Online verification of multiple safety criteria for a robot trajectory

Dario Beckert*, Aaron Pereira* and Matthias Althoff*

Abstract— **Ensuring the safety of humans in a collaborative environment with industrial robots is a major concern of human-robot co-working. Most current approaches give no formal guarantee of safety; where such guarantees are given, accounting for the unpredictability of the human may limit robot efficiency. We therefore developed a novel robot arm trajectory planner that formally guarantees the safety of humans from the robot for every possible human behaviour without restricting the robot more than necessary. To achieve this, the trajectory planner verifies online that the robot adheres to two separate safety criteria derived from ISO standards using reachable occupancies of surrounding humans and the robot. While one criterion anticipates all possible movement of the human and provides a reasonable safety guarantee, the other criterion additionally guarantees strict safety as long as the human behaves as expected according to ISO standards. We implemented the trajectory planner for a real robot arm and show some experimental results.**

I. INTRODUCTION

A limiting factor for the introduction of robots working alongside humans in industry is the challenge of guaranteeing safety. To avoid injury from collisions, robots must react to the behaviour of nearby humans, whose future motion is hard to predict. Accounting for only the most likely subset of the motion, as in most previous approaches, introduces the risk that the robot does not behave safely during unexpected motion, whereas accounting for all physically possible motion could restrict the robot performance to a large extent.

We present a novel approach whereby we simultaneously guarantee a strong safety condition accounting for expected human motion, and a weaker safety condition accounting for all possible movement. This avoids over-cautious robot behaviour which would lead to poor performance, while nevertheless guaranteeing safety for nearby humans. We guarantee these conditions in our formally verified trajectory planner for an industrial robot arm. The trajectory planner is based on the principle that *no movement is executed without previously being formally verified to be safe*.

Since the human moves fast and with uncertain intention, most approaches which predict future human movement to avoid collisions are probabilistic. Ding et al. present a Hidden Markov Model [**?**] to predict hand occupancy in reaching motions. Mainprice and Berenson use a Gaussian Mixture Model to estimate the probability of space around the robot being occupied by a human [**?**]; this is used in a trajectory planner. On a higher level, [**?**] uses information from objects in the environment to predict most likely human motion. Though probabilistic approaches often work well, they cannot be relied upon to provide a strict guarantee of safety, e.g. when a human performs an unexpected movement.

To account for this, Petti and Fraichard [**?**] propose planning short-term plans where an agent can provably avoid states which would lead to collision (Inevitable Collision States). This principle has been applied to autonomous vehicles [**?**] and robot manipulators [**?**]. In most cases, a robot has been considered safe when it is stationary. However, for close co-existence, this requirement may be too restrictive — when the human is near the robot, accounting for all movement may mean that the robot cannot move at all.

If collisions cannot be avoided, then collision severity should be minimised. Soft robots [**?**] are designed to be inherently safe, but the difficulty of controlling them accurately limits application. Impact energy minimisation is considered in [**?**], and the recent Technical Standard [**?**] describes limiting of impact force and pressure, among other safety criteria. The ISO standard governing human-robot collaborative operation [**?**] describes a *Safety-rated Reduced Speed* mode of operation which can be used where humans are nearby, with the justification that this should "allow sufficient time for people either to withdraw from the hazardous area or to stop the robot". Here, the Cartesian speed of the tool centre point (TCP) is reduced to below a certain value, maximally $0.25\frac{m}{s}$.

Our novel approach, however, guarantees that the robot will stop before the human can reach it, when the human behaves according to expectations. In the case that the human executes unexpected movement, it guarantees that the Cartesian speed of the robot at the point of impact is less than a certain value. Next, we introduce the trajectory planner and give a formal definition of the safety criteria. In Sec. III we present the developed trajectory planning algorithm step by step; in Sec. IV we show how to calculate robot and human occupancies in a conservative way. In Section V we describe our experimental results and conclude in Sec. VI.

II. PROBLEM STATEMENT AND TERMINOLOGY

We develop a robot trajectory planner that is able to adapt online to human behaviour to meet two safety criteria. To accomplish this, we verify the trajectory piecewise. A long-term trajectory is given by a higher-level planner. Since it would unnecessarily restrict the robot's movement if we were to certify the safety of the entire long-term trajectory, we calculate short-term plans in smaller steps ahead and formally verify the safety conditions for each of those, as shown in Fig. 1.

*The authors are with the Department of Informatics, Technical University of Munich, 85748 Garching, Germany. d.beckert@tum.de, aaron.pereira@tum.de, althoff@in.tum.de

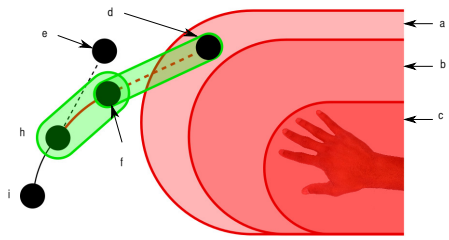

Fig. 1. The robot is currently moving on $[t_k, t_{k+1}]$ while verifying the short-term plan $[t_{k+1}, t_e]$. Since the reachable occupancies of robot (green capsules) and human (red capsules) intersect, this movement cannot be guaranteed to be safe and the robot executes the previously verified failsafe manoeuvre $[t_{k+1}, t_{e,prev}]$. Although the failsafe manoeuvre is shown off-path for illustration purposes, it is actually path-consistent with the desired trajectory.

A short-term plan consists of a piece of the desired long-term plan followed by a failsafe manoeuvre that brings the robot to a stop. As proposed in [**?**], we verify the subsequent short-term plan while executing the current one. Prior to t_k, we have verified that the short term plan from t_k to $t_{e,prev}$ is safe. While executing the first part from t_k to t_{k+1}, we verify the next short-term plan, which consists of the piece of the desired plan from t_{k+1} until t_{k+2}, followed by a failsafe manoeuvre from t_{k+2} to t_e. If the verification fails, at t_{k+1} the robot executes the failsafe manoeuvre from t_{k+1} to $t_{e,prev}$, otherwise it moves along the now guaranteed safe piece of the desired plan in $[t_{k+1}, t_{k+2}]$. We describe in III-A how these short-term plans are generated.

As previously mentioned, we verify two criteria: that the robot stops before the human reaches it, as long as the human moves as expected, and that the robot velocity is under a predefined speed, in case the human moves unexpectedly. We take expected movement to be defined by the maximum speed of the upper body given in ISO 13855 [**?**]. In [**?**], we show that this criterion is less conservative than the actual extreme movements of the human. To account for unexpected movement, we use a model from [**?**], which accounts for all human movement found in human-robot co-existence scenarios. To be able to provide a formal definition of safety we first define reachable occupancies:

Definition 1: ISO OCCUPANCY AND REACHABLE OCCUPANCY: Let $\gamma(t, \mathbf{x}) \subset \mathbb{R}^3$ be the set in Cartesian space occupied by the human at time t, for a state $\mathbf{x}$. The reachable occupancy over a time interval $[t_a, t_b]$ is

$$\Gamma_r([t_a, t_b]) = \{\gamma(t, \mathbf{x}) \,|\, \mathbf{x} \in \mathcal{X}(t), t \in [t_a, t_b]\},$$

where $\mathcal{X}(t)$ is the set of all states physically reachable at time t. We define the ISO occupancy:

$$\Gamma_{ISO}([t_a, t_b]) = \{\gamma(t, \mathbf{x}) \,|\, \mathbf{x} \in X_{ISO}(t), t \in [t_a, t_b]\},$$

where $X_{ISO}(t)$ is the set of all states reachable according to the condition that the maximum speed of any body part is $1.6\frac{m}{s}$ as specified in [**?**].

Γ_r captures the human model of all possible movement whereas Γ_{ISO} describes the human movement expected according to the ISO standard. They are represented as *capsules*, which are the Minkowski sum ($\oplus$) of a line segment G and a sphere H, i.e. $G \oplus H = \{g + h \mid g \in G, h \in H\}$. The *defining points* of the capsule are the endpoints of G. Examples of capsules are shown in Fig. 5.

We can now formally define the safety criteria our trajectory guarantees to fulfil. The strong guarantee states:

Definition 2: STATIONARY CRITERION: Let $\mathbf{q}(t)$ be the joint positions of a robot at time t, and $F(\mathbf{q}(t)) \subset \mathbb{R}^3$ be the set of Cartesian space it occupies. Then for any nearby humans the following holds:

$$\forall t : (F(\mathbf{q}(t)) \cap \Gamma_{ISO}(t) = \emptyset) \vee (\dot{\mathbf{q}}(t) = \mathbf{0}).$$

Either the robot is already stationary or it can stop before the human could possibly come into contact with it by performing expected movement. The weaker guarantee states:

Definition 3: REDUCED SPEED CRITERION: Let $\mathbf{q}(t)$ be the joint positions of a robot at time t, and $F(\mathbf{q}(t)) \subset \mathbb{R}^3$ be the set of Cartesian space it occupies. Further, let $v(\mathbf{q}(t), \dot{\mathbf{q}}(t))$ be the maximum speed of any point on the robot in Cartesian space. Then for any nearby humans the following holds:

$$\forall t : (F(\mathbf{q}(t)) \cap \Gamma_r(t) = \emptyset) \vee (v(\mathbf{q}(t), \dot{\mathbf{q}}(t)) \leq v_{\max})$$

where $v_{\max}$ is a velocity limit to be specified in advance. Note that in contrast to Def. 2, we use Γ_r and not Γ_{ISO}, since we account for all human motion.

We specify some conventions which we will use for the rest of the paper: t_k is the current time and the short term plan to be verified runs from time t_{k+1} to time t_e. The latter is the time the robot comes to a stop, i.e., the time after which the robot is safe according to the Stationary Criterion in Def. 2. Time $t_{e,v_{\max}}$ is the time on the short-term plan when the velocity of all points on the robot falls (and stays) below $v_{\max}$, i.e., from this time on, the robot is safe according to the Reduced Speed Criterion in Def. 3. We next present the methodology for our approach.

III. TRAJECTORY PLANNING

This section describes our trajectory planner. Algorithm 1 shows a high-level overview; Subsection III-A explains the planning of the short-term plan and Subsection III-B discusses the failsafe manoeuvre for the reduced speed criterion. The algorithm is running continuously while the robot executes a long-term trajectory. In the following discussion, we show how to generate the short-term plan.

A. Generation of short-term plan

We here explain lines 4 and 5 of Algorithm 1. We express the global trajectory as a mapping of a time parameter s to joint positions, $\boldsymbol{\xi} : [0, s_f] \rightarrow \mathcal{Q}$ where $\mathcal{Q} \subseteq \mathbb{R}^n$ is the joint space. Since the focus of this work is not path planning, the spatial paths of the robot are pre-programmed, and the robot executes a failsafe manoeuvre simply by scaling the rate of change of the time parameter s to zero, while keeping accelerations and jerks within limits. In other words, at the end of the failsafe manoeuvre, $\dot{s} = 0$. Time parameter scaling

is a well-known way of temporally modifying trajectories, see e.g. [**?**].

Algorithm 1 Formally verified trajectory planner

while not at end of global trajectory **do**
  Execute verified plan from t_k to t_{k+1}
  // calculate new short-term plan
  Calculate next desired position from $\zeta_1(t_{k+2})$
  Calculate failsafe manoeuvre ζ_0 starting at t_{k+2} to t_e
  // calculate robot occupancies
  Predict $F(\mathbf{q}([t_{k+1}, t_e]))$ for stationary criterion
  Obtain $t_{e,v_{\max}}$ on the short-term plan
  Predict $F(\mathbf{q}([t_{k+1}, t_{e,v_{\max}}]))$ for reduced speed criterion
  // calculate human reachable occupancies
  Calculate $\Gamma_{ISO}([t_{k+1}, t_e])$; verify stationary criterion
  Calculate $\Gamma_r([t_{k+1}, t_{e,v_{\max}}])$; verify red. speed criterion
  // update plan
  if proposed short-term plan satisfies both criteria **then**
    Adopt short-term plan
  else
    Continue on failsafe manoeuvre from current plan

If the robot has previously been verified unsafe and has had to execute the failsafe manoeuvre, during time t_{k+1} to t_{k+2} it should also try to recover to full speed. It does this by planning a *recovery manoeuvre*, at the end of which $\dot{s} = 1$. This is illustrated in Fig. 2. We next define a *time-scaling plan*, and then the *failsafe* and *recovery* manoeuvres:

Definition 4: TIME-SCALING PLAN: Given s_0, $\dot{s}_0$ and $\ddot{s}_0$ as the values of s, $\dot{s}$ and $\ddot{s}$ at time t_0, and $\eta \in [0, 1]$ a time-scaling plan starting at t_0 is a monotone function $\zeta_\eta : [t_0, t_1] \to [0, s_f]$ where $\zeta_\eta(t_0) = s_0$, $\dot{\zeta}_\eta(t_0) = \dot{s}_0$ and $\ddot{\zeta}_\eta(t_0) = \ddot{s}_0$, $\dot{\zeta}_\eta(t_1) = \eta$ and $\ddot{\zeta}_\eta(t_1) = 0$.

Definition 5: FAILSAFE AND RECOVERY MANOEUVRE: A failsafe manoeuvre ζ_0 starting at t_0 is a time scaling plan where $\eta = 0$. A recovery manoeuvre ζ_1 starting at t_0 is a time scaling plan where $\eta = 1$.

The short-term plan $\zeta : [t_{k+1}, t_e] \to [0, s_f]$ is planned thus: If from time t_k to t_{k+1} the robot is executing the failsafe manoeuvre, then we plan a recovery manoeuvre ζ_1 starting at t_{k+1}. If the robot is already performing a previously-planned recovery manoeuvre from t_k to t_{k+1}, we do not plan a new one. The position of the robot at time t_{k+2} is therefore $\boldsymbol{\xi}(\zeta_1(t_{k+2}))$. We then calculate a failsafe manoeuvre ζ_0 starting at t_{k+2} until t_e; the position of the robot at any time $t_{k+2} < t \leq t_e$ is $\boldsymbol{\xi}(\zeta_0(t))$. This constitutes the short-term plan, which is shown in bold in Fig. 2.

We next show how to generate the failsafe and the recovery manoeuvres ζ_0 and ζ_1, subject to joint acceleration and jerk limits. All norms are Euclidean.

1) Planning manoeuvres with limited acceleration and jerk: We adapt the method from [**?**]. This finds a time-optimal trajectory of joints $\mathbf{q}$ in time, subject to limits on $\ddot{\mathbf{q}}$ and $\dddot{\mathbf{q}}$, and given the values of $\mathbf{q}$, $\dot{\mathbf{q}}$ and $\ddot{\mathbf{q}}$ at the start and $\dot{\mathbf{q}}$ and $\ddot{\mathbf{q}}$ at the end of the trajectory. Since the failsafe and recovery manoeuvres are not trajectories of joint positions but instead trajectories of s in time, we use the same method to find a trajectory of s, subject to limits on $\ddot{s}$ and $\dddot{s}$.

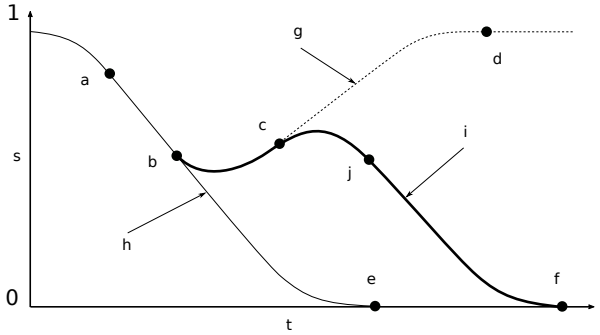

Fig. 2. Illustration of the planning of failsafe and recovery manoeuvres. From time t_k to t_{k+1} the robot executes the verified short term plan (here, the failsafe manoeuvre) and plans the next short term plan (bold line), which consists of one step along the recovery manoeuvre until t_{k+2} followed by a (new) failsafe manoeuvre.

By setting these limits conservatively, we can account for maximum allowable joint accelerations and jerks.

We call s_0 the position at the start of the manoeuvre and s_1 the position at the end. The joint accelerations and jerks are $\ddot{\boldsymbol{\xi}}(s)$ and $\dddot{\boldsymbol{\xi}}(s)$, which we must guarantee to be less than $\mathbf{a}_m$ and $\mathbf{j}_m$. Note that $\boldsymbol{\xi}, \mathbf{a}_m, \mathbf{j}_m \in \mathbb{R}^n$.

Lemma 1: Let $\frac{\mathrm{d}^2\boldsymbol{\xi}}{\mathrm{d}s^2}_m$ and $\frac{\mathrm{d}^3\boldsymbol{\xi}}{\mathrm{d}s^3}_m$ be the maximum magnitude of $\frac{\mathrm{d}^2\boldsymbol{\xi}}{\mathrm{d}s^2}$ and $\frac{\mathrm{d}^3\boldsymbol{\xi}}{\mathrm{d}s^3}$ over the desired trajectory. Then one can guarantee that $\ddot{\boldsymbol{\xi}}(s) \leq \mathbf{a}_m$ by setting $\ddot{s}_m$, the limit of $\ddot{s}$ over the manoeuvre, to:

$$\ddot{s}_m = \max(\min(\mathbf{c}), 0), \quad \mathbf{c} = \frac{\mathbf{a}_m - \frac{\mathrm{d}^2\boldsymbol{\xi}}{\mathrm{d}s^2}_m}{\left\|\frac{\mathrm{d}\boldsymbol{\xi}}{\mathrm{d}s}|_{s_0}\right\| + \frac{\mathrm{d}^2\boldsymbol{\xi}}{\mathrm{d}s^2}_m (s_1 - s_0)},$$

where division is elementwise in the above and following equation. Also, one can guarantee $\dddot{\boldsymbol{\xi}}(s) \leq \mathbf{j}_m$ by setting $\dddot{s}_m$, the limit of $\dddot{s}$ over the manoeuvre, to:

$$\dddot{s}_m = \max(\min(\mathbf{d}), 0), \quad \mathbf{d} = \frac{\mathbf{j}_m - \frac{\mathrm{d}^3\boldsymbol{\xi}}{\mathrm{d}s^3}_m - \frac{\mathrm{d}^2\boldsymbol{\xi}}{\mathrm{d}s^2}_m \ddot{s}_m}{\left\|\frac{\mathrm{d}\boldsymbol{\xi}}{\mathrm{d}s}|_{s_0}\right\| + \frac{\mathrm{d}^2\boldsymbol{\xi}}{\mathrm{d}s^2}_m (s_1 - s_0)}$$

Proof: Using the chain rule, we obtain the inequalities that must be satisfied by our choice of $\ddot{s}$ and $\dddot{s}$:

$$\ddot{\boldsymbol{\xi}}(s) = \frac{\mathrm{d}^2\boldsymbol{\xi}}{\mathrm{d}s^2}\dot{s}^2 + \frac{\mathrm{d}\boldsymbol{\xi}}{\mathrm{d}s}\ddot{s} \leq \mathbf{a}_m \tag{1a}$$

$$\dddot{\boldsymbol{\xi}}(s) = \frac{\mathrm{d}^3\boldsymbol{\xi}}{\mathrm{d}s^3}\dot{s}^3 + 3\frac{\mathrm{d}^2\boldsymbol{\xi}}{\mathrm{d}s^2}\ddot{s}\dot{s} + \frac{\mathrm{d}\boldsymbol{\xi}}{\mathrm{d}s}\dddot{s} \leq \mathbf{j}_m \tag{1b}$$

Note that the following are stricter criteria than (1a) and (1b):

$$\left\|\frac{\mathrm{d}^2\boldsymbol{\xi}}{\mathrm{d}s^2}\dot{s}^2\right\| + \left\|\frac{\mathrm{d}\boldsymbol{\xi}}{\mathrm{d}s}\ddot{s}\right\| \leq \mathbf{a}_m \tag{2a}$$

$$\left\|\frac{\mathrm{d}^3\boldsymbol{\xi}}{\mathrm{d}s^3}\dot{s}^3\right\| + \left\|3\frac{\mathrm{d}^2\boldsymbol{\xi}}{\mathrm{d}s^2}\ddot{s}\dot{s}\right\| + \left\|\frac{\mathrm{d}\boldsymbol{\xi}}{\mathrm{d}s}\dddot{s}\right\| \leq \mathbf{j}_m \tag{2b}$$

We rearrange (2a) and (2b) to obtain the requirements:

$$\|\ddot{s} \cdot \mathbf{1}\| \leq \frac{\mathbf{a}_m - \left\|\frac{\mathrm{d}^2\boldsymbol{\xi}}{\mathrm{d}s^2}\dot{s}^2\right\|}{\left\|\frac{\mathrm{d}\boldsymbol{\xi}}{\mathrm{d}s}\right\|} \tag{3a}$$

$$\|\dddot{s} \cdot \mathbf{1}\| \leq \frac{\mathbf{j}_m - \left\|\frac{\mathrm{d}^3\boldsymbol{\xi}}{\mathrm{d}s^3}\dot{s}^3\right\| - \left\|3\frac{\mathrm{d}^2\boldsymbol{\xi}}{\mathrm{d}s^2}\ddot{s}\dot{s}\right\|}{\left\|\frac{\mathrm{d}\boldsymbol{\xi}}{\mathrm{d}s}\right\|} \tag{3b}$$

The division in $(3a), (3b)$ and in the following equations (4) and (5) is elementwise and $\mathbf{1} \in \mathbb{R}^n$ is a vector of ones. We wish the inequalities of (3a) and (3b) to hold for $s \in [s_0, s_1]$.

Consider first (3a). Since the global trajectory is known in advance, we can substitute $\frac{\mathrm{d}^2\boldsymbol{\xi}}{\mathrm{d}s^2} = \frac{\mathrm{d}^2\boldsymbol{\xi}}{\mathrm{d}s^2}_m$, the maximum over the global trajectory[1]. Since $\dot{s} \in [0, 1]$, the numerator of the right-hand side attains a minimum at $\mathbf{a}_m - \frac{\mathrm{d}^2\boldsymbol{\xi}}{\mathrm{d}s^2}_m$. The global trajectory $\boldsymbol{\xi}$ is at least twice differentiable in s, so in the denominator we can use the Lagrange Remainder Theorem, which states $\frac{\mathrm{d}\boldsymbol{\xi}}{\mathrm{d}s} = \frac{\mathrm{d}\boldsymbol{\xi}}{\mathrm{d}s}|_{s_0} + \frac{\mathrm{d}^2\boldsymbol{\xi}}{\mathrm{d}s^2}|_{s^*}(s^* - s_0)$ for some $s^* \in [s_0, s_1]$. This is upper bounded by $\frac{\mathrm{d}\boldsymbol{\xi}}{\mathrm{d}s}|_{s_0} + \frac{\mathrm{d}^2\boldsymbol{\xi}}{\mathrm{d}s^2}_m(s_1 - s_0)$. A lower bound of the right-hand side of (3a) is therefore:

$$\frac{\mathbf{a}_m - \frac{\mathrm{d}^2\boldsymbol{\xi}}{\mathrm{d}s^2}_m}{\left\|\frac{\mathrm{d}\boldsymbol{\xi}}{\mathrm{d}s}|_{s_0}\right\| + \frac{\mathrm{d}^2\boldsymbol{\xi}}{\mathrm{d}s^2}_m(s_1 - s_0)} = \mathbf{c}, \tag{4}$$

and we take $\ddot{s}_m = \max(\min(\mathbf{c}), 0)$. Consider now (3b). By additionally using $\frac{\mathrm{d}^3\boldsymbol{\xi}}{\mathrm{d}s^3}_m$, the maximum over the global trajectory, and the result $\ddot{s}_m$, we lower-bound[2] the numerator of the right-hand side with $\mathbf{j}_m - \frac{\mathrm{d}^3\boldsymbol{\xi}}{\mathrm{d}s^3}_m - \frac{\mathrm{d}^2\boldsymbol{\xi}}{\mathrm{d}s^2}_m \ddot{s}_m$. The denominator is the same as (3a). The right-hand side of (3b) has a lower bound:

$$\frac{\mathbf{j}_m - \frac{\mathrm{d}^3\boldsymbol{\xi}}{\mathrm{d}s^3}_m - 3\frac{\mathrm{d}^2\boldsymbol{\xi}}{\mathrm{d}s^2}_m \ddot{s}_m}{\left\|\frac{\mathrm{d}\boldsymbol{\xi}}{\mathrm{d}s}|_{s_0}\right\| + \frac{\mathrm{d}^2\boldsymbol{\xi}}{\mathrm{d}s^2}_m(s_1 - s_0)} = \mathbf{d}, \tag{5}$$

and we choose $\dddot{s}_m = \max(\min(\mathbf{d}), 0)$. ■

As we do not know s_1 a priori, we use a conservative estimate which we expect to be longer than the actual end of the manoeuvre s_e; if it happens that $s_e > s_1$, we recalculate with an even more conservative estimate of s_1.

We can now plan a failsafe or recovery manoeuvre in s subject to the constraints that, at the end of the manoeuvre $\ddot{s}$ should be 0 and $\dot{s}$ should be 0 (failsafe) or 1 (recovery), and subject to the maximum acceleration and jerk $\|\ddot{s}\| \leq \ddot{s}_m$ and $\|\dddot{s}\| \leq \dddot{s}_m$. We refer the reader to [**?**] for details of how these manoeuvres are planned.

Having planned a short-term plan, we obtain t_e as the time at the end of the plan. This is also the time at which the robot is first stationary and hence safe according to the stationary criterion (Def. 2). It remains to calculate $t_{e,v_{\max}}$, the time after which all points on the robot are moving slower than $v_{\max}$ and the robot is considered safe according to the reduced speed criterion (Def. 3).

B. Obtaining $t_{e,v_{\max}}$

This subsection describes how to calculate $t_{e,v_{\max}}$ in line 8 of Algorithm 1, which is needed to verify the safety of a short-term plan regarding the reduced speed criterion. For brevity, we denote the time parameter at time t_a on the short-term plan, $\zeta(t_a)$, by s_a. This short-term plan starts at position s_{k+1} and time t_{k+1} while the robot is moving from a trajectory position s_k to s_{k+1} in the time interval $[t_k, t_{k+1}]$ (cf. Fig. 2). An intermediate result for $t_{e,v_{\max}}$ is $s_{e,v_{\max}}$, the position on the trajectory that the robot reaches at time $t_{e,v_{\max}}$. While the failsafe manoeuvre of the short-term plan (cf. III-A) is planned such that the robot stops as soon as possible, we cannot assume that the velocity in Cartesian space of all points on the robot is strictly decreasing during the manoeuvre. The approach we use here is therefore very general.

Previously we calculated t_e and the respective position on the trajectory s_e (which corresponds to the end position s_1 of the failsafe manoeuvre in III-A). Starting at s_e we backtrack on the trajectory until we either find a point where the robot is faster than the speed limit or we reach the start of the short-term plan s_{k+1}. If we find a point where the speed limit is violated, the position in the previous backtracking iteration with velocity below the limit is the sought $s_{e,v_{\max}}$. All points of the trajectory afterwards comply with the safety criterion. If backtracking terminates in the start position of the short-term plan at s_{k+1}, the whole short-term plan fulfills the criterion and $s_{e,v_{\max}}$ is just s_{k+1}. The reduced speed criterion then does not inhibit the movement of the robot at all. Algorithm 2 shows pseudocode for the described procedure.

Algorithm 2 Reduced speed criterion prediction (Line 8 of Algorithm 1)

```
1: function PREDICT(short-term plan starting at s_{k+1}, s_e)
2:     s_i ← s_e
3:     loop
4:         if s_i ≤ s_{k+1} then return s_{k+1}
5:         s_{i-1} ← simulate one step backwards
                     from s_i on short-term plan
6:         v_{i-1} ← calculate maximum Cartesian velocity
                     anywhere on robot, at point s_{i-1}
7:         if v_{i-1} > v_max then return s_i
8:         s_i ← s_{i-1}
```

Although the respective ISO norm only demands the TCP to be slower than $v_{\max}$, we present a novel, quick way to guarantee this for any point on the robot, as it is not certain where on the robot the human may collide with. Therefore we have to calculate the velocity of the fastest moving point on the robot to compare it to $v_{\max}$. We consider each robot segment as a capsule (cf. IV-A), calculate their maximum velocities separately and take the maximum of these.

To start off, we predict the movement of each capsule at the current position in the backtracking loop s_{i-1} and represent it as an screw axis movement. This screw axis movement consists of the axis (some offset vector $\mathbf{o} \in \mathbb{R}^3$ on it and a normalized direction $\mathbf{n} \in \mathbb{R}^3$) as well as a angular velocity $\omega \in \mathbb{R}$ around the axis and a translational velocity $v \in \mathbb{R}$ along it. With the joint coordinates $\mathbf{q} = \boldsymbol{\xi}(s_{i-1})$, the corresponding velocity $\dot{\mathbf{q}} = \dot{\boldsymbol{\xi}}(s_{i-1})$ and the Jacobian matrix $J(\mathbf{q})$ we calculate the segment's movement as twist $J(\mathbf{q})\,\dot{\mathbf{q}} = \binom{\boldsymbol{\omega}}{\mathbf{v}}$. The twist consists of the angular velocity $\boldsymbol{\omega} = \omega\mathbf{n}$ and a translational part $\mathbf{v} \in \mathbb{R}^3$. This can then be transformed into a screw axis movement by first

[1] this is feasible as long as $\frac{\mathrm{d}^2\boldsymbol{\xi}}{\mathrm{d}s^2}_m << \mathbf{a}_m$

[2] again, only feasible if $\frac{\mathrm{d}^3\boldsymbol{\xi}}{\mathrm{d}s^3}_m << \mathbf{j}_m$

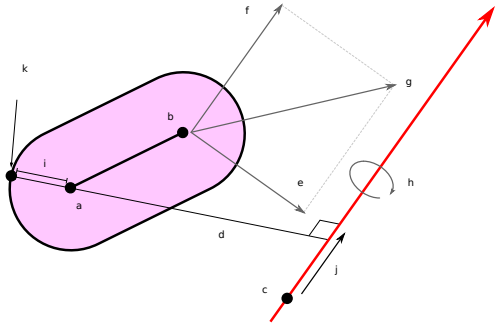

Fig. 3. Calculating the fastest moving point on the robot capsule. Shown in red is the screw axis. The fastest point is the furthest one from the axis. $\mathbf{p}_1$ and $\mathbf{p}_2$ are the defining points of the capsule. Other symbols are defined in the text.

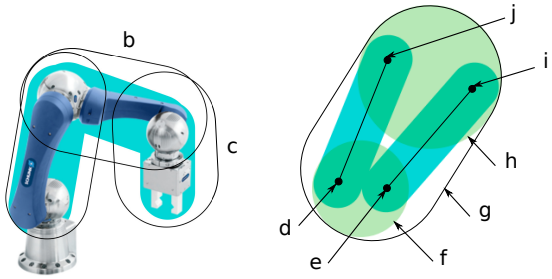

Fig. 4. (a) the enclosure of the occupancy of each link on the robot in capsules and (b) the generation of a capsule $\mathcal{C}_i$ to enclose the link occupancy during a short-term plan.

decomposing $\mathbf{v}$ into two parts, $\mathbf{v}_{\parallel} = (\mathbf{v}^T \cdot \mathbf{n})\mathbf{n}$ parallel to $\mathbf{n}$ and $\mathbf{v}_{\perp} = \mathbf{v} - \mathbf{v}_{\parallel}$ perpendicular to $\mathbf{n}$. The parallel part is the translational velocity of the screw axis $\mathbf{v}_{\parallel} = v\mathbf{n}$. The perpendicular part determines the offset $\mathbf{o}$ with the relation $\mathbf{v}_{\perp} = \mathbf{o} \times \boldsymbol{\omega}$. This allows us to calculate $\mathbf{o}$ by considering the cross product as multiplication with a skew-symmetric matrix $\mathbf{v}_{\perp} = [\boldsymbol{\omega}]_{\times}^{T}\mathbf{o}$ and picking one of the solutions of the system of linear equations. A special case is a pure translation ($\omega = 0$), but in this case all points of the capsule move with the same speed, i.e. $v\frac{m}{s}$.

Having calculated the screw axis movement we can now calculate the fastest moving point on the rigid body. The total velocity of a point $\mathbf{p}_x$ on the body is $\dot{\mathbf{p}}_x = v\mathbf{n} + \omega\mathbf{n} \times (\mathbf{p}_x - \mathbf{o})$, which also holds for the defining points of the capsule $\mathbf{p}_1$ and $\mathbf{p}_2$. Since the translational and the angular terms of the sum are orthogonal, the magnitude of $\dot{\mathbf{p}}_x$ can be calculated as:

$$\|\dot{\mathbf{p}}_x\| = \sqrt{\|\omega\mathbf{n} \times (\mathbf{p}_x - \mathbf{o})\|^2 + \|v\mathbf{n}\|^2}. \quad (6)$$

Finding the maximum of this for all points on a robot link boils down to finding the maximum of $\|\mathbf{n} \times (\mathbf{p}_x - \mathbf{o})\|$, which is the perpendicular distance between the screw axis and the point $\mathbf{p}_x$.

The furthest point of the line segment $\overline{\mathbf{p}_1\mathbf{p}_2}$ to the screw axis is which ever of the defining points is further. Since we consider the rigid link to be a capsule, the furthest point on the capsule is a point on the surface of it, that is one capsule radius away from the further defining point. The maximum perpendicular distance is then:

$$d_{\perp} := \max(\|\mathbf{n} \times (\mathbf{p}_1 - \mathbf{o})\|, \|\mathbf{n} \times (\mathbf{p}_2 - \mathbf{o})\|) + r, \quad (7)$$

where r is the radius of the capsule. Fig. 3 illustrates the situation. By plugging (7) back into (6) the maximum velocity is finally:

$$\max_x \|\dot{\mathbf{p}}_x\| = \sqrt{(|\omega| \cdot d_{\perp})^2 + |v|^2},$$

which we can now compare against $v_{\max}$.

After we obtained $s_{e,v_{\max}}$, $t_{e,v_{\max}}$ is then simple to derive by finding the difference to t_e. This difference is the time spent on the trajectory below the speed limit before a complete stop, or the number of iterations in the above algorithm multiplied by the time step size. To ensure deterministic latencies, we limit the number of iterations; if the limit is exceeded, the criterion is verified unsafe.

IV. PREDICTION OF OCCUPANCIES

This section deals with obtaining the occupancies of the robot (Sec. IV-A) and the human (Sec. IV-B) during the short-term plan. Previous work on obtaining the occupancy of a kinematic chain in a conservative way [**?**] generates an overapproximative sphere-swept volume (SSV) which encloses the robot's occupancy over a section of path. SSVs can be collision-checked by finding the minimum distance between them, e.g. using [**?**]. However, collision-checking of capsules is much faster and does not require iteration, e.g as in [**?**]. Since the human reachable occupancy can also be obtained as capsules (see Sec. IV-B), we present a novel method to obtain the robot occupancy as capsules, in order to simplify and speed up collision-checking.

A. Robot Occupancy

In this subsection we describe how to obtain the occupancy of the robot, $F(\mathbf{q}([t_{k+1}, t_e]))$ and $F(\mathbf{q}([t_{k+1}, t_{e,v_{\max}}]))$, in lines 7 and 9 of algorithm 1. All norms are Euclidean.

We first enclose each link of the robot in a capsule, as shown in Fig. 4a. The positions of the capsule end points are fixed on the robot and can be determined from forward kinematics. For each link i, we obtain a capsule $\mathcal{C}_i$ enclosing its occupancy over the short-term plan from t_{k+1} to t_e, see Fig. 4b. We do this by calculating two balls $\mathcal{S}_{i,1}$ and $\mathcal{S}_{i,2}$ which enclose the spatial path of the defining points $\mathbf{p}_{i,1}$ and $\mathbf{p}_{i,2}$ of the link capsule from t_{k+1} to t_e, enlarged by the radius of the link capsule. These balls are enclosed in a capsule $\mathcal{C}_i$. Since the defining points of the link capsule are in $\mathcal{C}_i$, by convexity, the entire link capsule is contained.

The endpoints of the link can be found from forward kinematics, i.e. $\mathbf{p}_{i,1}$ and $\mathbf{p}_{i,2}$ are functions of the joint positions, which in turn are a function of the path parameter s. For simplicity, we write: $\mathbf{p}_{i,1}(s_{k+1})$ and $\mathbf{p}_{i,1}(s_e)$ for the position of $\mathbf{p}_{i,1}$ at the path parameters at the start and end of the short-term plan; similarly for $\mathbf{p}_{i,2}$.

Of course, the path traced by $\mathbf{p}_{i,1}$ and $\mathbf{p}_{i,2}$ during the short-term plan cannot be assumed to be a straight line.

However, we approximate it by a straight line with some amount of deviation. Consider the point $\mathbf{p}_{i,1}$, without loss of generality. In the below lemma and proof we adapt Proposition 1 in [**?**] to bound the maximum deviation from the line segment between $\mathbf{p}_{i,1}(s_{k+1})$ and $\mathbf{p}_{i,1}(s_e)$, when the maximum value of $\|\frac{d^2\mathbf{p}_{i,1}}{ds^2}\|$ is known. This value can be found from the given trajectory and is denoted by α_i.

Lemma 2: Let $\mathbf{x}(\varsigma) \in \mathbb{R}^3$ be the position of a point at time parameter ς and suppose $\forall \varsigma : \|\ddot{\mathbf{x}}(\varsigma)\| \leq \alpha_i$. Let $\mathbf{x}_0$ and $\mathbf{x}_f$ be the known positions at time parameters $s = 0$ and $s = s_f$ and L be the line segment between them. For all $\varsigma \in [0, s_f]$, the distance from $\mathbf{x}(\varsigma)$ to L is no greater than $\alpha_i \frac{s_f^2}{8}$.

Proof: From the equations of motion we have:

$$\mathbf{x}(s) = \mathbf{x}_0 + \dot{\mathbf{x}}(0)s + \int_0^s \int_0^{\varsigma'} \ddot{\mathbf{x}}(\varsigma) d\varsigma d\varsigma' \tag{8}$$

Choosing $s = s^*$, we have the position of $\mathbf{x}(s^*)$ at an arbitrary time $s^* \in [0, s_f]$. The line segment L can be expressed as the set $L = \{\mathbf{x}_0 + \lambda(\mathbf{x}_f - \mathbf{x}_0) | \lambda \in [0, 1]\}$. Substituting $s = s_f$ in (8), we obtain an expression for $\mathbf{x}_f = \mathbf{x}(s_f)$, and choosing $\lambda = \frac{s^*}{s_f}$ in the expression for the set L, we see that the point $\mathbf{x}' = \mathbf{x}_0 + \dot{\mathbf{x}}(0)s^* + \frac{s^*}{s_f} \int_0^{s_f} \int_0^{\varsigma'} \ddot{\mathbf{x}}(\varsigma) d\varsigma d\varsigma'$ lies on L.

The difference $\mathbf{y}$ between $\mathbf{x}(s^*)$ and $\mathbf{x}'$ is:

$$\mathbf{y} = \frac{s^*}{s_f} \int_0^{s_f} \int_0^{\varsigma'} \ddot{\mathbf{x}}(\varsigma) d\varsigma d\varsigma' - \int_0^{s^*} \int_0^{\varsigma'} \ddot{\mathbf{x}}(\varsigma) d\varsigma d\varsigma' \tag{9}$$

Observing that $\int_0^s \int_0^{\varsigma'} \ddot{\mathbf{x}}(\varsigma) d\varsigma d\varsigma' = \int_0^{s^*} \int_0^{\varsigma'} \ddot{\mathbf{x}}(\varsigma) d\varsigma d\varsigma' + \int_{s^*}^{s_f} \int_0^{\varsigma'} \ddot{\mathbf{x}}(\varsigma) d\varsigma d\varsigma'$, we obtain:

$$\mathbf{y} = \frac{s^*}{s_f} \underbrace{\int_{s^*}^{s_f} \int_0^{\varsigma'} \ddot{\mathbf{x}}(\varsigma) d\varsigma d\varsigma'}_{b} - \frac{s_f - s^*}{s_f} \underbrace{\int_0^{s^*} \int_0^{\varsigma'} \ddot{\mathbf{x}}(\varsigma) d\varsigma d\varsigma'}_{c} \tag{10}$$

We observe that:

$$\begin{aligned} b &= \int_{s^*}^{s_f} \int_0^{s^*} \ddot{\mathbf{x}}(\varsigma) d\varsigma d\varsigma' + \int_{s^*}^{s_f} \int_{s^*}^{\varsigma'} \ddot{\mathbf{x}}(\varsigma) d\varsigma d\varsigma' \\ &= (s_f - s^*) \int_0^{s^*} \ddot{\mathbf{x}}(\varsigma) d\varsigma + \int_{s^*}^{s_f} \int_{s^*}^{\varsigma'} \ddot{\mathbf{x}}(\varsigma) d\varsigma d\varsigma' \\ c &= \int_0^{s^*} \int_0^{s^*} \ddot{\mathbf{x}}(\varsigma) d\varsigma d\varsigma' - \int_0^{s_f} \int_{\varsigma'}^{s^*} \ddot{\mathbf{x}}(\varsigma) d\varsigma d\varsigma' \\ &= s^* \int_0^{s^*} \ddot{\mathbf{x}}(\varsigma) d\varsigma - \int_0^{s_f} \int_{\varsigma'}^{s^*} \ddot{\mathbf{x}}(\varsigma) d\varsigma d\varsigma' \end{aligned} \tag{11}$$

Substituting (11) back into (10) we obtain:

$$\mathbf{y} = \frac{s^*}{s_f} \underbrace{\int_{s^*}^{s_f} \int_{s^*}^{\varsigma'} \ddot{\mathbf{x}}(\varsigma) d\varsigma d\varsigma'}_{d} + \frac{s_f - s^*}{s} \underbrace{\int_0^{s^*} \int_{\varsigma'}^{s^*} \ddot{\mathbf{x}}(\varsigma) d\varsigma d\varsigma'}_{e} \tag{12}$$

We observe that d is the double-integral of acceleration from $\varsigma = s^*$ until s_f, which is upper-bounded by $\frac{\alpha_i (s_f - s^*)^2}{2}$. The expression e, which can be rewritten:

$$\int_0^{s^*} \int_{\varsigma'}^{s^*} \ddot{\mathbf{x}}(\varsigma) d\varsigma d\varsigma' = \int_0^{s^*} -\int_{s^*}^{\varsigma'} \ddot{\mathbf{x}}(\varsigma) d\varsigma d\varsigma' = \int_{s^*}^{0} \int_{s^*}^{\varsigma'} \ddot{\mathbf{x}}(\varsigma) d\varsigma d\varsigma', \tag{13}$$

is the acceleration double-integrated "backward" from $\varsigma = s^*$ until 0. This is upper-bounded by $\frac{\alpha_i \varsigma^{*2}}{2}$. Both s^* and $s_f - s^*$ are positive, so:

$$\|\mathbf{y}\| \leq \frac{s^*}{s_f} \frac{\alpha_i (s_f - s^*)^2}{2} + \frac{s_f - s^*}{s_f} \frac{\alpha_i (s^*)^2}{2}. \tag{14}$$

The right hand side attains a maximum of $\alpha_i \frac{s_f^2}{8}$ at $s^* = \frac{s_f}{2}$ (this can be seen by setting to zero the differential of the right hand side with respect to s^*). Hence the distance between $\mathbf{x}(s^*)$ and L is no more than $\alpha_i \frac{s_f^2}{8}$. ∎

We now show how to calculate $\mathcal{C}_i$. We denote the closed ball centered at $\mathbf{z}$ with radius r as:

$$B(\mathbf{p}; r) = \{\mathbf{x} \mid \|\mathbf{x} - \mathbf{z}\| \leq r\}.$$

The operators $\text{CE}(b_1, b_2)$ and $\text{BE}(b_1, b_2)$ standing for *capsule enclosing* and *ball enclosing*. These output the capsule or the ball enclosing the balls b_1 and b_2. Their detailed description is found in [**?**]. Formally, the occupancy of the link i with radius $r_{l,i}$ is:

$$\begin{aligned} r_{z,i} &= \alpha_i \frac{(s_e - s_{k+1})^2}{8} + r_{l,i} \\ \mathcal{S}_{i,1} &= \text{BE}(B(\mathbf{p}_{i,1}(s_{k+1}); r_{z,i}) \ , \ B(\mathbf{p}_{i,1}(s_e); r_{z,i})) \\ \mathcal{S}_{i,2} &= \text{BE}(B(\mathbf{p}_{i,2}(s_{k+1}); r_{z,i}) \ , \ B(\mathbf{p}_{i,2}(s_e); r_{z,i})) \\ \mathcal{C}_i &= \text{CE}(\mathcal{S}_{i,1}, \mathcal{S}_{i,2}) \end{aligned}$$

By the property of convexity, since the link endpoints are enclosed, all points on the link i are inside capsule $\mathcal{C}_i$.

B. Prediction of Human Occupancy

This subsection deals with the prediction of the human reachable occupancies Γ_r and Γ_{ISO} from lines 11 and 12 from Algorithm 1. The human arm is a nonlinear hybrid dynamical system, and its exact reachable set is impossible to calculate [**?**]. To still be able to formally verify the safety of the robot movement, one needs a tight, overapproximative prediction of human motion, meaning a prediction which includes all possible reachable states of the human, while excluding as many unreachable states as possible.

In [**?**] we present an approach to calculate 3 overapproximative predictions, which individually account for acceleration, velocity and position limits (since a prediction which would account for all limits simultaneously requires a hybrid model of human motion, and reachability analysis of hybrid systems is time consuming). Since each of these 3 predictions is a superset of the exact reachable occupancy of the human Γ_r, then if any of them is verified safe against the robot's short-term plan, then the exact reachable occupancy Γ_r is also safe. We therefore perform verification on each of these occupancies and the reduced speed safety criterion is verified, if any of these occupancies are verified safe. Due to space

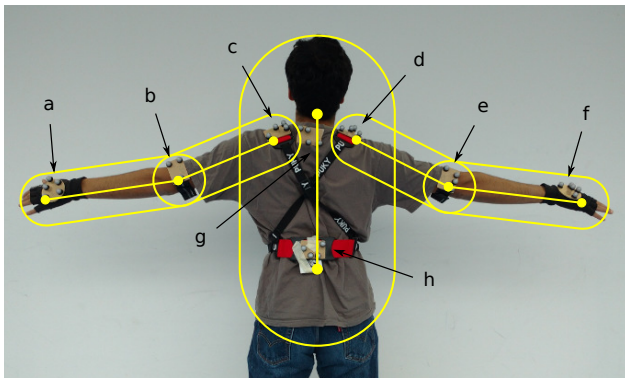

Fig. 5. Human reachable occupancy with the labelled marker clusters

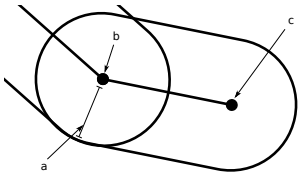

Fig. 6. Human reachable occupancy Γ_{ISO}

limitations, we do not detail the models here; the reader is referred to [**?**].

To calculate Γ_{ISO}, we assume a maximum speed of the human arm v_{human}. We enclose the human in capsules, as shown in Fig. 5. These capsules' radii are specified such that they enclose all body parts — for the torso and head capsule, the radius is $0.3m$ and for the upper arm and forearm capsules, the radii are both $0.1m$. To calculate $\Gamma_{ISO}([0,t])$, we simply augment the radii by $v_{human} \cdot t$, as shown in Fig. 6.

V. EXPERIMENTS AND DISCUSSION

We implemented and tested the described trajectory planner on a 6-DOF Schunk LWA 4P robot arm with a two-finger gripper mounted to the end effector. The robot is controlled by a Speedgoat SN2820 real-time target machine running a Simulink R2015b real-time kernel; robot and target machine communicate via CAN bus. Due to the Simulink kernel the robot control can be programmed in a Simulink model on a host computer connected to the target machine, and executed remotely. We use the robot arm in interpolated position mode and the controller runs at 500Hz, i.e. a position command is sent every $2ms$, and low-level control is performed by the motors in the joints.

A. Human Tracking

To track the upper body of the human we use six Vicon Vero 1.3 motion capture cameras that track reflecting markers at 100Hz. The camera data are aggregated by the Vicon Tracker 3 software, which tracks the position and orientation of coordinate systems attached to rigid clusters of markers. We made eight rigid marker clusters, designed to be uniquely identifiable by Tracker, and affixed them to the body as shown in Fig. 5. A separate program using the Vicon

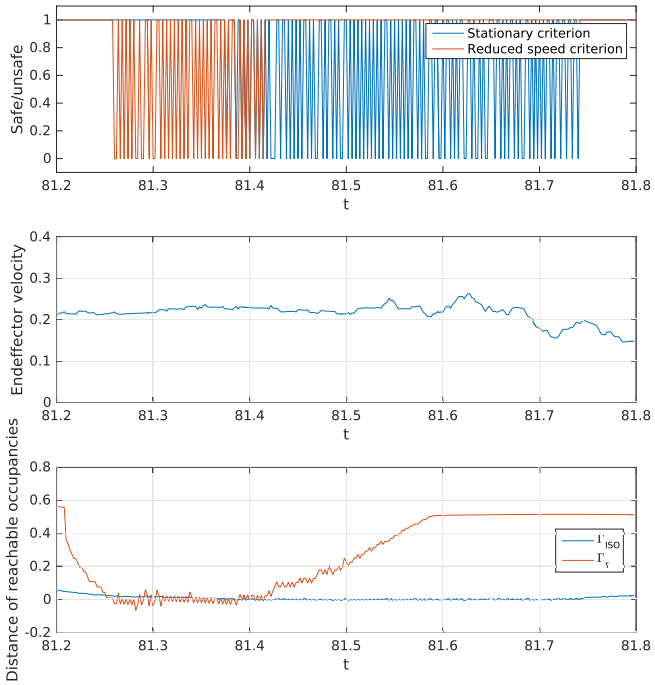

Fig. 7. Experimental results. Upper graph shows whether short-term plan could be verified safe, in the middle the end effector velocity is depicted, and the bottom graph shows the minimum distance between the reachable occupancies of human and robot.

DataStream SDK extracts the positions of the marker groups relative to the robot's base from Tracker 3 and send them to the target machine via UDP while it is executing the robot's routine. Although the Tracker 3 cannot run on a real-time computer, the latency claimed by the manufacturers is very low at around $2.8ms$[3]. We took the latency from camera to real-time target machine as $5ms$.

B. Comparison

In our experiment the robot executed a predefined routine that simulated a real world scenario. The human then entered the robot's workspace and performed various movements with varying velocity to trigger the failsafe behaviour of the trajectory planner. The setup is shown in Fig. 8. To test the effectiveness and correctness of our approach we performed this trial multiple times with different values for $v_{\max}$ and compared it with approaches from previous work.

In previous approaches our trajectory planner only used the stationary criterion to guarantee safety [**?**]. We tested this both with Γ_{ISO} as well as with Γ_r to demonstrate the shortcomings compared to the approach presented in this paper. When using Γ_r with the stationary criterion, the robot proved rather difficult to work with due to the high possible velocity and acceleration of the human. The human had to stand 2 to 3 meters away not to inhibit the robot. On the other hand with Γ_{ISO} the trajectory planner provided mostly sufficient safety without being too restrictive, as long as the human moved slower than $1.6\frac{m}{s}$. But as soon as the

[3] `vicon.com/products/software/tracker`, retrieved 21.2.2017

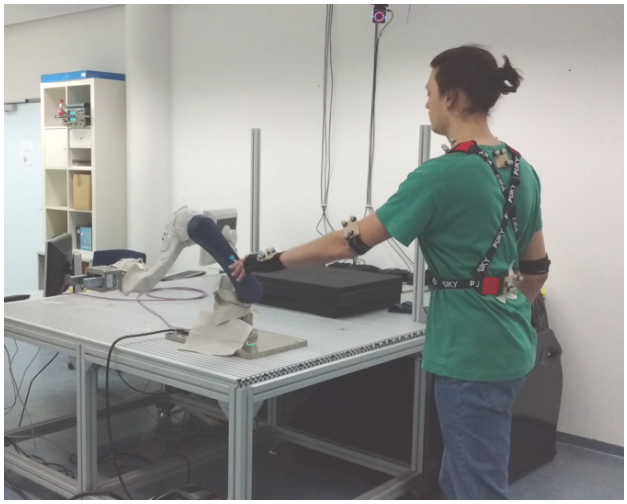

Fig. 8. Picture of the experimental trial. A video can be found at `www6.in.tum.de/pub/Main/Pereira/CDC.mp4`

human moved faster than that, situations could occur where it is not entirely obvious whether the robot truly stopped completely before the human was able to reach it. Finally, we tested the trajectory planner with the intended setting of $0.25\frac{m}{s}$. With this the robot mostly behaved according to the stationary criterion, but in cases where it was faster than $v_{\max}$ and the human was standing respectively close had to slow down to abide to the reduced speed criterion. We also used $v_{\max} = 0.1\frac{m}{s}$ for comparison.

Fig. 7 shows a short section of the data recording of a test with $v_{\max} = 0.1\frac{m}{s}$. The upper graph shows the boolean values whether the short-term plans fulfil the two criteria, blue for the stationary criterion and orange for the reduced speed criterion. They jump quickly between *safe* and *unsafe* as this slows the robot down just enough to separate reachable occupancies. The middle graph shows the velocity of the robot's end effector based on joint angle measurements. During the time where the robot slowed down for the reduced speed criterion one can see that the robot maintains a speed just above $v_{\max}$. In case the human would have come any closer, the robot could have easily decelerated below $0.1\frac{m}{s}$. The bottom graph shows the minimum distances between the reachable occupancies. Around time 81.3, the distance between Γ_r and the robot occupancy is around zero, meaning the reduced-speed criterion is often verified unsafe (top graph). Around time 81.5, the distance between Γ_{ISO} and the robot occupancy is around zero, meaning the stationary criterion is often verified unsafe.

VI. CONCLUSIONS

In this paper we present a method to formally plan and verify a safe robot trajectory online, without unnecessarily restricting robot motion. We guarantee both that, if the human movement as assumed by ISO standards, the robot can stop completely before impact, and that if not, the robot velocity will be under a pre-defined value at the point of impact.

We test our approach in an experimental setup. We show that guaranteeing the robot can stop before impact while accounting for all human motion hinders robot operation, whereas only accounting for human movement assumed in ISO standards does not guarantee safety when the human moves faster than the assumptions. Our novel, dual-criterion online formal verification allows the robot to move efficiently while still guaranteeing safety, paving the way for formally safe robots which are also efficient workers.