2013 IEEE/RSJ International Conference on
Intelligent Robots and Systems (IROS)
November 3-7, 2013. Tokyo, Japan

Path-consistent safety in mixed human-robot collaborative manufacturing environments

Andrea Maria Zanchettin and Paolo Rocco

***Abstract*— In order to improve production flexibility, it is widely agreed that future working environments will be populated by both humans and robot manipulators, sharing the same workspace. This scenario introduces a series of safety issues which are uncommon in industrial settings where physical separation of robot areas is typically enforced. While several approaches for safe human-robot interaction exist, none of them can be easily integrated with production constraints. This paper discusses the composition of safety constraints with production ones. An algorithm is derived in order to maximize productivity, while guaranteeing a safe separation distance of the robot from the human. Experimental results showing the effectiveness of the approach in a typical industrial setting are also discussed.**

I. Introduction

Robotic aided manufacturing is nowadays a technology with a high level of maturity. Thanks to their flexibility, industrial robots are adopted in several transformation processes such as welding, painting, deburring, assembly, etc.
Despite the benefits of articulated manipulators in industrial production, there are today several obstacles to obtain a more widespread use of robots, especially in SMEs. A robotized station still needs a lot of skilled engineering effort for installation, setup and programming. For safety reasons an industrial robotic station typical requires a dedicated space which is made inaccessible to the humans by means of physical barriers. It is widely agreed that this requirement limits an ubiquitous utilization of robots in SMEs, where space or cost constraints make the installation of robots less affordable. A better integration of small to medium sized robotized stations in production environments may benefit from recent research works on human-robot interaction, [1]. Unfortunately, safety actions do not usually take into account production constraints, which makes this technology difficult to be integrated, at least from an industrial perspective.

Figure 1 shows the relationship between production (in terms of robot velocity) and safety. As the distance between the robot and the human becomes smaller, the velocity of the robot should be reduced accordingly (e.g. according to the minimum distance criterion, see [2]), thus decreasing the productivity of the robot. On the other hand, even in case of a reduced separation distance, the robot should continue its task if its velocity is oriented so that the distance with the human operator will increase.

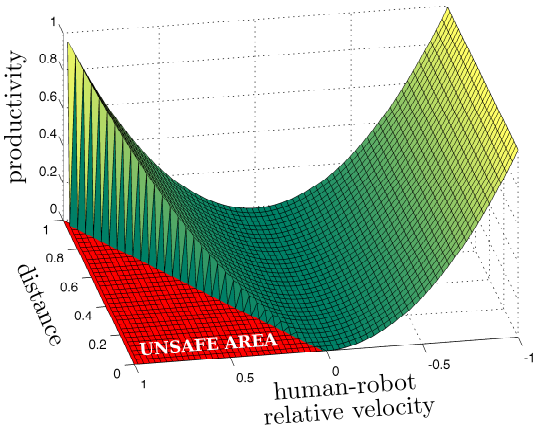

Fig. 1. Relation between productivity and safety

In this scenario, safety plays the role of a hard constraint, in which respect production could be somehow maximized. The robot trajectory must obey the following constraint

$$\text{distance} \geq \text{velocity} \cdot \text{braking time}$$

where the braking time possibly depends on the robot payload, [2]. This way, provided that all the task and safety constraints are satisfied, the robot motion should result as a compromise between production and safety.
Research in the field of safe human robot interaction has been gaining momentum since the last decades. For example, *Suh et al.* proposed in [3] a planning method which accounts for both optimality and safety. Although not specifically targeted to human-robot collaboration, the problem of finding an optimal trajectory is solved by using a dynamic programming method. In [4], the authors proposed a reactive algorithm to modify a pre-planned path in case of moving obstacles using virtual forces generated by virtual potential fields. In [5], the authors propose a control scheme which limits the torque commands of a position controlled robot to values that comply to safety restrictions. In [6] a reactive navigation algorithm is proposed with capabilities of real-time generating collision free trajectories in dynamic environments. Safety issues specifically arising in human-robot interactions were addressed by *Kulic and Croft* in [7], where a control/planning strategy ensuring safety during human-robot interaction is proposed, based on the computation of a danger index and on the impact forces occurring in a potential collision. In [8], a human aware motion planner (HAMP) is proved able

A.M. Zanchettin and P. Rocco are with Politecnico di Milano, Dipartimento di Elettronica, Informazione e Bioingegneria, Piazza L. Da Vinci 32, 20133, Milano, Italy (email: andreamaria.zanchettin@polimi.it; paolo.rocco@polimi.it).

The research leading to these results has received funding from the European Community Seventh Framework Programme FP7/2007-2013 - Challenge 2 - Cognitive Systems, Interaction, Robotics - under grant agreement No 230902 - ROSETTA.

to provide safe robot paths as well as socially acceptable and legible motion profiles. A reactive path planner for mobile manipulators in dynamic environments with moving obstacles has been developed in [9].

Particularly interesting is the so-called dynamic envelope developed by *Vatcha and Xiao* in [10], [11], which is a region, around the robot surface, whose size depends on both robot/obstacle relative position and velocity and is guaranteed to be collision free within a certain prediction horizon. Safe robot control strategies for specific sensing devices, like standards surveillance systems [12] or depth cameras [13], have been also addressed. More recently, a passivity based controller for safe human-robot coexistence has been presented [14]. While it has the nice property of producing slower motion profiles corresponding to a reduced human/robot distance, it cannot impose path constraints.

This paper contributes to the field of human-robot interaction by proposing a novel safety measure depending on both the robot configuration and velocity, which is then used to modify the traversing velocity along a given pre-planned path. In manufacturing environments, the given path cannot be relaxed or modified without violating production constraints. Therefore a methodology to adapt the robot speed is here developed to meet industrial requirements. The properties of the proposed metrics for safety assessment are discussed in the paper. In particular, its linearity in the robot velocity is exploited as an ingredient of an optimization algorithm to solve the problem of maximize the robot productivity, while guaranteeing a safe interaction with the human.

The remainder of this work is organized as follows. Section II describes a metrics inspired by industrial standards for safety assessment. Its linear dependance on the robot velocity will be further exploited in Section III, where a Linear Programming (LP) method will be derived to actively adjust the robot speed in order to meet the safety requirements. The implementation on typical closed-ended robot controllers is discussed as well. Section IV finally presents experimental results about the applicability of the derived method in a selected and relevant case study.

II. Safety constraints

In the following a condition to check whether the current state of motion (i.e. position and velocity) of a given robot manipulator can be regarded as safe or not, with respect to the given position of the human/obstacle, is discussed.

Consider a rigid link represented as a beam, as shown in Fig. 2. The position $\boldsymbol{r_s}$ of each point of the link and its velocity $\boldsymbol{v_s}$ can be written in terms of position and velocity of the two end points as follows:

$$\boldsymbol{r_s} = \boldsymbol{r_a} + s\left(\boldsymbol{r_b} - \boldsymbol{r_a}\right) \quad \boldsymbol{v_s} = \boldsymbol{v_a} + s\left(\boldsymbol{v_b} - \boldsymbol{v_a}\right) \tag{1}$$

where $s \in [0,1]$. Therefore, for a given obstacle (e.g. a part of the human body) detected at position $\boldsymbol{r_{obst}}$, the minimum separation distance can be reduced to the following constraints to be satisfied for every $s \in [0,1]$.

$$\left\|\boldsymbol{r_{obst}} - \boldsymbol{r_s}\right\| \geq t_b \frac{\left(\boldsymbol{r_{obst}} - \boldsymbol{r_s}\right)^T}{\left\|\boldsymbol{r_{obst}} - \boldsymbol{r_s}\right\|} \boldsymbol{v_s} \tag{2}$$

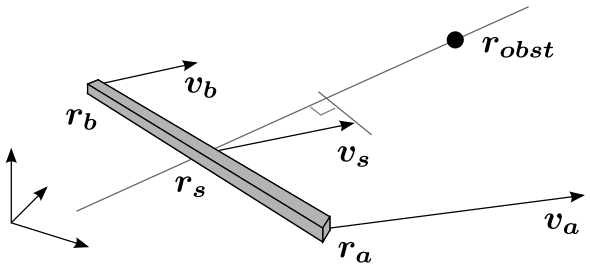

Fig. 2. A rigid beam representing one link

where t_b is the braking time, while

$$\frac{\left(\boldsymbol{r_{obst}} - \boldsymbol{r_s}\right)^T}{\left\|\boldsymbol{r_{obst}} - \boldsymbol{r_s}\right\|} \boldsymbol{v_s}$$

represents the projection of $\boldsymbol{v_s}$ onto the segment connecting $\boldsymbol{r_s}$ to $\boldsymbol{r_{obst}}$. Such a set of constraints can be then rewritten as follows

$$\left\|\boldsymbol{r_{obst}} - \boldsymbol{r_s}\right\|^2 - t_b \left(\boldsymbol{r_{obst}} - \boldsymbol{r_s}\right)^T \boldsymbol{v_s} \geq 0, \forall s \in [0,1] \tag{3}$$

Assuming the link rigid[1] we obtain

$$\left\|\boldsymbol{r_{obst}} - \boldsymbol{r_s}\right\|^2 + \beta s + \gamma \geq 0 \tag{4}$$

where

$$\begin{aligned} \beta &= t_b \left(\boldsymbol{r_b} - \boldsymbol{r_a}\right)^T \boldsymbol{v_a} - t_b \left(\boldsymbol{r_{obst}} - \boldsymbol{r_a}\right)^T \left(\boldsymbol{v_b} - \boldsymbol{v_a}\right) \\ \gamma &= -t_b \left(\boldsymbol{r_{obst}} - \boldsymbol{r_a}\right)^T \boldsymbol{v_a} \end{aligned} \tag{5}$$

A sufficient condition for (4) to be satisfied for all $s \in [0,1]$ is:

$$\beta s + \gamma + \min_s \left\|\boldsymbol{r_{obst}} - \boldsymbol{r_s}\right\|^2 \geq 0 \tag{6}$$

Notice that the first two terms in the left-hand side represent a linear, hence monotonic, function in s while the last term is a constant parameter with respect to s, representing the squared distance between the beam and the point obstacle or, like in [15], the distance between the link and a generic non-convex obstacle in the workspace.

A necessary and sufficient condition for (6) to be satisfied for all $s \in [0,1]$ is that the two extremes of the linear function are consistent with the inequality, i.e.

$$\begin{cases} \gamma + \min_s \left\|\boldsymbol{r_{obst}} - \boldsymbol{r_s}\right\|^2 \geq 0, s = 0 \\ \beta + \gamma + \min_s \left\|\boldsymbol{r_{obst}} - \boldsymbol{r_s}\right\|^2 \geq 0, s = 1 \end{cases} \tag{7}$$

Finally, by considering all the n robot links and noticing that β and γ are linear in the robot velocity $\dot{\boldsymbol{q}}$, the minimum separation distance criterion can be written as follows:

$$t_{b,i} \boldsymbol{E_i} \dot{\boldsymbol{q}} \leq \boldsymbol{f_i}, \forall i = 1, \ldots, n \tag{8}$$

where $t_{b,i}$ is the maximum joint breaking time up to joint i and

$$\begin{aligned} \boldsymbol{E_i} &= \begin{bmatrix} \left(\boldsymbol{r_{obst}} - \boldsymbol{r_a}\right)^T \boldsymbol{J_a} \\ \left(\boldsymbol{r_{obst}} - \boldsymbol{r_a}\right)^T \boldsymbol{J_b} - \left(\boldsymbol{r_b} - \boldsymbol{r_a}\right)^T \boldsymbol{J_a} \end{bmatrix} \\ \boldsymbol{f_i} &= \min_s \left\|\boldsymbol{r_{obst}} - \boldsymbol{r_s}\right\|^2 \begin{bmatrix} 1 \\ 1 \end{bmatrix} \end{aligned} \tag{9}$$

[1]Hence $\left\|\boldsymbol{r_b} - \boldsymbol{r_a}\right\|$ is constant.

Remark The set of inequalities in (8) represents an easy way to check whether the current robot state of motion satisfies the minimum separation distance or, in other terms, if the current robot velocity is sufficiently low to allow the robot to stop before a collision occurs.

As an example, Fig. 3 shows the regions to be avoided around an industrial robot in position $\boldsymbol{q}$ with the following nominal velocity $\dot{\boldsymbol{q}}$

$$\boldsymbol{q} = \begin{bmatrix} 20 & -20 & 40 & 0 & 0 & 0 \end{bmatrix}^T \text{deg}$$

$$\dot{\boldsymbol{q}} = \begin{bmatrix} 100 & 20 & 50 & 0 & 10 & 10 \end{bmatrix}^T \text{deg}/s$$

In case an obstacle or part of the human body $\boldsymbol{r_{obst}}$ lies within the highlighted region, the stopping time is not sufficient to avoid collisions.

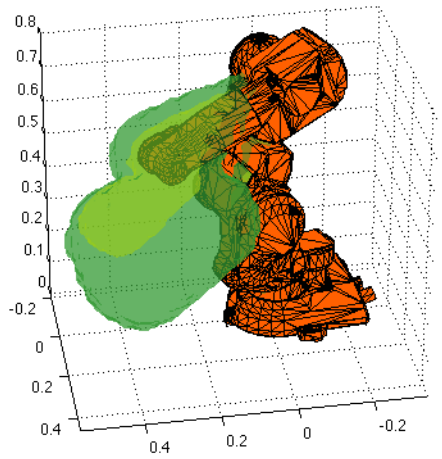

Fig. 3. Regions to be avoided around an industrial robot in motion with nominal velocity (green) and reduced speed to 50% (yellow)

III. Path-consistent safe motion planning

In this Section, we introduce an algorithm to efficiently solve the safety-oriented path-constrained motion planning problem by suitably scaling the trajectory in time. Moreover, the adaptation of the trajectory scaling technique to closed-ended industrial controller will be discussed. We here assume the following parameterization of the task:

$$\boldsymbol{x}(\tau) \quad \boldsymbol{x}'(\tau) = \frac{\partial \boldsymbol{x}}{\partial \tau} \tag{10}$$

where $\tau \in [0, t_f]$ is the time variable and $\boldsymbol{x}(\cdot)$ is a differentiable function specifying the desired trajectory. Let $\delta \in [0, 1]$ be a scalar quantity adopted to kinematically scale the trajectory in time. The value $\delta = 1$ corresponds to the nominal trajectory, i.e. executed at programmed speed, while $\delta = 0$ forces the robot to stop. In order to exploit the sensor-based trajectory scaling, similarly to the one originally developed in [16], we introduce the following Linear Programming (LP) optimization problem:

$$\max_{\delta, \dot{\boldsymbol{q}}} \delta \tag{11a}$$

$$t_{b,i} \boldsymbol{E_i}(\boldsymbol{q}) \dot{\boldsymbol{q}} \leq \boldsymbol{f_i}(\boldsymbol{q}), \forall i = 0, \ldots, n-1 \tag{11b}$$

$$\boldsymbol{J}(\boldsymbol{q}) \dot{\boldsymbol{q}} = \delta \boldsymbol{x}'(\tau) + \boldsymbol{K} \boldsymbol{\epsilon}(\tau, \boldsymbol{q}) \tag{11c}$$

$$0 \leq \delta \leq 1 \tag{11d}$$

where $\boldsymbol{\epsilon}$ is the kinematic error, e.g. $\boldsymbol{\epsilon} = \boldsymbol{x} - \boldsymbol{k}(\boldsymbol{q})$, being $\boldsymbol{k}(\cdot)$ the forward kinematics function, and $\boldsymbol{K} = \boldsymbol{K}^T > 0$ a control weight.
If $\boldsymbol{J}(\boldsymbol{q})$ is a square non singular matrix, which happens in 6-dof industrial manipulators away from singular configurations, the kinematic constraints can be solved with respect to $\dot{\boldsymbol{q}}$, yielding the following simplified LP optimization algorithm:

$$\max_{\delta} \delta \tag{12a}$$

$$t_{b,i} \boldsymbol{E_i}(\boldsymbol{q}) \dot{\boldsymbol{q}} \leq \boldsymbol{f_i}(\boldsymbol{q}), \forall i = 0, \ldots, n-1 \tag{12b}$$

$$0 \leq \delta \leq 1 \tag{12c}$$

where $\dot{\boldsymbol{q}} = \boldsymbol{J}(\boldsymbol{q})^{-1} (\delta \boldsymbol{x}'(\tau) + \boldsymbol{K} \boldsymbol{\epsilon}(\tau, \boldsymbol{q}))$.

When it comes to implement the algorithm in discrete time, the time update law can be expressed as follows:

$$\tau_{k+1} = \tau_k + \Delta t \delta_k \tag{13}$$

where Δt is the discrete time step and δ_k is the output of (12) at the discrete time step k, while matrices in (12) are evaluated at each discrete time step before solving the optimization problem. Figure 4 shows the architecture of the described algorithm. The block highlighted in gray implements the algorithm in (12) and is also responsible for sending joint position and velocity references to the lower-level controller.

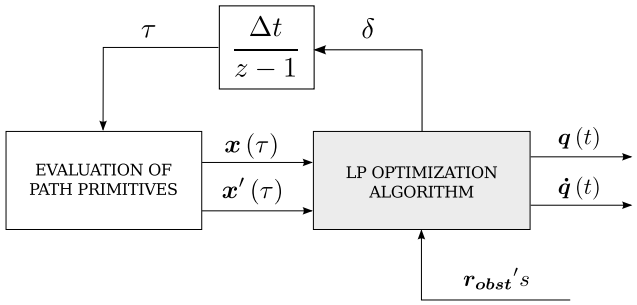

Fig. 4. Trajectory scaling algorithm for path-consistent and safe motion planning

Modern industrial controllers have dedicated functionalities to implement kinematic scaling of pre-planned trajectories in real-time. In this case, the methodology proposed so far in this Section can be adapted in order to completely exploit the available functionalities (i.e. kinematic inversion and online trajectory scaling) of the proprietary industrial controller. Assuming that joint positions and velocities are available through a real-time interface, an additional PC can solve the following LP problem:

$$\max_{\delta} \delta \tag{14a}$$

$$t_{b,i}\boldsymbol{E_i}\left(\boldsymbol{q}\right)\dot{\boldsymbol{q}}\delta \leq \boldsymbol{f_i}\left(\boldsymbol{q}\right), \forall i = 0, \ldots, n-1 \tag{14b}$$

$$0 \leq \delta \leq 1 \tag{14c}$$

where the optimal (maximum) value δ is forwarded to the robotic controller to be interpreted as a trajectory scaling command, as shown in Fig. 5. The block highlighted in gray implements the algorithm in (14) and, differently from the previous implementation, is not responsible for sending reference values to the low-level controller. Instead, the proprietary controller will interpret the value of δ and reduces the speed accordingly. Before applying the

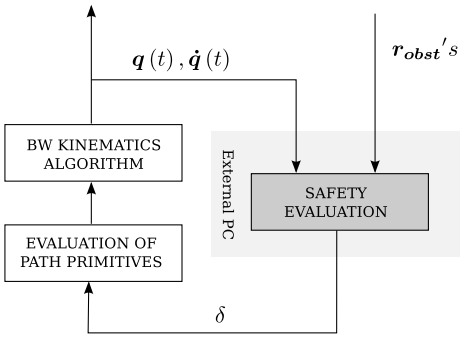

Fig. 5. Data flow between an industrial controller and an interfaced external PC for safety evaluation

described methodology to a relevant case study, some properties of the algorithm are discussed in the following.

First, notice that both the two LP algorithms in (12) and (14) guarantee feasibility of the output trajectory. In fact, the method assumes that a reference trajectory, which is feasible and singularity free, is given, while the only action to be performed is a kinematic rescaling which eventually affects the motion of the robot, by reducing the traversing speed. It follows that the modified trajectory still preserves the kinematic properties (joint position and velocity limits as well as singularity avoidance) of the programmed one.
Moreover, the algorithm can be easily extended to the case of multiple obstacles. In this case parallelization can be also adopted. Algorithms in (12) or (14) can be run concurrently for each obstacle, thus computing several values of δ, one per each obstacle. A voting system can be adopted to select the worst case value of δ. Finally, notice that the LP problem in (14) can be actually solved without using a LP-solver. Each inequality can be processed independently (and possibly in parallel) to compute a value for δ as follows:

$$\delta_i = \begin{cases} \dfrac{\boldsymbol{f_i}\left(\boldsymbol{q}\right)}{t_{b,i}\boldsymbol{E_i}\left(\boldsymbol{q}\right)\dot{\boldsymbol{q}}} & \boldsymbol{E_i}\left(\boldsymbol{q}\right)\dot{\boldsymbol{q}} > 0 \\ 1 & \text{otherwise} \end{cases} \tag{15}$$

and then return the minimum within δ_i's.
Finally, notice that in case the algorithm is split into two separate computers, namely the industrial robot controller and the external PC, some communication delay can be somehow expected. Therefore in (15), one can adopt the following quantity

$$t_{b,i} + t_{react} + t_{comm}$$

in place of the pure braking time $t_{b,i}$, where t_{react} is the reaction time of the safeguarding system (typically needed e.g. to handle interrupts), while t_{comm} represents an estimate of the communication delay.

IV. Case study

In this Section we describe a relevant case study which might benefit from the proposed approach. We here consider a task for which the robot has to maintain the programmed path, while the traversing speed can be reduced as need be, without disrupting the production.

A. Implementation

The 6 axes ABB IRB 140 robot with 6 kg maximum payload was used for this purpose. Relevant kinematic parameters are listed in Tab. I. The robot is position controlled

axis, i	$a_i[m]$	$d_i[m]$	α_i[deg]	θ_i	$t_b[ms]$
1	0.07	0	90	q_1	377
2	0.36	0	0	q_2	351
3	0	0	90	q_3	296
4	0	0.38	-90	q_4	232
5	0	0	90	q_5	267
6	0	0.065	0	q_6	245

TABLE I
KINEMATIC PARAMETERS

by an industrial ABB IRC 5 controller and programmed through the proprietary RAPID language. Typical movement instructions have the following syntax:

```
MoveL target, speed, zone, tool;
```

where target specifies both position and orientation of the tool frame, while speed and zone are parameters for the path planner, and tool specifies the parameters of the tool, if any. In particular the speed parameter specifies the programmed speed along the given path. The RAPID language also allows the programmer to reduce the programmed speed by a certain percentage, e.g. in response to sensed events, with the following instruction

```
SpeedRefresh p;
```

where $0 \leq p \leq 100$ indicates the desired speed reduction, and clearly plays the role of δ in (14).
The operator, regarded as a moving obstacle, is able at any time to enter the working area of the robot for inspection. For workspace surveillance a range camera (MICROSOFT KINECT) with the OPENNI drivers has been selected, see [17]. The LP algorithm in (14) has been implemented on an external real-time PC using the approach in (15) and tested against the solution provided by the *GNU Linear Programming Kit*, available online at [18]. Figure 6 shows the overall architecture.

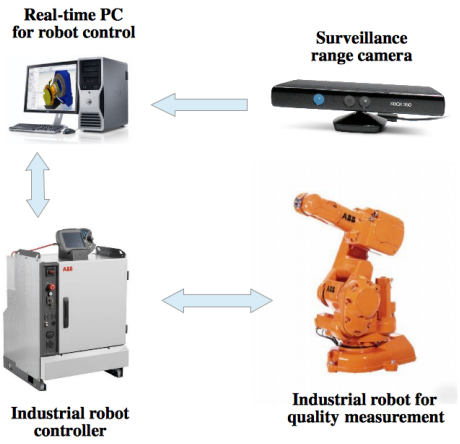

Fig. 6. System architecture

As for the computation of distances in (6), each link of the robot is regarded as a segment, while the human operator is regarded as a set of capsules (cylinders with hemispherical extremities), as shown in Fig. 7. The distance between a segment and a capsule can be easily calculated with few basic operations, as described in [19]. The evaluation of distances and the computation of the optimum value of δ have been implemented on an external Linux Xenomai PC with cycle time of $4\ ms$. Finally, the feedback to the robot

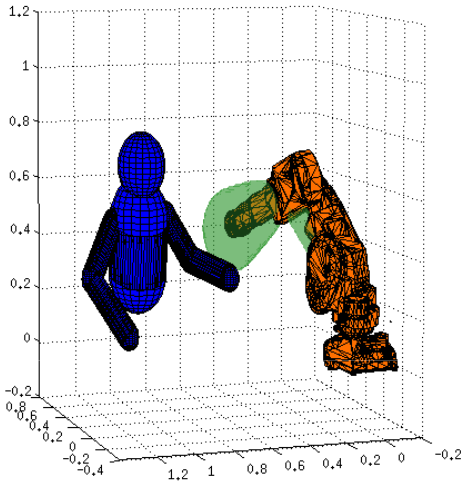

Fig. 7. Human-robot interaction showing the representation of the human as the union of capsules and the regions around the robot to be avoided depending on the current robot speed

controller of the computed value δ as in Fig. 5 has been implemented with standard TCP/IP sockets, with an expected communication delay of $t_{comm} = 100\ ms$ (TPC messages are sent at $10\ Hz$). Whenever an updated value of δ is available, the external PC sends this information to the ABB IRC 5 industrial controller which periodically polls for new messages and correspondingly updates the robot velocity, according to the following excerpt of RAPID code:

```
! Safety watcher
WHILE TRUE DO
  SocketReceive socket\Str:=buffer;
  StrToVal(buffer,p);
  SpeedRefresh p;
ENDWHILE
```

In order to avoid chattering behaviour of variable δ around zero, causing multiple activations $\delta > 0$ and suspensions of the task $\delta = 0$, a small hysteresis has been implemented, avoiding the robot to resume the task until a minimum safety distance can be guaranteed. In other words, once δ has been set to zero, the output of the algorithm in (15) is forced to zero, until the minimum distance, i.e. $\min_i \boldsymbol{f_i}$, see (9), exceeds a predefined threshold. This hysteresis has been implemented on the feedback from the External PC to the robot controller, see Fig. 5.

Finally, the motion of the robot has been programmed with RAPID instructions and consists of several repetitions of a working cycle:

```
! Motion task
WHILE TRUE DO
  off := 0;
  FOR j FROM 0 TO 15 DO
    MoveL Offs(p0,-10*off,0,0),v300,...;
    MoveL Offs(p1,-10*off,0,0),v300,...;
    off := off + 1;
    MoveL Offs(p1,-10*off,0,0),v300,...;
    MoveL Offs(p0,-10*off,0,0),v300,...;
    off := off + 1;
  ENDFOR
ENDWHILE
```

B. Experiments

We discuss here the experiments performed with an industrial robot controller, following the approach described so far. The experimental setup is sketched in Fig. 8. The

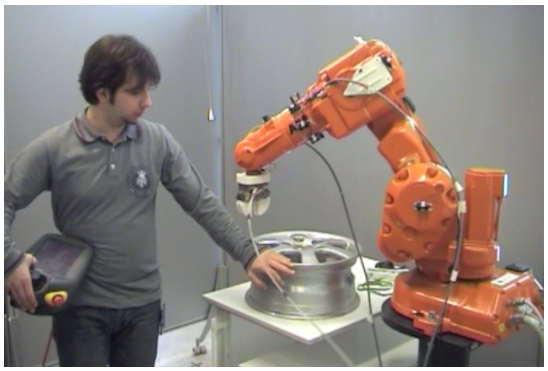

Fig. 8. Experimental setup

nominal operation of the robot consists in following a predefined path, e.g. for workpiece inspection, with programmed velocity of $300\ mm/s$. When the human is not present in the scene, the robot is able to accomplish its task at programmed speed, with no interruption. On the other hand, when the human enters the workspace of the robot, e.g.

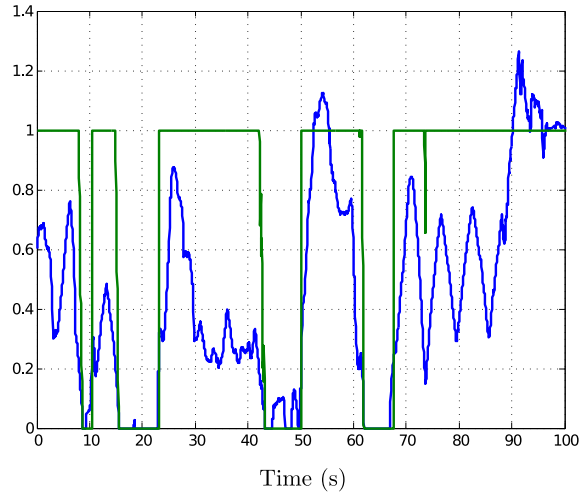

Fig. 9. Optimal value of δ (green) and measured minimum distance (blue) during the experiment

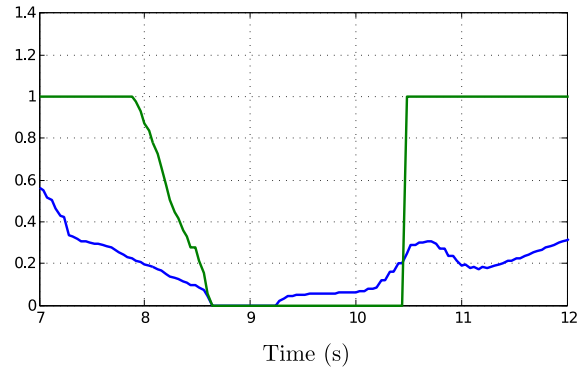

Fig. 10. Detail of δ (green) and measured minimum distance (blue) during the experiment

for inspection, her/his position is immediately tracked by the surveillance depth camera and the algorithm in (14) is cyclically executed. Figure 9 shows the time history of δ and of the minimum distance $\sqrt{\min_i \boldsymbol{f_i}}$ from the human operator to the robot, while Fig. 10 details a shorter time interval of the experiment. As expected, the robot is able to reduce its speed consistently with its state of motion and with the (minimum) distance to the human operator. At the same time, the value of δ is optimized to maintain the productivity at maximum level, provided that safety is guaranteed. From Fig. 10, it should be noticed that the robot is able to immediately stop whenever the human comes in contact with the robot itself, thus allowing soft contact with minimum energy exchange. As previously discussed, when the robot is stopped, the reactivation of the task is postponed until its distance to the human exceeds a predefined threshold. This fact can be verified in Fig. 10 around time instant $t = 10\ s$.

V. Conclusions

This paper contributes in introducing a metrics for human-robot safety evaluation which not only depends on the relative distance between the robot and the human, but also on their relative velocity. Dynamic and control characteristics, such as the braking time of each axis and the typical reaction time of the controller are also taken into account. The properties of the proposed metrics have been discussed, and experiments have shown the effectiveness of the approach.

Acknowledgment

The authors would like to thank Björn Matthias and Hao Ding (ABB Corporate Research, Ladenburg, Germany) for their useful suggestions and help.

References

[1] S. Haddadin, A. Schaeffer, and G. Hirzinger, "Requirements for safe robots: measurements, analysis and new insights," *International Journal of Robotics Research*, vol. 28, pp. 1507–1527, 2008.

[2] *ANSI/RIA R15 06 1999 "Safety requirements for industrial robots and robot systems"*.

[3] S. H. Suh and K. G. Shin, "A variational dynamic programming approach to robot-path planning with a distance-safety criterion," *IEEE Journal of Robotics and Automation*, vol. 4, no. 3, pp. 334–349, 1988.

[4] S. Quinlan and O. Khatib, "Elastic bands: Connecting path planning and control," in *IEEE International Conference on Robotics and Automation*, 1993.

[5] J. Heinzmann and A. Zelinsky, "Quantitative safety guarantees for physical human-robot interaction," *International Journal of Robotics Research*, vol. 22, no. 7-8, pp. 479–504, 2003.

[6] S. Petti and T. Fraichard, "Safe motion planning in dynamic environments," in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2005.

[7] D. Kulić and E. A. Croft, "Real-time safety for human–robot interaction," *Robotics and Autonomous Systems*, vol. 54, no. 1, pp. 1–12, 2006.

[8] E. A. Sisbot, L. F. Marin-Urias, R. Alami, and T. Simeon, "A human aware mobile robot motion planner," *IEEE Transactions on Robotics*, vol. 23, no. 5, pp. 874–883, 2007.

[9] J. Vannoy and J. Xiao, "Real-time adaptive motion planning (RAMP) of mobile manipulators in dynamic environments with unforeseen changes," *IEEE Transactions on Robotics*, vol. 24, no. 5, pp. 1199–1212, 2008.

[10] R. Vatcha and J. Xiao, "Perceiving guaranteed continuously collision-free robot trajectories in an unknown and unpredictable environment," in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2009.

[11] ——, "An efficient algorithm for on-line determination of collision-free configuration-time points directly from sensor data," in *IEEE International Conference on Robotics and Automation*, 2010.

[12] S. Kuhn and D. Henrich, "Fast vision-based minimum distance determination between known and unknown object," in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2007.

[13] F. Flacco, T. Kröger, A. De Luca, and O. Khatib, "A depth space approach to human-robot collision avoidance," in *IEEE International Conference on Robotics and Automation*, 2012.

[14] A. M. Zanchettin, B. Lacevic, and P. Rocco, "A novel passivity-based control law for safe human-robot coexistence," in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2012.

[15] S. Quinlan, "Efficient distance computation between non-convex objects," in *IEEE International Conference on Robotics and Automation*, 1994.

[16] S. Haddadin, A. Albu-Schaffer, A. De Luca, and G. Hirzinger, "Collision detection and reaction: A contribution to safe physical human-robot interaction," in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2008.

[17] "OpenNI - the standard framework for 3D sensing," downloaded from http://http://www.openni.org/, Mar. 2013.

[18] "GNU Linear Programming Kit," downloaded from http://www.gnu.org/software/glpk/, Feb. 2013.

[19] C. Ericson, *Real-Time Collision Detection*. Morgan Kaufmann, 2004.