Provably Safe Motion of Mobile Robots in Human Environments

Stefan B. Liu*†, Hendrik Roehm†, Christian Heinzemann†, Ingo Lütkebohle†,
Jens Oehlerking†, and Matthias Althoff*
*Department of Informatics, Technische Universität München, Germany, Email: [stefan.liu | althoff]@tum.de
†Robert Bosch GmbH, Corporate Research, Renningen, Germany, Email:[firstname.lastname]@de.bosch.com

***Abstract*— Mobile robots operating in a shared environment with pedestrians are required to move provably safe to avoid harming pedestrians. Current approaches like safety fields use conservative obstacle models for guaranteeing safety, which leads to degraded performance in populated environments. In this paper, we introduce an online verification approach that uses information about the current pedestrian velocities to compute possible occupancies based on a kinematic model of pedestrian motion. We demonstrate that our method reduces the need for stopping while retaining safety guarantees, and thus goals are reached between 1.4 and 3.5 times faster than the standard ROS navigation stack in the tested scenarios.**

I. Introduction

Mobile service robots often need to operate freely and flexibly in environments occupied by pedestrians. Because collisions could cause serious harm, particularly in settings with heavier robots, safety mechanisms always have to be considered. Despite the large body of work in path planning and obstacle avoidance [1], [2], in practice, most production robots still rely on hardware safety devices such as certified laser scanners. The main reason is that demonstrating the safety of software to the satisfaction of a safety body is difficult, and difficulty scales with the complexity of the algorithm.

In environments with none or only a few humans, a common way to reduce the problem is through a simple model of human motion: Either assume people will always stop (ISO 3691-4 [3]), or assume they always move at full speed (ISO 13855 [4] and ISO 13482 [5]). The latter is usually applied and results in a circular safety area as illustrated in the left part of Fig. 1.

Unfortunately, a circular field seriously restricts robot motion, including in areas which common sense would indicate as usable, such as beside or following a walking pedestrian. In more populated environments, it leads to frequent stopping of the robot and is therefore almost unusable.

This paper, in contrast, proposes a safeguard that predicts all possible motions based on a kinematically-accurate model of human motion, as well as the humans' current position and velocity. It guarantees the same level of safety but allows much more efficient motion.

Specifically, we compute so-called *reachable sets* that include all possible future occupancies of pedestrians and the robot based on their kinematic models. Based on these sets, we consider a velocity command as verified safe if the robot can stop before entering the reachable set of any pedestrian, i.e., no collision can occur before the robot stops (passive safety [6]). From the example reachable sets shown on the right side of Fig. 1 it is immediately obvious that they leave much more maneuvering space compared to the static approach on the left.

While our current kinematic model considers walking pedestrians only, our approach is extensible to multiple models, e.g., to other dynamic behaviors (running, wheelchairs, etc.) as well as to structured environments. For instance, reachability analysis has also already been used in other safety-related applications such as autonomous driving [7] and robot manipulators [8].

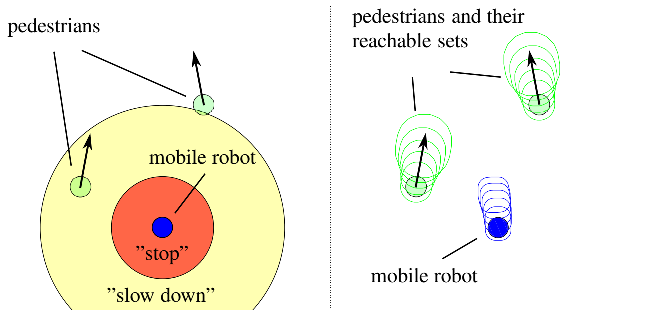

Fig. 1: Standard safety field (left) compared to our method.

We evaluate the performance of our approach regarding safety and efficiency in a Gazebo-based simulation environment which combines the standard ROS navigation stack[1] with a safeguard based on either i) the one proposed in ISO 13482, ii) an obstacle model of the *braking ICS* approach [9], or iii) our method. We use three different scenarios and two pedestrian densities. Here, our approach improves the average velocity by a factor of 1.4 – 3.5. In absolute numbers, the average velocities for an easy situation are 1 m/s (our approach) compared to 0.64 m/s (ISO 13482), or, more striking for a difficult situation, 0.92 m/s vs 0.27 m/s.

In the following, we first outline the main ideas of reachability analysis, passive safety, and reachable set conformance in Sect. II. In Sect. III, we describe the modeling and online verification approach. Sect. IV shows the evaluation results. Works related to our approach are described in Sect. V.

II. Preliminaries / Definition of Terms

In this section we briefly introduce the main methods and terms relevant to our approach.

[1]`http://www.ros.org/navigation`

A. Reachability analysis of continuous systems

We model a system as a differential equation $\dot{x}(t) = f(x(t), u(t))$, where t, x and u are time, state, and input respectively. The initial state $x(0)$ can be chosen arbitrarily within the initial set $\mathscr{X}^0$. The time-dependent input trajectory $u(t)$ is allowed to vary, but is assumed to stay in a time-dependent input trajectory set $\mathscr{U}(t)$. These two sources of non-determinism are used to model the fact that we do not exactly know the current positions and velocities of objects and their behavior in the future, see Sect. III-A.

Reachability analysis is the computation of reachable sets of states over time of such a model. For instance, the reachable sets for a robot and two pedestrians for different points in time are illustrated in Fig. 1. Reachable sets are formally defined as follows:

Definition 1 (Reachable Set (Reachset), see [7]). Given an initial set $\mathscr{X}^0$ and a time-dependent input trajectory set $\mathscr{U}(t)$, the reachable set $\mathscr{R}(t)$ at time t of a system of the form $\dot{x}(t) = f(x(t), u(t))$ is the set of all reachable states at time t:

$$\mathscr{R}(t) = \Big\{ x(t) = \int_0^t f(x(\tau), u(\tau)) d\tau + x(0) \Big| \\ x(0) \in \mathscr{X}^0, \forall \tau \in [0, t] : u(\tau) \in \mathscr{U}(\tau) \Big\}. \quad (1)$$

We are using CORA for efficient computation of reachable sets for high dimensional and nonlinear problems. Although CORA itself has yet to be proved formally in a theorem prover such as in [10], the method proposed in this work can be proven similarly.

For safety analysis it is very important to account for system uncertainties. Since we do this by using the non-determinism of the model, $\mathscr{R}(t)$ is a set containing *all* possible future states of the system at a time t. We verify safety by checking that there is never an intersection between reachable sets $\mathscr{R}(t)$ of our system and sets of unsafe states (e.g. position of surrounding objects, unsafe velocities, etc.).

B. Passive Safety & Safe Motion Trajectory

Passive safety means that no collisions with surrounding pedestrians are allowed to happen when the robot moves [11]. This is equivalent to ensuring a complete stop before a potential collision. We therefore choose an input trajectory that brings the robot to a stop. The trajectory is considered safe if the following property holds:

Definition 2 (Safe Motion Trajectory). An input trajectory $u(t)$ of a robot system that brings the robot to a safe stop at t_{stop} is safe according to passive safety, if

$$\forall t \in [0; t_{stop}] : \mathscr{R}_{ped}(t) \cap \mathscr{R}_{rob}(t) = \emptyset, \quad (2)$$

where $\mathscr{R}_{ped}(t)$ are pedestrian reachable sets and $\mathscr{R}_{rob}(t)$ are robot reachable sets with input trajectory $u(t)$.

C. Model Conformance

Validating that our pedestrian model conforms to real pedestrian behavior is very important for the overall verification technique. For our model-based results to hold in reality, our model of the pedestrian has to conform to real behavior. We therefore check our pedestrian model against real measured data before using it for verification.

Recently, we showed that for verifying the absence of collisions, *reachset conformance testing* is a suitable approach to check the conformance of models to real behavior [12].

Definition 3 (Reachset Conformance). Measured state data $p_1, \ldots, p_n$ of a pedestrian with timestamps $t_1, \ldots, t_n$ is reachset conformant to the pedestrian model, if the following holds:

$$\forall i : p_i \in \mathscr{R}_{ped}(t_i). \quad (3)$$

We evaluate reachset conformance in Sect. IV-A based on the pedestrian model introduced in the next section.

III. Modelling and Verification

In this section, we provide the models for the pedestrians and the robot that we employ for the reachable set computations and give an algorithm for the online computation of safe motion trajectories based on reachable sets.

A. Pedestrian Modeling

We model a single pedestrian as a point on a two-dimensional plane. The shape of the pedestrian is then taken into account after the reachable set computation by enlarging the reachable sets accordingly. We assume that we can measure the pedestrian's position and velocity with some known uncertainty. Also, we assume that the pedestrian performs a forward walking motion while possibly changing directions and that the pedestrian has a maximum speed and acceleration. We represent these constraints as two separate differential equation models: one constraining the acceleration and one constraining the velocity. Reachable states of the pedestrian are then states which are reachable under both models.

It would be possible to merge these two models into one that includes state constraints. This can be realized in CORA by a hybrid model, for which reachable sets are difficult to obtain. However, it has been shown in [13] that for reachability analysis it is possible to define multiple abstracting models, such that their reachable set intersection overapproximates the reachable sets of the hybrid model.

Therefore, we define the following two models. The acceleration-constrained model

$$\dot{p}_x = v_x, \qquad \dot{p}_y = v_y, \qquad \dot{v}_x = a_x, \qquad \dot{v}_y = a_y \\ \mathscr{U}_{ped}^{(a)} = \left\{ (a_x, a_y) \in \mathbb{R} \times \mathbb{R} \middle| a_x^2 + a_y^2 \leq a_{max}^2 \right\} \quad (4)$$

has the two-dimensional position p and velocity v as its state variables. The input trajectory is a time-invariant set representing all possible two-dimensional accelerations and bounded by a_{max}. The velocity-constrained model

$$\dot{p}_x = v_x, \qquad \dot{p}_y = v_y \\ \mathscr{U}_{ped}^{(v)} = \left\{ (v_x, v_y) \in \mathbb{R} \times \mathbb{R} \middle| v_x^2 + v_y^2 \leq v_{max}^2 \right\} \quad (5)$$

has only the two-dimensional position p as its state variables, while the velocity v is instead an input bounded by v_{max}.

The initial position $[p_x(0), p_y(0)]$ and initial velocity $[v_x(0), v_y(0)]$ are assumed to lie in the sets $\mathscr{X}_{ped}^{(a),0}$ and $\mathscr{X}_{ped}^{(v),0}$, respectively. Since both models are used to predict possible pedestrian behavior, their initial states can be interpreted as the currently measured position and velocity of the pedestrian, plus some assumed measurement uncertainty.

The reachable sets of a single pedestrian are obtained by computing the reachable sets $\mathscr{R}_{ped}^{(a)}(t)$ and $\mathscr{R}_{ped}^{(v)}(t)$ (Fig. 2) of both models and then taking their intersection $\mathscr{R}_{ped}(t) = \mathscr{R}_{ped}^{(a)}(t) \cap \mathscr{R}_{ped}^{(v)}(t)$. Lastly, we enlarge all $\mathscr{R}_{ped}(t)$ by a circle in the (p_x, p_y)-dimensions to account for the shape of the human; any other shape can also be used.

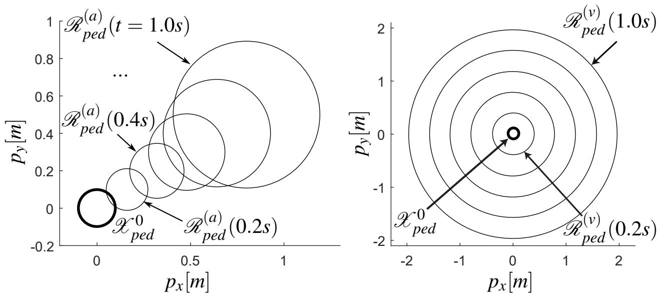

Fig. 2: Reachable sets according to the acceleration-constrained (left) and velocity-constrained (right) model.

B. Robot Modeling

For modeling the mobile robot we use a kinematic model of a differential-drive robot

$$\dot{p}_x = v_{tra}\cos(\phi), \qquad \dot{p}_y = v_{tra}\sin(\phi), \qquad \dot{\phi} = v_{rot}.$$

The initial state $[p_x(0), p_y(0), \phi(0)]^T$ represents the current pose of the robot and is bounded by an initial set $\mathscr{X}_{rob}^{0}$ that accounts for the inaccuracy of the robot's localization algorithm. The input of the system is the vector $[v_{tra}, v_{rot}]^T$, consisting of the translational and rotational velocities of the differential drive. For verification we consider that the input is not allowed to change at a larger rate than the maximum acceleration of the robot. In the same fashion as for the reachable sets of the pedestrians, we add the shape of the robot to the (p_x, p_y)-dimensions of all $\mathscr{R}_{rob}(t)$.

C. Online Motion Trajectory Verification

In our approach, we verify passive safety (Sect. II-B) of motion commands for every step k, where the sampling time is Δt. To this end, we employ reachability analysis to predict whether the robot can still come to a collision-free stop after applying the input $u_{plan}^{(k)} = [v_{tra}^{(k)}, v_{rot}^{(k)}]^T$ to the robot motors for a time step. For that we define a candidate input trajectory $u^*(t)$: It begins with the planned motion command $u^*(t) = u_{plan}$ for $t \in [0, \Delta t]$ and continues with $u^*(t) = u_{brk}(t)$ for $t \geq \Delta t$. The braking trajectory $u_{brk}(t)$ brings the robot to a stop $u(t_{stop}) = [0, 0]^T$ and the slope is the maximum deceleration of the robot system (see Fig. 3). Note that $t = 0$ always corresponds to the current time step in this analysis, while signal values and reachable sets at $t > 0$ correspond to (predictions of) future behavior.

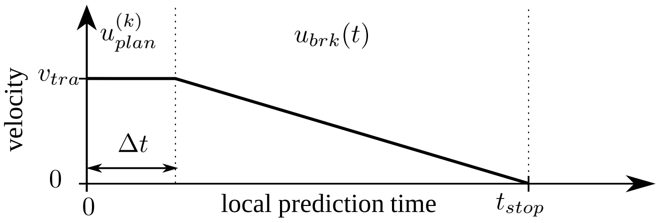

Fig. 3: Candidate input trajectory $u^*(t)$ that we use to compute $\mathscr{R}_{rob}(t)$, here shown for v_{tra}. v_{rot} is analogous.

We then compute pedestrian and robot reachable sets and verify $u^*(t)$ by checking whether $u^*(t)$ satisfies the property in Def. 2. If $u^*(t)$ is verified, we store $u_{safe}^{(k)}(t) := u^*(t)$ as a safe input trajectory and apply $u_{safe}^{(k)}(t)$ to the motors for the next time step Δt.

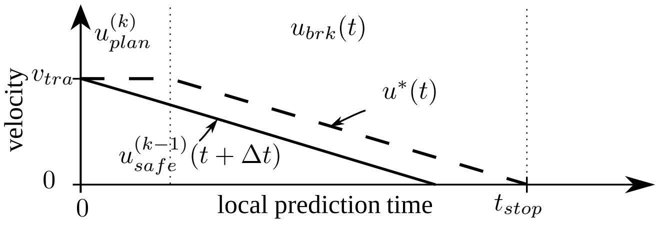

Fig. 4: Continued use of previously verified safe trajectory $u_{safe}^{(k-1)}(t)$, shifted by Δt, if $u^*(t)$ is not safe

If $u^*(t)$ is not successfully verified, we simply execute the previously verified safe input trajectory $u_{safe}^{(k)}(t) := u_{safe}^{(k-1)}(t + \Delta t)$ (see Fig. 4), which is shifted by Δt. Thus, we ensure that only inputs verified as safe are applied to the motors. The robot does not necessarily approach zero velocity if during the braking maneuver a new motion command is verified as safe and applied to the motors. This approach is implemented as a safety control module (Alg. 1) between the planning module and the robot motor control (see Fig. 5). We initially set $u_{safe}^{(k-1)}(t) := [0, 0]^T$.

Depending on the hardware, robot control often encounters a system delay. We can easily account for that by extending $u^*(t)$ with all the delayed motion commands in $[-t_{delay}; 0]$.

IV. Evaluation

In the following, we present the results of evaluating our pedestrian model (Sect. IV-A), and we present results on the performance of our online verification approach (Sect. IV-B).

A. Model Evaluation

We check whether the pedestrian model of Sect. III-A overapproximates the real behavior of a walking-only human by performing a reachset conformance test (Def. 3) using ground truth pedestrian trajectories from a labeled video source of a street scene in Zurich, Switzerland [14]. We check if the trajectories lie inside the computed pedestrian reachable sets. This test was performed for a time horizon

Algorithm 1 Online safety control (step k, sampl. time Δt)

```
Input: $u_{safe}^{(k-1)}(t), u_{plan}^{(k)}, \mathscr{X}_{rob}^{0}, \mathscr{X}_{ped,1..n}^{0}$ for $n$ pedestrians
Output: $u_{safe}^{(k)}(t)$
1: Set $u^*(t) := \begin{cases} u_{plan}^{(k)} & 0 \leq t < \Delta t \\ u_{brk}(t) & \Delta t \leq t \leq t_{stop} \\ 0 & t_{stop} < t \end{cases}$ (see Fig. 3)
2: Compute $\mathscr{R}_{rob}(t)$ with $\mathscr{X}_{rob}^{0}$, $u^*(t)$ for $t \in [0;t_{stop}]$
3: Compute $\mathscr{R}_{ped,i}(t)$ with $\mathscr{X}_{ped,i}^{0}$ for $t \in [0;t_{stop}]$ for all $i = 1..n$ pedestrians
4: if $\forall i : \mathscr{R}_{ped,i}(t) \cap \mathscr{R}_{rob}(t) = \emptyset$ for all $t \in [0;t_{stop}]$ then
5:     Set $u_{safe}^{(k)}(t) := u^*(t)$ for all $t \geq 0$
6: else
7:     Set $u_{safe}^{(k)}(t) := u_{safe}^{(k-1)}(t + \Delta t)$ for all $t \geq 0$
8: end if
```

TABLE I: Pedestrian model and conformance test results

Pedestrian Model		**Conformance Test**	
Time horizon	1.6 s	Pedestrians	420
a_{max}	0.6 m/s^2	Gener. test cases	20084
v_{max}	2.0 m/s	Passed tests	19843
Ped. diameter	0.54 m	Rate	98.80 %
$\mathscr{X}_{ped}^{0}$: (p_x, p_y)-uncertain.	± 0.1 m		
$\mathscr{X}_{ped}^{0}$: (v_x, v_y)-uncertain.	± 0.1 m/s		

of 1.6 s, which is larger than the largest t_{stop} of the robot in our evaluation (Sect. IV-B).

For the pedestrian model, we parameterize $v_{max} = 2.0$ m/s as suggested by [4], because it is the transition speed between walking and running. To set a_{max}, we apply numerical differentiation and filtering on the velocity data of the video source and then set $a_{max} = 0.6$ m/s^2 as an overapproximative value. The parameters of our model are shown in Tab. I.

The results (Tab. I) show good reachset conformance results. However, there are some unsuccessful tests. A closer look at these failed cases reveals that the unsuccessful tests are caused by special pedestrian behavior lying outside of our initial assumptions such as changing directions too fast (12 cases), and velocities faster than v_{max} (229 cases).

This conformance test shows that our pedestrian model is reachset conformant to walking-only pedestrians which do not change their direction of movement very abrupt. This pedestrian model can therefore be used for our verification approach if we are able to constrain human behavior to walking-only and slow-direction-changing, which is possible in a closed environment setting, as in production plants. However, our model is not reachset conformant to all pedestrian behaviors and there are two possible solutions. First, we could increase the bounds a_{max} and v_{max} leading to a richer set of behaviors and thus, to bigger reachable sets. Second, one may consider hybrid models switching to more conservative models, as suggested in [13], once the special cases above are detected. In addition, runtime monitors as proposed by ModelPlex [15] could be used to continuously validate the correctness of the used model at runtime.

B. System Evaluation

We evaluate the performance of our online verification in a ROS simulation for different scenarios where the robot has to navigate in the presence of pedestrians.

Considered Approaches: We compare three approaches with different obstacle models. First, we use our approach introduced in the previous sections. Second, we consider an ISO13482-compliant safety field [5] with 360° warning and protective fields. The size of the safety field is fixed and dimensioned based on the maximum speed of the robot and the assumption that a pedestrian may approach the robot at full speed at any point in time. In contrast, the size of the reachable sets in our approach is dynamic and depends on the current velocity of the robot and pedestrians. The third approach is based on the obstacle model used in braking ICS [9] and by Mitsch et al. [11]. This obstacle model assumes that obstacles may always move at full speed in any direction if we do not know their future behavior and requires that the robot is able to come to a rest before the obstacle may hit it. We refer to this approach as braking ICS in the following. In contrast, our approach computes reachable sets based on current velocity and direction of movement.

Experiment Setup: We execute our evaluation based on ROS Indigo. The physics simulation is carried out in Gazebo 7[2] and the robot uses a standard move base based on the Dynamic Window Approach (DWA, [16]). We use the default parameters from the Indigo release for the move base, except that we set the maximum velocity and acceleration for the differential drive robot to $v_{tra} = 1.5$ m/s, $v_{rot} = 2.0$ rad/s, $a_{tra} = 1.5$ m/s^2, and $a_{rot} = 1.0$ rad/s^2. Initially, the robot is stationary. The robot model is based on the Robotino models for Gazebo by RWTH Aachen[3], where we use its laser scanner for navigation and use the standard Planar Move Plugin to steer the robot.

The setup of our ROS system is shown in Fig. 5. The pedestrian simulation (upper left box) computes the pedestrian positions and velocities that are then sent to Gazebo (upper right box) and to the online safety control. The robot simulation in Gazebo (left box) simulates the robot actuators and provides laser scan data for localization in our map. The localization is performed in amcl, and information on robot pose and static obstacles contained in the laser scans are used by the ROS move base for computing a path to the goal position (lower right boxes). In addition, the move base contains the DWA implementation that generates the velocity commands for the robot. These velocity commands are then fed into the online safety control node (middle right box) and only forwarded to the robot actuators in the Gazebo Robot Simulation if they are safe. In our experiments, the whole setup of the ROS system remains unchanged except that we change the safety checker inside the online safety control.

For improving the efficiency of our approach and to enable real-time performance, we created a library of reachable sets for the pedestrians at design time that we store in a look-

[2]http://gazebosim.org/
[3]https://git.fawkesrobotics.org/gazebo-models.git

up table (14 MB). Since the pedestrian model equations are independent from the initial position, we only need to sample based on the initial velocity, which we do in steps of 0.1 m/s.

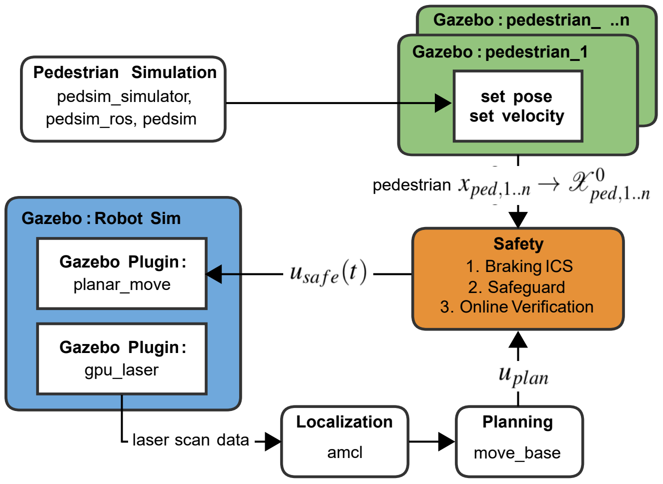

Fig. 5: Setup of the simulation environment in ROS.

The evaluations are carried out on a map that is illustrated in Fig. 6. The map is 24 m times 30 m from wall to wall and pedestrians walk continuously counter-clockwise along the green area. For obtaining realistic pedestrian motion, we simulate pedestrian motion in a dedicated Pedestrian Simulator[4] (PedSim) that is based on social forces. The simulated pedestrian positions are then transferred to Gazebo, while the robot position is also considered in PedSim such that the pedestrians react to the robot.

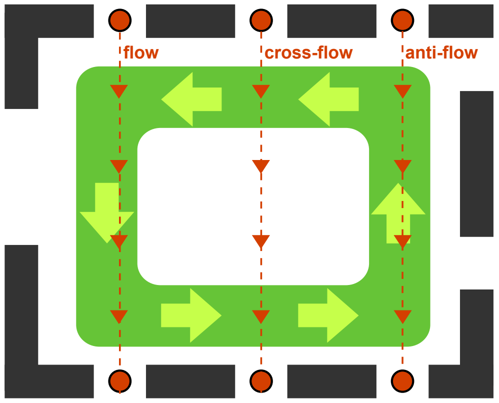

Fig. 6: Illustration of the map used for the experiments.

In our experiments, the robot will always move from top to bottom through the green area with different starting positions. Depending on the starting position, we create three scenarios for encountering pedestrians: flow (move in same direction as pedestrians), cross-flow (pedestrians coming from left or right), and anti-flow (pedestrians approaching from front). In addition, we consider two pedestrian densities: light population and dense population. For light population, we place 25 pedestrians uniformly at random

[4]https://github.com/srl-freiburg/pedsim_ros

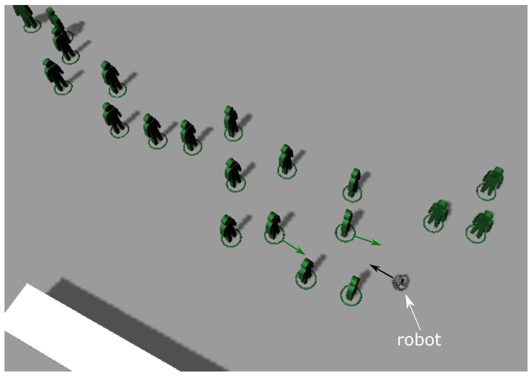

Fig. 7: Gazebo screenshot for a dense antiflow scenario.

in the green area; for dense population, we distribute 60 pedestrians. We create 10 different placements of pedestrians for each density with a minimum distance of 0.5 m between any two pedestrians. Fig. 7 shows a Gazebo screenshot of a typical situation in anti-flow scenarios with dense population.

Assumptions: The current pose and velocity of each pedestrian, which we require as inputs to our online verification, would need to be provided by a people-tracking approach on a real robot. In simulation, we instead take this information directly from PedSim. This therefore represents the best case, where we can track all pedestrians perfectly and exactly know their current position and velocity. In order to account for the imprecision of current perception and tracking approaches, we add an uncertainty of 0.1 m to the pedestrian positions and 0.1 m/s to the pedestrian velocities to make our simulation more realistic. Finally, we have also included the actual control delay of 100 ms that the real hardware exhibits.

Experimental Execution: For each scenario (flow, cross-flow, anti-flow), and for both light and dense populations, we generate 10 different pedestrian placements. All three approaches are executed on all of the situations. Based on the collected data, we compute (1) whether the goal has been reached, (2) how long it took to reach the goal, (3) the distance traveled by the robot, (4) the average velocity, and (5) the number of unsafe collisions. An unsafe collision is one in which the robot's velocity is greater than 0.

Results: The results are summarized in Table II for lightly populated situations and in Table III for the densely populated ones. Throughout our simulation runs, no unsafe collisions occurred for any of the approaches, so we omitted the corresponding column in the result tables. All values are arithmetic means over all runs.

The results clearly show that our method performs best in all cases by a large margin. Even in the simplest situation, motion with a lightly populated flow, our method is 1.4 times faster, and in the dense situation this even increases to a factor of 3.5. The example in Fig. 8 illustrates how the robot is able to follow a group of pedestrians in a flow scenario with light population when applying our online verification approach. It is also notable that both safety field and Braking ICS exhibit very bad performance in the anti-flow situation. This is despite the fact that our pedestrian

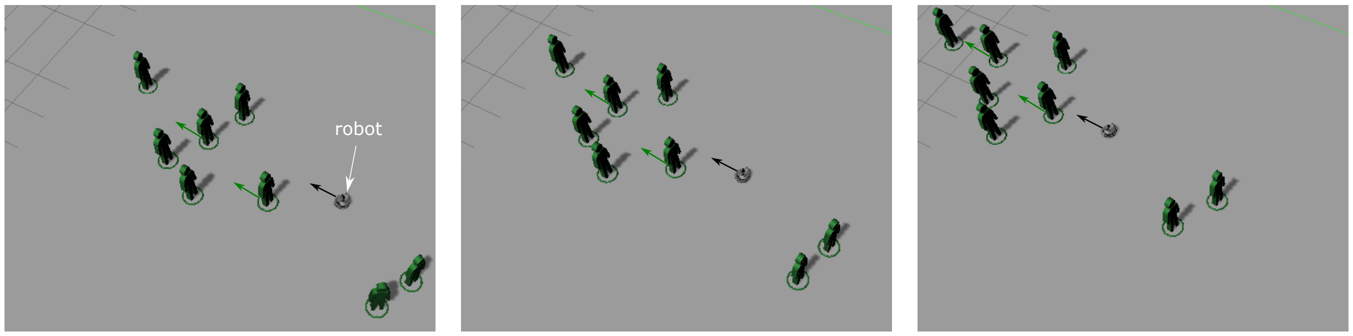

Fig. 8: Gazebo screenshots from the same position with a time step of approx. 1 s for a light flow scenario where the robot uses the online verification approach.

TABLE II: Results from ROS Simulation (Lightly Populated Scenarios)

	Flow				**Cross-flow**				**Anti-flow**			
Approach	@Goal	Time(s)	Len(m)	Vel(m/s)	@Goal	Time(s)	Len(m)	Vel(m/s)	@Goal	Time(s)	Len(m)	Vel(m/s)
Braking ICS	10	34.3	22.9	0.73	10	40.0	23	0.63	10	116.1	23.0	0.21
Safety Field	10	37.9	22.9	0.64	10	35.7	23	0.68	10	74.8	22.9	0.32
Onl. Verif.	10	22.4	22.9	1.04	10	26.3	23.2	0.91	10	52.3	23.0	0.45

TABLE III: Results from ROS Simulation (Densely Populated Scenarios)

	Flow				**Cross-flow**				**Anti-flow**			
Approach	@Goal	Time(s)	Len(m)	Vel(m/s)	@Goal	Time(s)	Len(m)	Vel(m/s)	@Goal	Time(s)	Len(m)	Vel(m/s)
Braking ICS	10	108.0	22.9	0.25	10	114.2	23.1	0.21	10	519.5	23.2	0.05
Safety Field	10	96.0	22.9	0.27	10	76.9	23	0.31	10	251.5	23.0	0.10
Onl. Verif.	10	26.0	23	0.92	10	37.8	23.1	0.65	10	159.2	23.2	0.15

simulator is cooperative, i.e. humans attempt to actively avoid the robot, and that the robot also uses a normal obstacle avoidance algorithm (albeit one that makes a static obstacle assumption). To this end, the effect of the obstacle avoidance seems to be minimal because the path lengths traveled by the robot are nearly the same for the three approaches in all considered situations. That means the robot takes almost the same path in all situations and only adjusts its speed instead of going round the populated areas, which we attribute to the static obstacle assumption.

For dense population, our online verification method provides significant improvements in average velocity for the flow and cross-flow scenarios. For the anti-flow scenario, the online verification still enables an average velocity that is a factor 2 (safety field) or 3 (braking ICS) higher compared to the other approaches, but an absolute average velocity of 0.15 m/s still leaves significant room for improvement.

Last, but not least, it might be surprising that the Braking ICS approach often performs worse than the ISO 13482 safety field. We surmise that this is because the safety field includes a warning field, which just reduces speed but still enables to robot to move. In contrast, the Braking ICS approach always stops the robot when it detects a potential collision, which essentially corresponds to a having safety field without a warning field.

V. Related Work

We discuss related work that aims at establishing provably safe motion of mobile robots with respect to a mathematical model considering moving obstacles, particularly humans.

Full Obstacle Knowledge: The first approaches in this regard are inevitable collision states (ICS, [17]), non-linear velocity obstacles [18], and the FD* path planner [19]. All of these approaches, however, make the assumption of exactly knowing the future behavior of all obstacles during the planning horizon, which is unrealistic for pedestrians.

Conservative Obstacle Model: Braking ICS [9] use a conservative obstacle model where an obstacle may always move with maximum speed in any direction. This obstacle model corresponds to the relevant safety norms [5], [4]. Similar to our approach, braking ICS append to each verified trajectory a braking trajectory for proving passive safety [6]. In contrast to our approach, they compute and check several possible braking trajectories for the robot. Mitsch et al. [11] and Zhang et al. [20] use theorem proving for showing that the DWA enables passively safe motion for differential drive robots using the same obstacle model as braking ICS. Likewise, Dabadie et al. [21] assume that obstacles behave in the worst possible way while proving collision-free motion based on a reach-avoid problem formulation. Aniculaesei et al. [22] construct an observer monitor that considers a fixed braking distance and obstacle velocity for constructing a safety circle around the robot. Similar to a safeguard, the monitor triggers a safe braking maneuver if an obstacle enters the safety circle. As indicated by our evaluation results, such models with a fixed maximum velocity are conservative and lead to decreased performance in populated environments.

Motion Primitives: The approaches by Hess et al. [23] and Majumdar et al. [24] build a library of motion primitives from which they construct motion plans. These motion plans include occupancies of the robot along the trajectory and may be verified against occupancies of obstacles. Such approaches could be a useful extension to our approach if the online computation of robot reachsets is too resource demanding.

Occluded Obstacles: The approaches by Alami et al. [25] and Chung et al. [26] consider (partially) occluded obstacles, e.g., if humans appear from a crossing corridor. Obstacles are assumed to appear with maximum speed at any time. For these cases, our model cannot be applied and the use of such conservative models is necessary.

Probabilistic Approaches: Probabilistic approaches like probabilistic ICS [27] and probabilistic collision states [28], [29] accept a small probability of collision, which is not acceptable in settings with heavier robots such as intralogistics.

VI. Conclusions

We present a safety approach for mobile robots that guarantees passive safety regarding walking pedestrians by an online verification using reachability analysis. Based on models of pedestrians and the mobile robot, we compute their reachable future occupancy at every timestep to determine whether a braking trajectory leads to a safe stop. In our evaluation, we demonstrate the validity of the pedestrian model using reachset conformance testing and discuss possible improvements. The evaluation of the online verification in a ROS simulation shows that our approach enables significantly improved navigation performance through crowds compared to standard safety approaches.

Future work focuses on real world applicability of our approach. Today's people-tracking algorithms still lack accuracy and reliability especially in velocity estimation, which needs to be the major focus. The effect of sensor occlusion on our approach is also of interest. Furthermore, additional modelling effort is needed, e.g., to extend our pedestrian model or to consider other dynamical objects in the environment.

Acknowledgment

The authors gratefully acknowledge financial support by the European Commission project UnCoVerCPS under grant number 643921.

References

[1] T. Kruse, A. K. Pandey, R. Alami, and A. Kirsch, "Human-aware robot navigation: A survey," *Robotics and Autonomous Systems*, vol. 61, no. 12, pp. 1726 – 1743, 2013.

[2] J. Minguez, F. Lamiraux, and J.-P. Laumond, "Motion planning and obstacle avoidance," in *Springer Handbook of Robotics*, B. Siciliano and O. Khatib, Eds. Springer, 2016, pp. 1177–1202.

[3] *Industrial trucks - Safety requirements and verification - Part 4: Driverless industrial trucks and their systems (ISO/DIS 3691-4:2006)*, ISO Std., 2011.

[4] *Safety of machinery – Positioning of safeguards with respect to the approach speeds of parts of the human body (ISO 13855:2010)*, ISO Std., 2010.

[5] *Robots and robotic devices – Safety requirements for personal care robots (ISO 13482:2014)*, ISO Std., 2014.

[6] K. Maček, D. Vasquez, T. Fraichard, and R. Siegwart, "Towards safe vehicle navigation in dynamic urban scenarios," *Automatika*, vol. 50, no. 3-4, pp. 184–194, 2009.

[7] M. Althoff and J. M. Dolan, "Online verification of automated road vehicles using reachability analysis," *IEEE Trans. Robot.*, vol. 30, no. 4, pp. 903–918, 2014.

[8] A. Pereira and M. Althoff, "Safety control of robots under computed torque control using reachable sets," in *Proc. of ICRA 2015*, pp. 331–338.

[9] S. Bouraine, T. Fraichard, and H. Salhi, "Relaxing the inevitable collision state concept to address provably safe mobile robot navigation with limited field-of-views in unknown dynamic environments," in *Proc. of IROS 2011*, pp. 2985–2991.

[10] F. Immler, "Verified reachability analysis of continuous systems," in *TACAS 2015*, ser. Lecture Notes in Computer Science. Springer, 2015, vol. 9035, pp. 37–51.

[11] S. Mitsch, K. Ghorbal, and A. Platzer, "On provably safe obstacle avoidance for autonomous robotic ground vehicles," in *Proc. of Robotics: Science and Systems*, 2013.

[12] H. Roehm, J. Oehlerking, M. Woehrle, and M. Althoff, "Reachset conformance testing of hybrid automata," in *Proc. of the HSCC*, 2016, pp. 277–286.

[13] M. Althoff, D. Hess, and F. Gambert, "Road occupancy prediction of traffic participants," in *16th International IEEE Conf. on Intelligent Transportation Systems*, 2013, pp. 99–105.

[14] S. Pellegrini, A. Ess, K. Schindler, and L. van Gool, "You'll never walk alone: modeling social behavior for multi-target tracking," in *Proc. of ICCV*, 2009, pp. 261–268.

[15] S. Mitsch and A. Platzer, "Modelplex: verified runtime validation of verified cyber-physical system models," *Form. Methods Syst. Des.*, vol. 49, no. 1, pp. 33–74, Oct. 2016.

[16] D. Fox, W. Burgard, and S. Thrun, "The dynamic window approach to collision avoidance," *IEEE Robot. Autom. Mag.*, vol. 4, no. 1, pp. 23–33, 1997.

[17] L. Martinez-Gomez and T. Fraichard, "Collision avoidance in dynamic environments: An ics-based solution and its comparative evaluation," in *Proc. of ICRA 2009*, pp. 100–105.

[18] F. Large, C. Laugier, and Z. Shiller, "Navigation among moving obstacles using the nlvo: Principles and applications to intelligent vehicles," *Auton Robots*, vol. 19, no. 2, pp. 159–171, 2005.

[19] M. Seder and I. Petrović, "Dynamic window based approach to mobile robot motion control in the presence of moving obstacles," in *Proc. of ICRA 2007*, pp. 1986–1991.

[20] M. Zhang and X. Zhang, "Formally verifying navigation safety for ground robots," in *2016 IEEE Intern. Conf. on Mechatronics and Automation*, 2016, pp. 1000–1005.

[21] C. Dabadie, S. Kaynama, and C. J. Tomlin, "A practical reachability-based collision avoidance algorithm for sampled-data systems: Application to ground robots," in *Proc. of IROS 2014*, pp. 4161–4168.

[22] A. Aniculaesei, D. Arnsberger, F. Howar, and A. Rausch, "Towards the verification of safety-critical autonomous systems in unknown environments," in *Proc. of the 1st International Workshop on Verification and Validation of Cyber-Physical Systems*, 2016.

[23] D. Hess, M. Althoff, and T. Sattel, "Formal verification of maneuver automata for parameterized motion primitives," in *Proc. of IROS 2014*, 2014, pp. 1474–1481.

[24] A. Majumdar and R. Tedrake, "Robust online motion planning with regions of finite time invariance," in *Algorithmic Foundations of Robotics X*, ser. Springer Tracts in Advanced Robotics. Springer, 2013, vol. 86, pp. 543–558.

[25] R. Alami, K. M. Krishna, and T. Siméon, "Provably safe motions strategies for mobile robots in dynamic domains," in *Autonomous Navigation in Dynamic Environments*, ser. Springer Tracts in Advanced Robotics. Springer, 2007, vol. 35, pp. 85–106.

[26] W. Chung, S. Kim, M. Choi, J. Choi, H. Kim, C.-b. Moon, and J.-B. Song, "Safe navigation of a mobile robot considering visibility of environment," *IEEE Trans. Ind. Electron.*, vol. 56, no. 10, pp. 3941–3950, 2009.

[27] A. Bautin, L. Martinez-Gomez, and T. Fraichard, "Inevitable collision states: A probabilistic perspective," in *Proc. of ICRA 2010*, pp. 4022–4027.

[28] D. Althoff, M. Althoff, D. Wollherr, and M. Buss, "Probabilistic collision state checker for crowded environments," in *Proc. of ICRA 2010*, pp. 1492–1498.

[29] D. Althoff, J. J. Kuffner, D. Wollherr, and M. Buss, "Safety assessment of robot trajectories for navigation in uncertain and dynamic environments," *Auton Robots*, vol. 32, no. 3, pp. 285–302, 2012.