

A Probabilistic Methodology for Predicting Injuries to Human Operators in Automated Production lines

Ruslan Asaula, Daniele Fontanelli, Luigi Palopoli

Abstract—Mobile robots are increasingly utilised in automated plants to the purpose of moving wares and material between the different production lines and logistic areas. In this context, the presence of human operators in the facility is frequently allowed to carry out or supervise some phases of the production. The problem arises of how to make the co-existence possible with controlled risks for the operator and without affecting the productivity with frequent interruptions. In this paper we propose a solution to this problem based on a probabilistic technique. A system of visual sensor (mounted on the mobile robots) detects the presence of a human operator and a discrete abstraction (essentially a discrete-time Markov chain) is used to predict his/her motion and hence find the probability of an accidental injury. For the computation of the latter, we combine the probability of having a collision with a given speed with the probability of receiving an injury out of the collision (taken from physiological models suggested by the automotive literature).

I. INTRODUCTION

Robots represent an increasingly effective solution for industrial production processes due to their relative availability, autonomy and flexibility. A problem that still hinders their penetration into the wide industrial market is the need for coexistence of robotic systems and human workers in the same environment. The problem is evident when part of the production process is delegated to robotic cells consisting of articulated robot arms. In this case fencing off the robotic cell, as required by the most of the current regulations, makes the interaction between robots and human workers awkward and unproductive. The increasing use of autonomous mobile vehicles for the goods handling in warehouses and production plants makes the fencing solution inconceivable (it would be equivalent to preventing the movement of human operators in the factory altogether). Hence, the compelling need for active systems that monitor the area (e.g., by visual sensors) raising alarms only when a real situation of danger is detected. In the particular case of vehicle and human interaction, the “perception” of a danger should not simply related to the probability of a collision but, more precisely, to the probability of having a human injury as a result of the collision. In fact a collision at a very low speed of the vehicle on the factory floor (e.g., in the order of a very few meters per second) can be tolerated.

The research leading to these results has received funding from the European Community’s Seventh Framework Programme FP7 under grant agreement no IST-2008-224428 “CHAT - Control of Heterogeneous Automation Systems”

Authors are with the Department of Engineering and Information Science, University of Trento, Via Sommarive 14, 38050 Trento (TN), Italy. Phone: +390461883967. Fax: +390461882093. ruslan.asaula@gmail.com, fontanelli@disi.unitn.it, palopoli@dit.unitn.it

Classical approaches to the problem are based on the existing literature for the detection and avoidance of moving obstacles [12], [2], [5]. A first possibility is to avoid any possible collision by a static and conservative approach. The plant is divided into zones: as soon as the human enters in a certain zone, identified by some static sensor (e.g., photo-cells or surveillance cameras), all the robots in the zone are stopped till the human leaves it. In its practical effects this solution is hardly different from the installation of physical fences limiting the possible movement of humans inside the plant. In the second solution, more flexible than the first one, the vision system and/or a proximity sensor is placed on the robot, which detects the presence of a human and then stops the robot or re-plans the robot trajectory to avoid the obstacle. Both the solutions presented provide a high level of safety but they can also results in useless stops, that reflect in a high loss of energy/time.

This paper provides an effective way for predicting the probability of an accident/injury and then, incidentally, taking an appropriate course of actions (e.g., deciding to stop or to go ahead). We use a combination of two technologies: a vision system, which allows us to detect human targets in front of the robot, and a prediction engine, which produces in real-time the probability of a collision in a given time horizon. In this paper, we focus on the latter component. In order to produce the prediction, we start from a dynamic model describing the motion of a human being, inspired from [13], [14]. The model is used to produce a Discrete Time Markov Chain (DTMC) that serves as a discrete approximation of the system. The use of discrete approximations for dynamic systems has become quite popular in the last few years [11], [6], as an effective means to carry out control synthesis and verification of complex properties for both linear and nonlinear systems using the tools developed for discrete systems. The particular technique that we advocate lies in the track opened by [7], where the authors show how to construct a DTMC to predict the collision between two aircraft flying at the same altitude. Given the simplicity of our problem, we can use an algorithm for the probability computation developed ad-hoc. The use of more general tools such as probabilistic model checkers [9], [1] would enable one to use the same framework for predicting the occurrence of more complex events (e.g., trapping).

A. Problem Description and solution overview

Consider a mobile robot equipped with a fixed camera, moving on a plane, e.g., on a factory floor. The robot is assumed to move along a predefined trajectory. For the sake

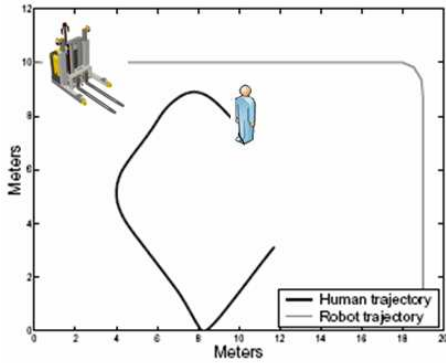


Fig. 1. Human and mobile robot trajectories in a fixed frame.

of simplicity, we do not consider deviations from the planned trajectory (e.g., due to possible controller errors). Due to its rather extensive use in factory automation, we model the vehicle kinematic assuming a unicycle-like robot. Since we are interested in a factory environment, we can assume that the robot is moving with constant forward velocity without loss of generality. Curved trajectories are obtained imposing a constant angular velocity, thus resulting in curves with well defined, constant, curvature radius.

The factory is populated by human operators whose position, velocity and acceleration are estimated in the mobile robot frame $\langle R \rangle$ using visual information. The possible trajectories of the human in the robot frame are predicted in a given time horizon by using an abstraction technique. Namely, the state space is partitioned using a grid and a DTMC is constructed, in which each state is associated with a position in the grid. The transition probabilities between the states of the DTMC are computed using a stochastic model for the motion of the human operator. Since the coordinates of the human are measured with respect to $\langle R \rangle$ frame (figure 1) it is possible to derive a set of positions and relative velocities in which collisions will most probably happen. Such positions and velocities on the plane of motion, i.e., points in \mathbb{R}^4 , are referred to as *bad states*. Notice that the bad states are defined in the moving frame $\langle R \rangle$, hence they are constant in time. It follows that the probability of collision is the probability that a human reaches this set of dangerous states. If such probability is above a predefined safety threshold, a possible collision is detected and, therefore, the robot is stopped.

The safety threshold is estimated using recent results in the automotive literature [3] that relate the probability for a pedestrian of getting injured in a collision with a car at a given relative speed. Figure 2-A shows cumulative speed distribution for any kind of injury, while figure 2-B shows the data only for non-minor injuries, as reported in [3].

The rest of the paper is organized as follows: in Section II the model for the human motion is presented, in Section III the discrete time stochastic abstractions for human and robot motions are presented, while description of the safety algorithm is presented in Section IV. Finally Section V

reports data from several simulations, that demonstrate the effectiveness of the proposed solution in comparison with more conservative solutions.

II. HUMAN MOTION MODEL

Human workers in factories usually move according to specific patterns determined by the working activity to be executed. In this paper, we make the conservative assumption that the possible movements are solely determined by the physical limitations inherent to the human body. As a result, the probability of having a collision is overestimated (being all possible trajectories a superset of the ones observable in a working environment).

Since the severity of the accident is related to the relative motion of the robot and of the human target, we need to reconstruct and predict the trajectory of the latter in a frame attached to the robot. In this section, we show how this task can be carried out. In particular, we consider two different sampling instants (k and $k + 1$) and evaluate the possible movements of the target in these time interval. This is done in three steps. In the first one, we translate the measurements of the target position and velocity from the robot frame $\langle R \rangle$ to a fixed frame $\langle F \rangle$. In the second one, we use a kinematic model for the human motion to predict the target position in $\langle F \rangle$. In the third one, we compute the new position in $\langle R \rangle$, taking into account the motion of the robot in the interval.

A. Converting the target position/velocity from $\langle R \rangle$ to $\langle F \rangle$

The target presence is detected by on-board sensors, i.e., a camera in our case, that measure the position ${}^R P_h(k)$, the velocity ${}^R V_h(k)$ and the acceleration ${}^R a_h(k)$ of the target in the robot frame $\langle R \rangle$. Notice that ${}^R V_h(k)$ and ${}^R a_h(k)$ measurements are affected by the forward $v(k)$ and angular $\omega(k)$ velocities of the robot, known and given by the vehicle motion controller. Since the target motion model is defined in a fixed frame $\langle F \rangle$, we need to recover the velocity ${}^F V_h(k)$ and the acceleration ${}^F a_h(k)$ of the target in $\langle F \rangle$. The aforementioned quantities are related by

$$\begin{cases} {}^R V_h(k) = {}^R V_{h,v}(k) + {}^R V_{h,\omega}(k) + {}^F V_h(k) \\ {}^R a_h(k) = {}^R a_{h,\omega}(k) + {}^F a_h(k) \end{cases}, \quad (1)$$

where ${}^R V_{h,v}(k)$ is the target velocity in $\langle R \rangle$ due to the forward robot velocity $v(k)$, ${}^R V_{h,\omega}(k)$ is the target velocity in $\langle R \rangle$ due to the angular robot velocity $\omega(k)$ and ${}^R a_{h,\omega}(k)$ represents the fictitious accelerations (Coriolis and centrifugal) in $\langle R \rangle$ due to the angular robot velocity $\omega(k)$, equals to ${}^R a_{h,\omega}(k) = -\omega(k)^2 {}^R P_h(k)$, where ${}^R P_h(k) = [x, y]^T$ is the position of the target in $\langle R \rangle$ at time step k .

Since $v(k)$ is supposed to be constant (as it is customary in a factory floor) and it is always directed along the X axis of $\langle R \rangle$, it follows that ${}^R V_{h,v}(k) = [v(k), 0]^T$. Since $\omega(k)$ is positive for counter-clockwise rotation on the plane of motion, defined indifferently by the X and Y axis of $\langle R \rangle$ or $\langle F \rangle$, its vectorial representation is given by $[0, 0, \omega(k)]^T$ in all the frames. Recalling that the 3D target position is given

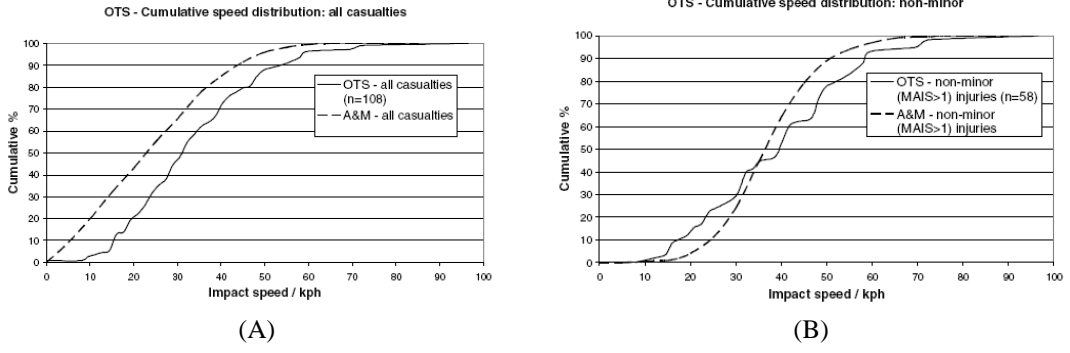


Fig. 2. Cumulative speed distribution for any kind of injury (A) and only for non-minor injuries (B). The dotted lines refers to statistical figures from real accidents collected in 1979 (OTP), while the continuous line (A&M) refer to statistics collected in 2007.

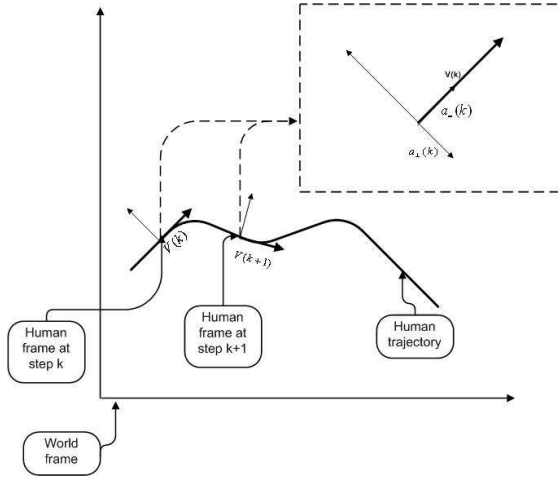


Fig. 3. Motion model adopted for human trajectory predictions.

by $[x, y, 0]^T$, we get

$$\begin{bmatrix} 0 & -\omega(k) & 0 \\ \omega(k) & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} x \\ y \\ 0 \end{bmatrix} \Rightarrow {}^R V_{h,\omega}(k) = \begin{bmatrix} -\omega(k)y \\ \omega(k)x \end{bmatrix}.$$

Using (1) is then possible to retrieve the velocity ${}^F V_h(k)$ and the acceleration ${}^F a_h(k)$ of the target w.r.t. $\langle F \rangle$ frame, in which the target motion model can be applied.

B. Human motion in $\langle F \rangle$

Let (x, y) represent the positions on the plane of the target and (v_x, v_y) the velocities w.r.t. to the X and Y axis respectively in $\langle F \rangle$. As customary in human motion model literature, the accelerations are instead referred to a reference frame, $\langle H \rangle$, attached to the human body, in which a_{\parallel} represent the tangential acceleration and a_{\perp} is the perpendicular acceleration. Figure 3 depicts the described quantities in terms of the fixed and moving reference frames. A widely adopted discrete-time model [13], [10] for the accelerations of a target that moves freely in the environment (without any specific restriction dictated by the execution of

a particular task) is given by

$$\begin{bmatrix} a_{\parallel}(k) \\ a_{\perp}(k) \end{bmatrix} = \begin{bmatrix} a_{\parallel}(k-1)e^{-\alpha\Delta t} + \sqrt{1-e^{-2\alpha\Delta t}}\sigma_{\parallel}u_{\parallel} \\ a_{\perp}(k-1)e^{-\alpha\Delta t} + \sqrt{1-e^{-2\alpha\Delta t}}\sigma_{\perp}u_{\perp} \end{bmatrix}, \quad (2)$$

where $a_{\parallel}(k)$ and $a_{\perp}(k)$ are random variables. Indeed, u_{\parallel} and u_{\perp} are two zero-mean white noise sequences with unit standard deviation, modelling the randomness of the target maneuvering motions. σ_{\parallel} and σ_{\perp} are the standard deviations of a modified uniformly distributed random variable modelling human physical limits related to accelerations ([14]). The presence of a dynamics in the evolution of the acceleration dictates an extension of the state space, which has to encompass the two acceleration terms and becomes 6-dimensional. Δt is the discretization time step (more on this in the next sections) and α is the reciprocal of the maneuver (acceleration) time constant (in our model has been estimated by visual data to be 1.5). In order to determine all the quantities of interest in the same reference frame, $\langle F \rangle$, we transform tangential and perpendicular accelerations into accelerations along the X and Y axis, by applying

$$\begin{bmatrix} a_x(k) \\ a_y(k) \end{bmatrix} = \begin{bmatrix} \frac{(v_x(k)a_{\parallel}(k) - v_y(k)a_{\perp}(k))}{\sqrt{v_x^2(k) + v_y^2(k)}} \\ \frac{(v_y(k)a_{\parallel}(k) + v_x(k)a_{\perp}(k))}{\sqrt{v_x^2(k) + v_y^2(k)}} \end{bmatrix}. \quad (3)$$

The velocity of a person is then modelled by the mean of the accelerations in the period Δt . More precisely

$$\begin{bmatrix} v_x(k+1) \\ v_y(k+1) \end{bmatrix} = \begin{bmatrix} \text{Sat}_x \left(v_x(k) + \frac{a_x(k+1) + a_x(k)}{2} \Delta t \right) \\ \text{Sat}_y \left(v_y(k) + \frac{a_y(k+1) + a_y(k)}{2} \Delta t \right) \end{bmatrix}, \quad (4)$$

where $\text{Sat}(\cdot)$ is a saturation function that takes into account the physical limits of the human motion, modelled as a maximum allowable velocity v_{max} . We have validated this model and identified realistic parameters by a long sequence of real data captured from a camera.

C. Prediction of the target position in $\langle R \rangle$

The trajectory of the robot is described by a set of possible positions, given by the time discretization of its continuous path with sample time Δt (more on this later). Therefore, consider the robot be in a certain position at time $k\Delta t$ in

which it detects a human. From (1), the velocity ${}^F V_h(k)$ and acceleration ${}^F a_h(k)$ are derived. Since (2), (3) and (4) do not rely on the human position in the fixed frame and since velocities and accelerations are free vectors, it is possible to select the fixed frame coincident with the robot frame at time $k\Delta t$, i.e., $\langle F \rangle \equiv \langle R \rangle_k$. This way, ${}^{R_k} V_h(k)$ and acceleration ${}^{R_k} a_h(k)$ are used in (2), (3) and (4) to derive ${}^{R_k} V_h(k+1)$ and acceleration ${}^{R_k} a_h(k+1)$. It is worthwhile to note that by inverting (3), it is possible to recover the tangential and perpendicular accelerations to be used in (2).

Using the measured human position ${}^{R_k} P_h(k)$ at step k is then possible to retrieve the predicted human position

$${}^{R_k} P_h(k+1) = {}^{R_k} P_h(k) + \frac{{}^{R_k} V_h(k+1) + {}^{R_k} V_h(k)}{2} \Delta t. \quad (5)$$

In order to recursively apply the same algorithm to predict position, velocity and acceleration for $k+2$, we need to transform the computed quantities from $\langle R \rangle_k$ to $\langle R \rangle_{k+1}$, which is known by integration of $v(k)$ and $\omega(k)$ over Δt . The position ${}^{R_{k+1}} P_h(k+1)$ is obtained using the transformation matrix between $\langle R \rangle_k$ and $\langle R \rangle_{k+1}$. Velocity ${}^{R_{k+1}} V_h(k+1)$ and acceleration ${}^{R_{k+1}} a_h(k+1)$ are instead obtained using only the rotation matrix between $\langle R \rangle_k$ and $\langle R \rangle_{k+1}$. At this point, since the new fixed frame coincides with $\langle R \rangle_{k+1}$ ($\langle F \rangle \equiv \langle R \rangle_{k+1}$), it is possible to iterate the algorithm here presented, until the desired prediction horizon is reached ($k+2, k+3, \dots, k+N$). The relative velocities between the robot and the human w.r.t. $\langle R \rangle_{k+1}$, that are needed to compute the probability of injuries (see the next Section), are obtained using (1).

III. DISCRETE STOCHASTIC APPROXIMATION

As shown in the previous section, under the reasonable assumption that the robot control task and the safety task are hosted on the same computing unit (or on communicating computing units), we can use a stochastic model to predict the trajectory of a human target in the robot frame. In this section, we show how this model can be used to compute the probability of an accidental collision. Our strategy is based on the construction of a discrete approximation of the system. In the terminology used in hybrid systems literature, a discrete approximation of a system is a discrete-state system generating behavior close enough to those of the original systems. Roughly speaking, by increasing the number of discrete states, the behaviors of the two systems should asymptotically coincide. Due to the stochasticity of our system, a suitable way for approximating its behaviors is by a DTMC. Likewise, a good notion for convergence of the approximation to the original system is in our case given by convergence in probability: increasing the number of states we should be able to decrease arbitrarily the difference between the probability of an event for the two systems (e.g., going in a bad state). This issue will be discussed at the end of the section. As shown in the rest of the paper, the introduction of a discrete approximation greatly simplifies the computation of the probability of relevant events (in our case, collision and trapping).

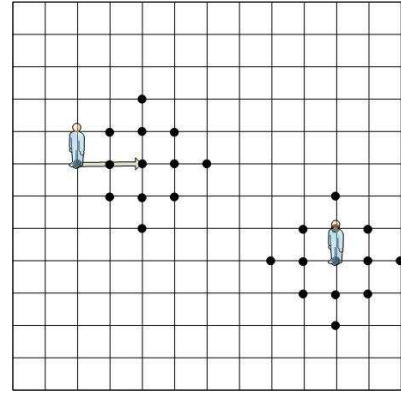


Fig. 4. Modelled human motion in discrete space.

The stochastic abstraction is constructed by gridding the six-dimensional state space of the human motion. The rationale behind this approach is to detect possible (discretized) future positions and velocities reached by the human worker using the model of accelerations given in (2). For example, Figure 4 depicts position predictions for two possible configuration of a human. More precisely, the person on the left has non zero velocities and accelerations in the initial configuration. Black dots represent positions reachable from the current states, function of the velocities and accelerations. It turns out that if a person is standing still (as the person on the right in Figure 4), the black dots configuration will be not biased by the velocities or accelerations (that are null) and all the possible reachable positions are symmetric w.r.t. the current position of the person.

Denoting with Δx and Δy the position distances and with Δv_x and Δv_y the velocity distances between grid intersections in X and Y directions respectively ($\Delta v_x = \Delta x / \Delta t$), the discrete state vector $S(k) = [X(k), Y(k), V_x(k), V_y(k), A_{\parallel}(k), A_{\perp}(k)]^T$ is described by the position coordinates $X(k)$ and $Y(k)$ for all the actual human positions $x \in (X(k) \pm \Delta x / 2)$ and $y \in (Y(k) \pm \Delta y / 2)$. Similarly, $V_x(k), V_y(k)$ represent the velocities on the grid if human velocities $v_x \in (V_x(k) \pm \Delta v_x / 2)$ and $v_y \in (V_y(k) \pm \Delta v_y / 2)$.

Knowing the accelerations in the discrete state space $A_{\parallel}(k), A_{\perp}(k)$, it is possible to derive future positions of the human using the models given in (2), (4) and (5). For example, it is possible to derive the probability density function f_p of the positions given the randomness introduced in the accelerations. Hence, by integration, we compute the probability of transition from state $S_p(k) = [X(k), Y(k)]$ to the state $S_p(k+1) = [X(k+1), Y(k+1)]$ using

$$P[S_p(k+1)|S_p(k)] = \int_{-\frac{\Delta x}{2}}^{\frac{\Delta x}{2}} \int_{-\frac{\Delta y}{2}}^{\frac{\Delta y}{2}} f_p(\tilde{S}_p(k+1)|S_p(k)) dx dy,$$

where $\tilde{S}_p(k+1) = [X(k+1) + x, Y(k+1) + y]$.

Alternatively, the probability density function $f_{a_{\parallel}}$ and $f_{a_{\perp}}$ of the accelerations may be used. Indeed, the new positions are a function of the parallel and orthogonal accelerations. Defining the function $Pos_x = Pos(a_{\parallel}(k+1), a_{\perp}(k+1), S(k))_x$

$(Pos_y = Pos(a_=(k+1), a_=(k+1), S(k))_y)$ that determines the possible reachable positions along the X (Y) axis given the accelerations and the current position and velocities, the transition probability $P[S(k+1)|S(k)]$ is derived computing

$$\int_{X(k+1)-\frac{\Delta x}{2} \leq Pos_x}^{X(k+1)+\frac{\Delta x}{2} \geq Pos_x} \int_{Y(k+1)-\frac{\Delta y}{2} \leq Pos_y}^{Y(k+1)+\frac{\Delta y}{2} \geq Pos_y} f_{a_=(a_)} f_{a_=(a_)} da_=(a_)=da_=(a_)=.$$

A. Grid Choice

The reliability of the proposed model very much depends on the convergence, in probability, of the discretized model to the continuous one. The typical way this problem is approached is by imposing ‘‘local consistency’’ as defined in [8], [4]. A good example of how to choose the grid parameters enforcing local consistency can be found in [7]. We validated our choice by comparing the probabilities obtained with our technique with the ones that we could find with a Montecarlo simulation. However, it is useful in this section, to explain the heuristic criteria that guided our choice by a simple example.

Consider an unidimensional motion of a particle and observe it for a period of time Δt . Suppose that the particle starts from an initial position $x_0 = 0$, with an initial velocity $v_0 = 0$ and robot is not moving, with an acceleration a supposed constant over the interval Δt . Taking inspiration from the model in Equation (2), we can assume that a is a stochastic variable uniformly distributed between $[-c, c]$ with $c = \sqrt{1 - e^{-\alpha \Delta t}} \sigma_=($. Hence, the variable $x(\Delta t)$ will be uniformly distributed. The variance of this variable can be easily computed as $\text{var}_x = \frac{c^2}{12} \Delta t^4$. Now, suppose you approximate the evolution of the variable by a grid: $\eta \Delta x$, where $\eta \in \mathbb{Z}$. It is easy to compute the distribution of the random variable η

$$P(\eta(\Delta t) = \bar{\eta} | \eta(0) = 0) = P(\eta \Delta x - \Delta x/2 \leq x \leq \eta \Delta x + \Delta x/2)$$

and then find its variance, which is given by:

$$\text{var}(\eta) = \frac{(\frac{c \Delta t^2}{\Delta x} + 1)(\frac{c \Delta t^2}{\Delta x} + 3)}{12}.$$

Consequently we can compute the approximation error in the variances as

$$\varepsilon = \frac{\text{var}(x) - \Delta x^2 \text{var}(\eta)}{\text{var}(x)} = \frac{3 \Delta x^2 + 4 c \Delta t^2 \Delta x}{c^2 \Delta t^4}.$$

As one would expect, the error decreases with the grid size.

IV. ALGORITHM

The DTMC abstracts the human motion model using finite set of states. Each state is six-dimensional, since it describes the position, the velocity and the acceleration of a human moving on a plane. The DTMC is used in this paper to determine the probability of human injuries in a given timing horizon. A possible way to do this is by using a model checker for stochastic system (such as PRISM), identifying by a logic formula the states to be marked as *bad*. This possibility makes our tool extremely flexible allowing us to specify a wide range of error conditions (e.g., related to a

concatenation of possible actions). However, we restrict our attention to a quite simple verification task, for which an ad-hoc solver has been developed.

Before going into the solution details, we have to define what is our concept of *collision*, upon which the logic expression pinpointing the bad states can be determined.

- **Impact:** if the human position or its predicted trajectory are, at some point in time, in a space occupied by the robot, then a collision is straightforwardly detected.
- **Trapping:** as it was mentioned earlier, the flexibility of the safety system here proposed allows to detect also trapping conditions, i.e., when a person is between the robot and some object, like a wall. The error expression in this case will evaluate to true when a human operator is in between the robot and an external object, and the distance between the robot and the external object is lower than the size of the human body (we assume that the human body is approximated as a cylinder by the vision system).

A. Identifying the bad states

In order to estimate the probability of collision, the discrete states in which such an events occurs must be identified. Recall that target position, velocity and acceleration are expressed in the moving frame $\langle R \rangle$, which implies that the collision states are constant in time. However, since we are using relative velocities, that can generate a large displacement if integrated for Δt , there can be the possibility to jump over the collision states in one time step. There are two possibility to avoid this undesirable side effect. The first solution reduces the time step Δt , paying the price of a reduced prediction horizon (due to the limited computation power) and/or a dramatic increase of the number of the reachable states (see Section III-A). The second solution, selected in this paper, modifies the set of bad states as a function of the relative velocity between the human and the robot. More precisely, the set of bad states is enlarged with all the possible states for which the human position can be reached from the current position by a jump-through the robot. In Figure 5 we show how to compute this set of bade states for three different relative velocities.

To detect trapping we can use either information about external objects from the vision system or a map of the factory provided in advance. In both cases, knowing the current position of the mobile robot w.r.t. to the object, the set of bad states is further enlarged with trapping-related bad states. Trivially, since the position of the robot w.r.t. the object is time dependent, it follows that the set of bad states will be time dependent too.

B. Computing the probability of collision

Since our goal is to compute the probability of having an accident, we augment the states of the DTMC describing the motion of the human with an additional state, called *bad* state, to which the DTMC is forced to switch whenever a collision is detected. In order to keep track of such a collision, the bad state is absorbing (i.e., it has a self

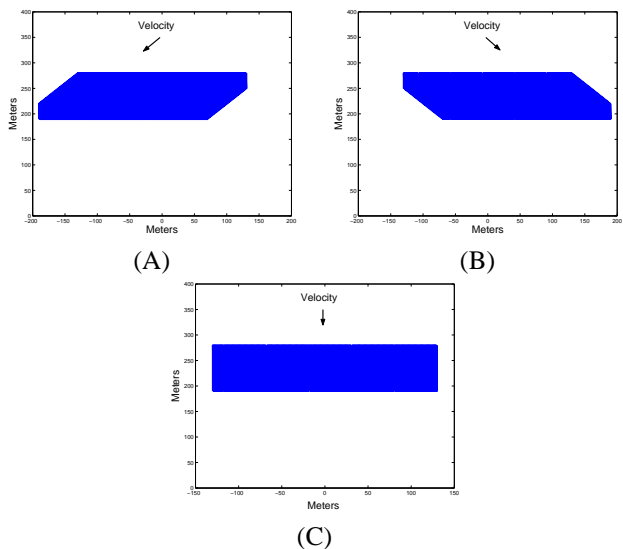


Fig. 5. Bad states for different relative velocities.

transition marked with probability 1). Therefore, the bad state collects the probability of collision accumulate in the considered time horizon. To increase the performance of the proposed safety system, we also define a similar absorbing state, a *safe* state, that collects the states with zero probability of collision during the prediction horizon. This way, the prediction of these states is no more propagated.

At each time instant, we associate a vector $\pi(k) = [P(S_1(k)), \dots, P(S_i(k)), \dots, P_{bad}(k), P_{safe}(k)]^T$ to the states of the resulting DTMC, whose dimension is given by the all the possible positions, velocities and accelerations in the grid reachable in the time horizon of interest. More in depth, $P(S_i(k))$ is the probability to be in the state $S_i(k)$ at time instant k , $i = 0, \dots, N$ where N is the number of possible target reachable states. Notice that the position of the robot is propagated within the time interval used for prediction, i.e., an integer multiple of the sampling time Δt , say g . If the robot motion does not intersect the region spanned by all the possible reachable state from $S_i(k)$ within the time horizon, region that is estimated once and for all off-line, then $P(S_i(k)) = 0, \forall i$, $P_{bad}(k) = 0$ and $P_{safe}(k) = 1$, for all time instants between k and $k + g$. Conversely, if $S_i(k)$ is already on the position occupied by the robot at time $k\Delta t$ (when the verification starts), then $P(S_i(k)) = 0, \forall i$, $P_{bad}(k) = 1$ and $P_{safe}(k) = 0$, for all time instants between $k\Delta t$ and $(k + g)\Delta t$.

In all the other cases, $P_{bad}(k) = P_{safe}(k) = 0$, $P(S_i(k)) = 1$, where $S_i(k)$ is the current position of the human, and the probability $\pi(k)$ have to be propagated for g steps¹. The step by step prediction is given using the transition probability matrix Π of the derived DTMC, whose entries are computed as shown in Section III. Notice that Π has the dimension determined by the grid granularity in position,

¹We could easily capture the case in which the position of the target is not exactly known by associating non zero probability to more than one state

velocity and acceleration. For the human motion prediction, only the portion of interest, determined by the dimension of $\pi(k)$, is used. Moreover, the transition probability matrix is “hemmed” with two additional columns and rows of zeros, plus two additional 1 on the diagonal, obtaining

$$\bar{\Pi}(k) = \begin{bmatrix} \Pi & 0 \\ 0 & I_2 \end{bmatrix},$$

where I_2 is a two dimensional identity matrix. This additional states correspond to the absorbing bad and safe states. $\bar{\Pi}(k)$ is not time invariant, since it depends on the trajectory of the robot motion (moving straight, turning left, turning right).

Therefore, $\pi(k+1)$ is iteratively computed by the following steps:

- 1) $\pi(k+1)^T = \pi(k)^T \bar{\Pi}(k+1)$;
- 2) For each $S_i(k+1)$, with $P(S_i(k+1)) \neq 0$, if $S_i(j+1)$ is a bad one then $P_{bad}(k+1) = P_{bad}(k) + P(S_i(k+1))$, while $P(S_i(k+1))$ is reset to 0. If the predicate is false, no operations are performed.
- 3) For each $S_i(k+1)$, with $P(S_i(k+1)) \neq 0$, the reachable future states are computed (by means of the off-line estimate of the reachable region). If all the future states are outside the region spanned by the robot in the prediction horizon of the robot, then no collision is possible. In this case, $P_{safe}(k+1) = P_{safe}(k) + P(S_i(k+1))$, while $P(S_i(k+1))$ is forced to 0. Otherwise, no operation is performed.

The algorithm above is applicable to the general case in which the bad states can change in time (e.g., to consider trapping with fixed obstacles). If the set of bad state is always the same (mere collision) we can make remarkable optimizations to the algorithm.

V. SIMULATION RESULTS

Due to the simplicity of the error condition, the algorithm presented in Section IV is particularly easy to implement. Therefore, we implemented an ad-hoc software tool using the standard C++ language, with additional libraries for sparse matrices handling. Moreover, the software development has been optimized exploiting the information on the known finite time horizon of the prediction. This way, it is possible to save memory and computational power with respect to standard tools such as PRISM. To validate the tool, we run several simulations on a toy system example on both our tool and PRISM, obtaining the same results.

Two simulation examples are here reported, stemming from a realistic scenario of a paper factory. For all the simulations, we chose a sampling time $\Delta t = 0.3$ s. The mobile robot is assumed to behave like a unicycle used for goods handling in a warehouse. The vehicle length is 5 m and the width is 2 m. In the simulations, the robot moves with constant linear velocity $v = 2$ m/s. Curved trajectories are obtained with constant angular velocity ω (constant radius curves), different for each example proposed. The stop time of the robot, due to its inertia, is assumed to realistically be of 1 s.

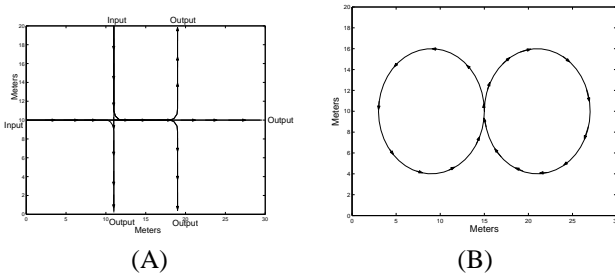


Fig. 6. A) Mobile robot trajectories in the factory floor. “Inputs” and “outputs” represent all the possible points in which the robot enters or leave the area. B) Mobile robot trajectory for the circular path.

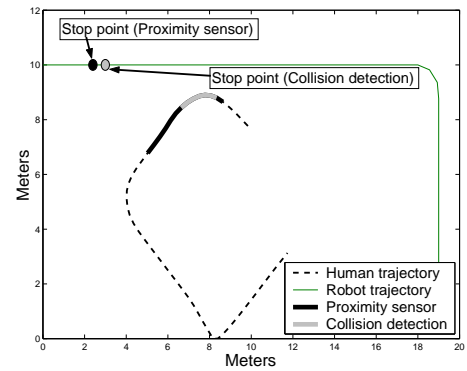
The proposed algorithm is compared with a conservative approach, based on proximity sensors. In this case, whenever the target is close to the robot less than a certain threshold, the robot stops. The threshold is determined for the worst case, i.e., assuming the human moving towards the robot with maximum velocity.

A. Realistic Factory Example

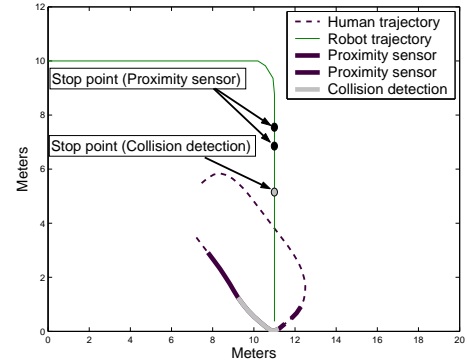
In the first simulation, we show a typical industrial case inspired to a paper production factory. In this case, all the possible trajectories are depicted in figure 6-A, and consist of straight line paths and curved trajectories of 1 m radius (i.e., with angular velocity $\omega = 2$ m/s). The target randomly moves inside the area of 30×20 meters, while the robot enters in the area from the “input” points and leaves the area from the “output” points (see figure 6-A). In the simulations, the robot is always inside the area, i.e., as soon as it leaves from an output, it enters again from an input, chosen randomly.

For a clear visualization of how our algorithm performs with respect to the conservative algorithm, we depict in figure 7 three different trajectories of the target (dashed, followed in counter clockwise direction) and three trajectories of the robot (solid, straight line, curve, straight line), followed from left to right. Superimposed to the dashed trajectory, the black bold trajectory shows the segment of trajectories of the human target in which the robot is stopped using the conservative approach with proximity sensors. The black circle on the trajectory shows the point in which, in this case, the robot stops. In this case the robot is restarted when the target is outside of the proximity sensors. The gray bold trajectory shows the target positions in which the robot stops using the proposed prediction algorithm (see figure 7). The robot is restarted when the probability of injuring the target decreases below the chosen threshold. Notice that the duration of the halting period is in the second cases much smaller. Figure 7-C clearly emphasizes this behavior.

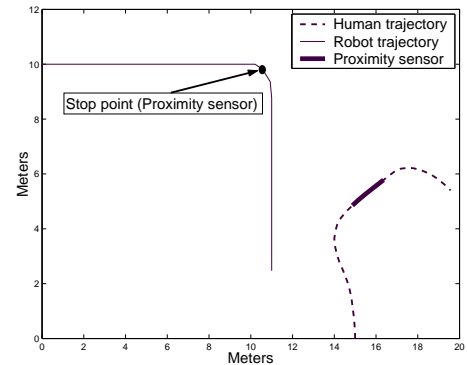
To easily quantify the difference between the proposed solution and the conservative approach on proximity sensors, the probability of human injury is plotted in figure 8, solid line. As in the example above, the graph is derived assuming that if the safety system detects a collision with probability larger than a safety threshold the robot is stopped and waits when the probability of collision decreases. For each given safety threshold (probability of non-minor injury, see



(A)



(B)



(C)

Fig. 7. Human and mobile robot trajectories in the environment for three different runs.

figure 2-B), the percentage of time in which the robot stops against the conservative approach is reported, summing up the time when the robot was stopped for each collision threshold. Even though the human is assuming to move randomly in the area and allowing a probability of injury of 10^{-7} , the robot remains still 34% less than a conservative approach. Notice that the data has been collected simulating a 24 hours day work, in which the human worker and the robot are constantly inside the area.

B. Circular Trajectory

In the second set of simulations, we considered an “8” trajectory, comprised of two tangent circles (see figure 6-B). Of course, this can hardly be a realistic trajectory followed in

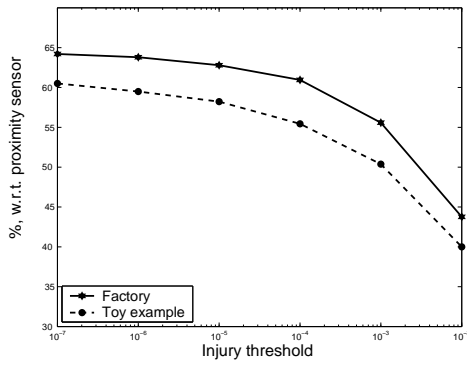


Fig. 8. Probability of injury against percentage of robot stopped time with respect to the conservative approach.

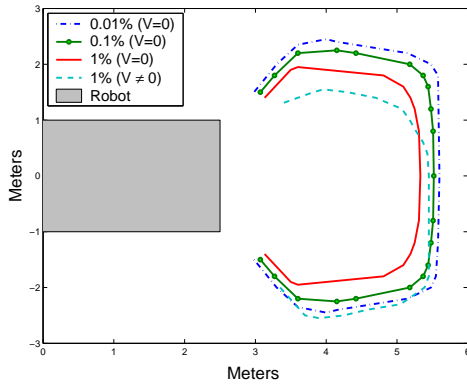


Fig. 9. Pre-computation of collision probability. Each line (in the space of motion) is associated with an equal probability of collision.

a factory floor, but it is useful to show how the safety system deals with the curved trajectory of the robot. In this case, the probability of injury against percentage of robot stopped time with respect to the conservative approach is reported in figure 8, dashed. Again, we can see the radical improvement in the stopping time, and hence in the productivity, even with very low probability thresholds.

C. Computing the Collision Probabilities off-line

If we assume that the robot moves in a single mode (e.g., straight line), it is possible to make a radical simplification in the way the collision probability is computed. Indeed, in this case, not only are the set of the safe and bad states invariant, but the same applies to the transition probability of the DTMC. Building on this consideration, we can compute off-line the probability of a collision and store them in a look up table in which each initial position, velocity and acceleration of the target can be associated with a value of the probability. As an example, we report in figure 9 the curves in space that are associated with a fixed probability of collision (assuming a fixed value for the initial velocity and acceleration). A similar computation can be carried out for the different values of the velocity and acceleration building surfaces associated with equal probability in the six-dimensional space. This approach bears considerable advantages in terms of computation time and the way it can

be applied when the robot is allowed to change its mode (e.g., switching from straight to curved trajectories) represents an interesting future line of research.

VI. CONCLUSIONS

In this paper, we have proposed a methodology for active recognition of dangerous situations in working environment where robotic vehicles and human workers coexist. The procedure is based on the computation of a stochastic approximation of human motion (a DTMC), which is used in an algorithm to predict the probability of a collision or of a trapping. The probability thresholds have been selected considering recent studies produced by the automotive industry that relate the probability of getting an injury with the relative velocity in the collision.

As a future work, we plan to study the Markov approximation of the stochastic differential equation describing the human motion, to study numerically efficient solutions for the computation of the probability of an accident and to test our technique in a real-life deployment.

REFERENCES

- [1] C. Baier, B. Haverkort, H. Hermanns, and J.-P. Katoen. Model-checking algorithms for continuous-time markov chains. *Software Engineering, IEEE Transactions on*, 29(6):524–541, June 2003.
- [2] R.V. Bostelman, T.H. Hongm, R. Madhavan, and T. Y. Chang. Safety standard advancement toward mobile robot use near humans. In *Proc. of Intl. Conf. on Safety of Industrial Automated Systems (RIA SIAS)*, Chicago, IL, US, September 26–28 2005.
- [3] Richards D. Hill J. Cuerden, R. Pedestrians and their survivability at different impact speeds. In *Proceedings of the 20th International Technical Conference on the Enhanced Safety of Vehicles*, Lyon, France, 2007.
- [4] R. Durrett. *Stochastic Calculus*. CRC Press, 1996.
- [5] D. Fox, W. Burgard, S. Thrun, and A.B. Cremers. A hybrid collision avoidance method for mobile robots. In *Proc. of Intl. Conf. on Robotics and Automation*, volume 2, pages 1238–1243, Leuven, Belgium, May 16–20 1998.
- [6] Kevin A. Grasse. Simulation and bisimulation of nonlinear control systems with admissible classes of inputs and disturbances. *SIAM Journal on Control and Optimization*, 46(2):562–584, 2007.
- [7] S. Sastry, J. Hu, M. Prandini. Aircraft conflict detection in presence of a spatially correlated wind field. In *IEEE Trans. on Intelligent Transportation Systems*, 2005.
- [8] H.J. Kushner and Pail Dupuis. *Numerical Methods for Stochastic Control Problems in Continuous Time*. Springer Verlag, 2001.
- [9] M. Kwiatkowska, G. Norman, D. Parker, and J. Sproston. *Modeling and Verification of Real-Time Systems: Formalisms and Software Tools*, chapter Verification of Real-Time Probabilistic Systems, pages 249–288. John Wiley & Sons, 2008.
- [10] X.R. Li and V.P. Jilkov. Survey of maneuvering target tracking. Part I: Dynamic models. *IEEE Transactions on Aerospace and Electronic Systems*, 39(4):1333, 2003.
- [11] Giordano Pola, Antoine Girard, and Paulo Tabuada. Approximately bisimilar symbolic models for nonlinear control systems. *Automatica*, 44(10):2508 – 2516, 2008.
- [12] C. Pradalier, J. Hermosillo, C. Koike, C. Braillon, P. Bessière, and C. Laugier. The cycab: a car-like robot navigating autonomously and safely among pedestrians. *Robotics and Autonomous Systems*, 50:51–67, 2005.
- [13] Y. Ricquebourg and P. Bouthemy. Real-time tracking of moving persons by exploiting spatio-temporal image slices. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 22(8):797–808, 2000.
- [14] R. Singer. Estimating optimal tracking filter performance for manned maneuvering targets. *IEEE Trans. Aerospace and Electronic Systems*, 6(4):473–483, July 1970.