions.

ACKNOWLEDGMENT

The authors gratefully acknowledge financial support by the Central Innovation Programme of the German Federal Government under grant ZF4086004LP7 and the Horizon 2020 EU Framework Project CONCERT under grant 101016007.

References

[1] J. Burgner-Kahrs, D. C. Rucker, and H. Choset, "Continuum robots for medical applications: A survey," *IEEE Trans. on Robotics*, vol. 31, no. 6, pp. 1261–1280, 2015.

[2] A. Bertelsen, J. Melo, E. Sanchez, and D. Borro, "A review of surgical robots for spinal interventions," *Int. J. of Medical Robotics and Computer Assisted Surgery*, vol. 9, no. 4, pp. 407–422, 2013.

[3] E. Matheson, R. Minto, E. Zampieri, M. Faccio, and G. Rosati, "Human–robot collaboration in manufacturing applications: A review," *Robotics*, vol. 8, no. 4, 2019.

[4] A. Djuric, R. Urbanic, and J. Rickli, "A framework for collaborative robot (CoBot) integration in advanced manufacturing systems," *SAE Int. J. of Materials and Manufacturing*, vol. 9, no. 2, pp. 457–464, 2016.

[5] S. Pedersen, S. Fountas, and H. Have, "Agricultural robots–system analysis and economic feasibility," *Precision Agriculture*, vol. 7, pp. 295–308, 2006.

[6] L. Cousineau and N. Miura, *Construction Robots: the Search for New Building Technology in Japan*. Reston, Virginia: ASCE Press, 1994.

[7] M. Althoff, A. Giusti, S. B. Liu, and A. Pereira, "Effortless creation of safe robots from modules through self-programming and self-verification," *Science Robotics*, vol. 4, no. 31, 2019.

[8] I. Aaltonen, T. Salmi, and I. Marstio, "Refining levels of collaboration to support the design and evaluation of human-robot interaction in the manufacturing industry," *Procedia CIRP*, vol. 72, pp. 93–98, 2018.

[9] A. M. Zanchettin, N. M. Ceriani, P. Rocco, H. Ding, and B. Matthias, "Safety in human-robot collaborative manufacturing environments: Metrics and control," *IEEE Trans. on Automation Science and Engineering*, vol. 13, no. 2, pp. 882–893, 2016.

[10] S. Zhang, S. Wang, F. Jing, and M. Tan, "A sensorless hand guiding scheme based on model identification and control for industrial robot," *IEEE Trans. on Industrial Informatics*, vol. 15, no. 9, pp. 5204–5213, 2019.

[11] A. Pereira and M. Althoff, "Overapproximative human arm occupancy prediction for collision avoidance," *IEEE Trans. on Automation Science and Engineering*, vol. 15, no. 2, pp. 818–831, 2017.

[12] ISO/TS 15066:2016, "Robots and robotic devices - collaborative robots," Int. Org. for Standardization, Geneva, Switzerland, 2016.

[13] S. B. Liu and M. Althoff, "Online Verification of Impact-Force-Limiting Control for Physical Human-Robot Interaction," in *IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, 2021, pp. 777–783.

[14] B. Lacevic, P. Rocco, and A. Zanchettin, "Safety assessment and control of robotic manipulators using danger field," *IEEE Trans. on Robotics*, vol. 29, no. 5, pp. 1257–1270, 2013.

[15] S. Haddadin, A. Albu-Schaffer, A. D. Luca, and G. Hirzinger, "Collision detection and reaction: A contribution to safe physical human-robot interaction," in *IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, 2008, pp. 3356–3363.

[16] P. A. Lasota, T. Fong, and J. A. Shah, "Safety through prediction," in *A Survey of Methods for Safe Human-robot Interaction*. Boston-Delft: Now Foundations and Trends, 2017, vol. 5, no. 4, pp. 293–313.

[17] R. Halme, M. Lanz, J. Kämäräinen, R. Pieters, J. Latokartano, and A. Hietanen, "Review of vision-based safety systems for human-robot collaboration," *Procedia CIRP*, vol. 72, pp. 111–116, 2018.

[18] A. Bajcsy, S. Bansal, E. Ratner, C. J. Tomlin, and A. D. Dragan, "A robust control framework for human motion prediction," *IEEE Robotics and Automation Letters*, vol. 6, no. 1, pp. 24–31, 2020.

[19] S. Kim, C. Laschi, and B. Trimmer, "Soft robotics: a bioinspired evolution in robotics," *Trends in Biotechnology*, vol. 31, no. 5, pp. 287–294, 2013.

[20] R. Qi, T. L. Lam, and Y. Xu, "Mechanical design and implementation of a soft inflatable robot arm for safe human-robot interaction," in *IEEE Int. Conf. on Robotics and Automation*, 2014, pp. 3490–3495.

[21] P. Aaron, "Guaranteeing safe robot motion," Ph.D dissertation, Technical University Munich, Munich, 2018, Accessed on: Feb. 28, 2022. [Online]. Available: https://mediatum.ub.tum.de/doc/1443612/1443612.pdf.

[22] S. B. Liu, H. Röhm, C. Heinzmann, I. Lütkebohle, J. Oehlerking, and M. Althoff, "Provably safe motion of mobile robots in human environments," in *IEEE/RSJ Int. Conf. Intelligent Robots and Systems*, 2017, pp. 1351–1357.

[23] S. Bouraine, T. Fraichard, and H. Salhi, "Provably safe navigation for mobile robots with limited field-of-views in dynamic environments," *Autonomous Robots*, vol. 32, no. 3, pp. 267–283, 2012.

[24] M. Koschi and M. Althoff, "Set-based prediction of traffic participants considering occlusions and traffic rules," *IEEE Trans. on Intelligent Vehicles*, vol. 6, no. 2, pp. 249–265, 2021.

[25] P. Holmes, S. Kousik, B. Zhang, D. Raz, C. Barbalata, M. Johnson-Robertson, and R. Vasudevan, "Reachable sets for safe, real-time manipulator trajectory design," in *Robotics: Science and Systems*, 2020.

[26] D. Beckert, A. Pereira, and M. Althoff, "Online verification of multiple safety criteria for a robot trajectory," in *IEEE 56th Annual Conf. on Decision and Control (CDC)*, 2017, pp. 6454–6461.