2014 IEEE/RSJ International Conference on
Intelligent Robots and Systems (IROS 2014)
September 14-18, 2014, Chicago, IL, USA

Modeling and Controller Design of Cooperative Robots in Workspace Sharing Human-Robot Assembly Teams

Changliu Liu and Masayoshi Tomizuka, *Fellow, IEEE*

***Abstract*— Human workers and robots are two major workforces in modern factories. For safety reasons, they are separated, which limits the productive potentials of both parties. It is promising if we can combine human's flexibility and robot's productivity in manufacturing. This paper investigates the modeling and controller design method of workspace sharing human-robot assembly teams and adopts a two-layer interaction model between the human and the robot. In theoretical analysis, enforcing invariance in a safe set guarantees safety. In implementation, an integrated method concerning online learning of closed loop human behavior and receding horizon control in the safe set is proposed. Simulation results in a 2D setup confirm the safety and efficiency of the algorithm.**

I. Introduction

In modern factories, human workers and robots are two major workforces. For safety reasons, the two are separated by metal cages. Robots focus on repetitive work, such as machining, while human workers focus on delicate work, such as assembly. As a result, in many car factories, final assembly is still done extensively and expensively by hand. People have been curious for a long time about the possibility of bringing human workers and robots together in production lines.

The potentials of human-robot teams can be huge and can solve a lot of problems in current manufacturing. One obvious advantage is that human-robot teams can be introduced in flexible assembly lines to take the advantage of human's flexibility and robot's productivity. With human workers doing flexible work and robots doing assistive work that is repetitive or requiring high precision, the overall productivity can be boosted.

Recently, several manufacturers including BMW and Volkswagen launched cooperative robots in their final assembly lines [1], [2]. In the BMW's factory in Spartanburg, South Carolina, the robot co-operates with a human worker to insulate and water-seal vehicle doors. The robot's job is to glue down the materials held by the human worker with his agile fingers. Before the introduction of these robots, workers must be rotated off this uncomfortable task after one hour or two to prevent elbow strain. Meanwhile, the introduction of cooperative robots also poses new challenges to the controller design of the robots, since the robots should be made safe to human workers and also efficient in finishing their own tasks [3], [4].

In human-robot cooperated manufacturing, robots interact with humans in two ways [4]: workspace sharing and time sharing. In a workspace sharing system, humans and robots perform separate tasks. They can do their jobs on any time horizon. In a time sharing system, humans and robots jointly perform one task, as discussed in the case of BMW's cooperative robot. In this paper, the workspace sharing system will be studied as it forms the basic interaction type in cooperative robotics. One possible scenario is shown in Fig.1. During the final assembly of a car, the robot (with 2 degrees of freedom) picks and feeds large assembly parts for the human (we only model one human worker in the analysis, for simplicity), while the human picks the tools and delicate parts to do the assembly. In this scenario, the robot needs to approach its goal without interfering with the human. This problem will be formalized in section III.

Fig. 1: Human robot cooperation on car assembly [4]

This paper focuses on the modeling and controller design method of cooperative robots working in a human-involved environment. The biggest challenge comes from human factors [3]. We can model the human either as a cost minimizer [5] or as a stochastic agent [6]. However, the human's utility function or the probability distribution of his behavior is hard to obtain for sophisticated problems. In this paper, we just assume that human's motion control law can be linearized locally, which will be discussed in section IV. Another challenge is the robot motion-planning in a dynamic and uncertain environment. Usually, the motion-planning problem is formulated as an optimization problem, having the safety in the hard constraints [7]. The author of [8] proposed a safety-oriented method to solve the motion-planning problem by introducing a supervisory loop to locally modify the reference trajectory once the safety constraint is violated. The system is guaranteed to go back to the safe region in a finite time. In this paper, the optimization approach and the safety-oriented approach will be combined to guarantee both safety and efficiency.

The remainder of this paper is organized as follows: In section II, a workspace sharing cooperative robotics problem is proposed. In section III, theoretical analysis concerning

C.Liu and M.Tomizuka are with University of California, Berkeley, CA 94720 USA (e-mail: changliuliu@berkeley.edu, tomizuka@me.berkeley.edu).

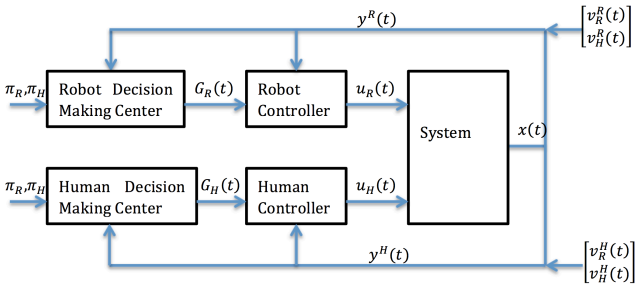


Fig. 2: System block diagram

the safe set is presented. In section IV, an online learning algorithm of human behavior, and a receding horizon control algorithm constrained in the safe set are proposed. In section V, the simulation results are presented.

II. Workspace Sharing Interaction Model

A. Interaction Model

There are two agents in the system, a robot (denoted by R) and a human (denoted by H). We consider a two-layer interaction model as shown in Fig.2. Both the robot and the human listen to a manager on a set of goals they need to approach, i.e. π_R for the robot and π_H for the human. Assume that the two agents share the knowledge of π_R and π_H. In the decision making center, the robot and the human decide which goal to go at current time (i.e. $G_R(t)$, $G_H(t)$) based on their observations of each other. Then the robot generates the control signal $u_R(t)$ to approach its goal, while the human generates the control signal $u_H(t)$. The two-layer control offers the robot more freedom to pursue safety and efficiency during the interaction with the human.[1]

B. System Model

Both the robot and the human are dynamic systems. For simplicity, we just model their kinematics. In the scenario shown in Fig.1, we take eight states to describe the system dynamics. Let the state x_1 be the robot's x-position, x_2 the robot's x-velocity, x_3 the robot's y-position, x_4 the robot's y-velocity, x_5 the human's x-position, x_6 the human's x-velocity, x_7 the human's y-position, and x_8 the human's y-velocity. The robot and the human control their motion through the accelerations, $u_R, u_H \in \mathbb{R}^2$ respectively. Thus, we have the following state equations for the robot and the human:

$$\dot{x}_R(t) = A_R x_R(t) + B_R u_R(t) \quad (1)$$
$$\dot{x}_H(t) = A_H x_H(t) + B_H u_H(t) \quad (2)$$

where $x_R = [x_1, x_2, x_3, x_4]^T$, $x_H = [x_5, x_6, x_7, x_8]^T$,

$$A_R = A_H = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \end{bmatrix}, B_R = B_H = \begin{bmatrix} 0 & 0 \\ 1 & 0 \\ 0 & 0 \\ 0 & 1 \end{bmatrix}$$

[1]There are time differences between the human and the robot in decision making and control. However, in our approach, the robot is basically playing reactive strategy to the human's motion as discussed in section III and IV. The time difference is not a significant issue from the robot's point of view, if the robot can update his belief of the human's dynamics fast enough.

Let $x = [x_R, x_H]$. The system's dynamic equation is:

$$\dot{x}(t) = \begin{bmatrix} A_R & 0 \\ 0 & A_H \end{bmatrix} x(t) + \begin{bmatrix} B_R \\ 0 \end{bmatrix} u_R(t) + \begin{bmatrix} 0 \\ B_H \end{bmatrix} u_H(t) \quad (3)$$

This is a two-agent system. The two agents share the same interest in that they both want to stay in the safe region (i.e. do not collide with each other). However, their goals are not identical. So there are conflicts of interests that need to be solved.

C. Observation Model

In the present model, we assume no communication and the only information the robot and the human can get is the measured system states. Suppose the robot has the measurements as follows (the subscript represents the agent being measured, the superscript represents the agent taking the measurement):

$$y_R^R(t) = x_R(t) + v_R^R(t) \quad (4)$$
$$y_H^R(t) = x_H(t) + v_H^R(t) \quad (5)$$

Symmetrically, the human has the measurements:

$$y_R^H(t) = x_R(t) + v_R^H(t) \quad (6)$$
$$y_H^H(t) = x_H(t) + v_H^H(t) \quad (7)$$

where $v_R^H, v_R^R, v_H^H, v_H^R$ are the measurement noises assumed to be zero-mean, Gaussian, white and independent to each other. The measurements are to be used by the human and the robot in their decision making and control process.

D. System Requirements

In order to be profitable for manufacturers and safe for human workers, the system needs to satisfy several requirements. In this paper, we consider the following two: (1) The system should always be safe; (2) The motion of the cooperative robot should be efficient. Mathematically, the first condition states that the control effort should make the safe set invariant to the system. The second condition states that the robot needs to do optimal control in that safe set.

III. Theoretical Analysis

A. Geometry

Since the safety constraint is usually a geometric constraint, we consider the geometry first and define several important quantities as follows. To start with, define the relative state as

$$dx(t) = x_R(t) - x_H(t) \quad (8)$$

The relative distance vector is denoted by $\mathbf{d_{rel}}(t)$ and the relative velocity vector by $\mathbf{v_{rel}}(t)$. Then the normed distance, the dot product between the relative distance and the relative velocity and the normed velocity are

$$||\mathbf{d_{rel}}(t)||^2 = dx(t)^T P_1 dx(t) \quad (9)$$
$$\mathbf{d_{rel}}(t) \cdot \mathbf{v_{rel}}(t) = dx(t)^T P_2 dx(t) \quad (10)$$
$$||\mathbf{v_{rel}}(t)||^2 = dx(t)^T P_3 dx(t) \quad (11)$$

where[2]

$$P_1 = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}, P_2 = \begin{bmatrix} 0 & 0.5 & 0 & 0 \\ 0.5 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0.5 \\ 0 & 0 & 0.5 & 0 \end{bmatrix}$$

$$P_3 = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Define the scalar distance as $d(t) = ||\mathbf{d_{rel}}(t)||$. The first and the second order derivatives of the scalar distance are

$$\dot{d}(t) = \frac{\mathbf{d_{rel}}(t)\cdot\mathbf{v_{rel}}(t)}{d(t)} \tag{12}$$

$$\ddot{d}(t) = \frac{||\mathbf{v_{rel}}(t)||^2 + \mathbf{d_{rel}}(t)\cdot\mathbf{a_{rel}}(t)}{d(t)} - \frac{(\mathbf{d_{rel}}(t)\cdot\mathbf{v_{rel}}(t))^2}{d^3(t)} \tag{13}$$

where $\mathbf{a_{rel}}(t) = u_R(t) - u_H(t)$. Thus the relative degree from $d(t)$ to the inputs $u_R(t), u_H(t)$ in the lie derivative sense is two.

B. Geometric Invariance for the Safe Set

Suppose the state space of the system can be divided into two sets, the safe set and the unsafe set. The safety requirement is to make the safe set invariant.

We describe the safe set by introducing a safety index $\phi : X \to \mathbb{R}$, which is a functional on the state space. The system is considered safe if the safety index is non-positive, i.e. the set $X_\phi = \{x : \phi \leq 0\}$ is safe. The control strategy that enforces $\dot{\phi}(t) < 0$ when $\phi(t) \geq 0$ will make the set X_ϕ asymptotically stable and make it invariant if $\phi(t_0) < 0$, where t_0 is the start time. The proof is straightforward using Lyapunov theorem [8].

Now we have the tool to enforce invariance in the safe set. The problem left is how to define the safety index. Intuitively, one candidate for the safety index ϕ is

$$\phi_0(t) = d_{min} - d(t) \tag{14}$$

where $d_{min} > 0$ is the minimum acceptable distance between the robot and the human. However, since the relative degree from $d(t)$ to $u_R(t)$ is two, it is hard to design controllers in this set. We consider a new safety index with a velocity term, and take quadratic form of $d(t)$ to make small relative distance even more undesirable.

$$\phi(t) = d_{min}^2 - d^2(t) - k_\phi \dot{d}(t) \tag{15}$$

where $k_\phi > 0$ is a constant to adjust the relative emphasis between the velocity term and the distance term. Then we have the following proposition. (The proof is shown in Appendix.)

Proposition 1. The set defined by $\{d : \phi(t) \leq 0, \forall t > t_0, d(t_0) > d_{min}\}$ is guaranteed to be a subset of the set $\{d : d(t) \geq d_{min}, \forall t > t_0\}$.

So the set of safe control is

$$U_R(t) = \{u_R(t) : \dot{\phi}(t) < 0, \ when\ \phi(t) \geq 0\} \tag{16}$$

[2]Notice that $||\mathbf{d_{rel}}(t)||^2 = (x_1 - x_5)^2 + (x_3 - x_7)^2$, $\mathbf{d_{rel}}(t)\cdot\mathbf{v_{rel}}(t) = (x_1 - x_5)(x_2 - x_6) + (x_3 - x_7)(x_4 - x_8)$, $||\mathbf{v_{rel}}(t)||^2 = (x_2 - x_6)^2 + (x_4 - x_8)^2$.

This set is dependent on $\mathbf{a_{rel}}(t)$, thus $u_H(t)$, since $\dot{\phi}(t)$ has the following expression:

$$\dot{\phi}(t) = -2d(t)\dot{d}(t) - k_\phi \ddot{d}(t) \tag{17}$$

In order to get a safe action set, the robot needs to predict the human's future states, which will be discussed in IV-B.

IV. Implementation

A. Discretization of the Model

In the real world, information is sampled and control inputs are decided at each time step. Thus the continuous time model needs to be discretized. Taking the sampling interval to be T_s, we have the following discrete time state equations:

$$x_R(k+1) = A_R^d x_R(k) + B_R^d u_R(k) \tag{18}$$

$$x_H(k+1) = A_H^d x_H(k) + B_H^d u_H(k) \tag{19}$$

where

$$A_R^d = A_H^d = \begin{bmatrix} 1 & T_s & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & T_s \\ 0 & 0 & 0 & 1 \end{bmatrix}, B_R^d = B_H^d = \begin{bmatrix} \frac{T_s^2}{2} & 0 \\ T_s & 0 \\ 0 & \frac{T_s^2}{2} \\ 0 & T_s \end{bmatrix}$$

B. Online Learning Algorithm of the Human Behavior

We assume that human's motion control law can be linearly approximated locally. When observing the human's behavior, the robot can run expectation maximization (EM) algorithm to get the human's closed loop dynamics. Suppose the human's motion control law is $u_H(k) = \tilde{f}(y_H^H(k), y_R^H(k), G_H(k))$, which can be locally linearized as

$$u_H(k) = K_1 y_H^H(k) + K_2 y_R^H(k) + K_3 G_H(k) \tag{20}$$

When the robot is learning the human behavior, it can only estimate $y_R^H(k)$ and $G_H(k)$ in (20) by its own measurement $y_R^R(k)$ and inference $G_H^R(k)$ (derivation of $G_H^R(k)$ will be discussed in IV-D.1). Define $u_H^{cl}(k) = [y_R^R(k)^T, G_H^R(k)^T]^T$. Substituting (20) into (19) and considering (4) (6) (7), we can have the closed loop dynamic equation of the human[3]:

$$x_H(k+1) = A_H^* x_H(k) + B_H^* u_H^{cl}(k) + w_H(k) \tag{21}$$

Equation (21) and (5) define a linear Gaussian system with unknown parameters. In order to estimate the states and learn the parameters, the robot can run online EM algorithm [9]. Define $\hat{x}_H(k|k-1)$ to be the *a priori* estimate of $x_H(k)$ and $\hat{x}_H(k|k)$ the *a posteriori* estimate of $x_H(k)$. Let the parameter matrix be $C(k) = [\hat{A}_H^*(k), \hat{B}_H^*(k)]$, and the data vector be $\psi(k) = [\hat{x}_H^T(k|k), u_H^{cl}(k)^T]^T$. The online EM algorithm can be set-up as shown below.

Expectation step:

$$\hat{x}_H(k+1|k) = C(k)\psi(k) \tag{22}$$

$$\hat{x}_H(k+1|k+1) = (1-\alpha)\hat{x}_H(k+1|k) + \alpha y_H^R(k+1) \tag{23}$$

[3]In (21), $A_H^* = A_H^d + B_H^d K_1$, $B_H^* = [B_H^d K_2, B_H^d K_3]$. $w_H(k) = B_H^d K_1 v_H^H(k) + B_H^d K_2 (v_R^H(k) - v_R^R(k))$ is Gaussian.

Maximization step:

$$C(k+1) = C(k) + \epsilon(k+1)\psi^T(k)F(k+1) \quad (24)$$

$$\epsilon(k+1) = \hat{x}_H(k+1|k+1) - \hat{x}_H(k+1|k) \quad (25)$$

$$F(k+1) = \frac{1}{\lambda}[F(k) - \frac{F(k)\psi(k)\psi^T(k)F(k)}{\lambda + \psi^T(k)F(k)\psi(k)}] \quad (26)$$

In the E-step, (22) is the dynamic update, and (23) is the measurement update. α is the filter gain. It represents how much we trust the current measurement. Usually, α is set to be the Kalman filter gain. However, since our dynamic system is time-varying, we use constant gain to ensure that we always incorporate the information from the measurements. In the M-step, (24) is the parameter update which depends on the estimation error ϵ in (25) and the learning gain F in (26). λ is the forgetting factor. Since the human behavior is time-varying, we use the forgetting factor to make the parameter adaptation algorithm active all the time. Sometimes, the human will adapt to the robot's motion. Sometimes, when the human is occupied, he may not pay attention to the robot's motion. The only assumption we make is that human does not change very fast. For a certain period of time, his motion is consistent so that the control law can be approximated by linear functions and learned online.

C. Set of Safe Control

In the discrete time, the set of safe control is different from the one in the continuous case. Instead of constraining the derivative of the safety index, we constrain the value of the safety index in the next time step (which is a stronger constraint), i.e.

$$U_R(k) = \{u_R(k) : \phi(k+1) \leq 0\} \quad (27)$$

where

$$\phi(k+1) = D^* - dx(k+1)^T P_1 dx(k+1) - \frac{dx(k+1)^T P_2 dx(k+1)}{\sqrt{dx(k+1)^T P_1 dx(k+1)}} \quad (28)$$

Equation (28) is the discrete time version of (15), except that we take $k_\phi = 1$ and $D^* > d_{min}^2$ to account for the uncertainties from the state measurement and prediction. All calculations in (27) depend on the relative state. At time step k, the prediction of $dx(k+1)$ is

$$\hat{dx}(k+1|k) = I(k) + B_R^d u_R(k) \quad (29)$$

where $I(k) = A_R^d \hat{x}_R(k|k) - \hat{x}_H(k+1|k)$ and $\hat{x}_R(k|k)$ is the estimate of $x_R(k)$ using Kalman filter. Since the input acceleration only has limited effect on the position vector, we approximate $B_R^d u_R(k)$ by $B_R u_R(k)$. By (29), the inequality in (27) becomes:

$$2I^T(k)P_2 B_R u_R(k) \geq D^* \sqrt{I^T(k)P_1 I(k)} - \left(I^T(k)P_1 I(k)\right)^{\frac{3}{2}} - I^T(k)P_2 I(k) \quad (30)$$

So the set of safe control $U_R(k)$ is a half plane.

D. Algorithm in the Decision-Making Level

1) Inference on the Human's Goal: Though the robot knows π_H, it does not know the exact human goal at the current time step. Assume the human is always facing towards his goal. Then the likelihood of $G \in \pi_H$ being the human's current goal is inversely proportional to the time needed to reach G given the human's velocity component that is pointing towards G ((9) divided by (10)). So the most likely goal is:

$$G_H^R(k) = \arg\min_{G \in \pi_H} \{t = -\frac{(y_H^R(k)-G)^T P_1 (y_H^R(k)-G)}{(y_H^R(k)-G)^T P_2 (y_H^R(k)-G)} : t > 0\} \quad (31)$$

2) Goal Generation for the Robot: We generate goals for the robot according to the distance to the goal, the robot's current velocity and the distance to the human. Equation (32) is the goal generation equation. The first term in (32) is the absolute value of the time needed to travel to the closest point to the goal given the robot's current velocity ((10) divided by (11)). The time will become negative if the robot is going away from the goal. In order to take into account of possible goal point overshoot when the robot is trying to avoid the human, we consider the absolute value of the calculated time (i.e. it is still worthwhile to go back if the overshoot is small). The second term evaluates the distance from the goal point to the human in a preview window H_r. If the human is wandering around one of the goal points of the robot, the robot will be less likely to go to that goal point. k_H is a constant to adjust the ratio between the two terms.

$$G_R(k) = \arg\min_{G \in \pi_R} J_G = |\frac{(y_H^R(k)-G)^T P_2 (y_H^R(k)-G)}{(y_H^R(k)-G)^T P_3 (y_H^R(k)-G)}| + k_H e^{-\sum_{i=k}^{k+H_r} (\hat{x}_H(i|k)-G)^T P_1 (\hat{x}_H(i|k)-G)} \quad (32)$$

In order not to frequently change goals during operation, a threshold is added, i.e. the robot can only change from G_{old} to G_{new} if (33) is satisfied for $\Delta J > 0$.

$$J_G(G_{new}) + \Delta J < J_G(G_{old}) \quad (33)$$

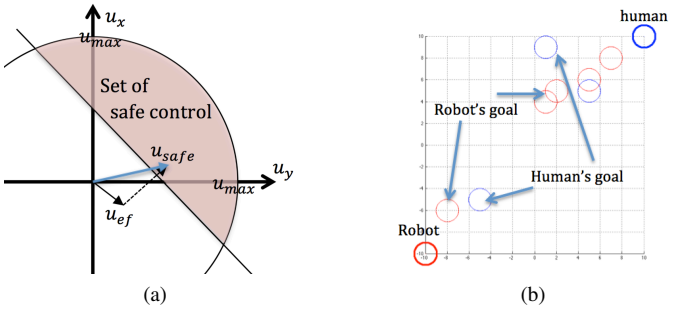
E. Optimal Control

To make the robot approach the goal safely and efficiently, we use receding horizon control with the preview horizon H_r. The optimization problem is formulated as follows:

$$\min J = \sum_{i=k_0}^{H_r+k_0} \{u_R^T(i) R_R u_R(i) + (x_R(i) - G_R(i))^T Q_R (x_R(i) - G_R(i)\} \quad (34)$$

$$s.t. \quad u_R(k_0) \in U_R(k_0); |u_R(i)| \leq u_{max}, \forall i$$

The cost function is analogous to one in the standard linear quadratic regulator (LQR) where $R_R \in \mathbb{R}^{2\times 2}$, $Q_R \in \mathbb{R}^{4\times 4}$ are positive definite. k_0 is the current time step. The safety requirement is put as a hard constraint while the efficiency term is being optimized. Only the safety requirement at k_0-th time step is enforced. At the next time step, the robot will solve this optimization again and the safety requirement for (k_0+1)-th time step is then considered. u_{max} is the maximum acceptable acceleration.



(a) (b)

Fig. 3: (a) Suboptimal solution; (b) Simulation environment

TABLE I: Parameters in the Simulation

Symbol	Name	Value
α	Filter gain	0.8
u_{max}	Maximum acceptable acceleration	10
λ	Forgetting factor	1 or 0.98
H_r	Preview horizon	5
T_s	Sampling interval	0.1s
D^*	Safety threshold	10

V. Simulation

The simulation environment (20×20) is shown in Fig.3b. Both the human and the robot are represented by circles of radius 1, while the robot is in red and the human in blue. The thin blue and the thin red circles are the goals for the human and the robot respectively. They are randomly generated in order to test the robot in different scenarios. The goals do not have any order. When there are less than three goals for an agent, new goals will be generated to keep the simulation running. The human-robot team needs to reach as many goals as possible without colliding with each other in a given time.

There are two simulation versions. One simulates the human behavior by computer (non-human involved simulation), and the other directly reads the human input through a multi-touch pad (human involved simulation). Under direct human control, the human-robot team can clear 83 goals in 100s on average, compared to 41 goals on average under simulated human case. This shows that the human understands a way to make the team operate more efficiently.

In the simulation, to make the calculation fast, we adopt a suboptimal way in solving (34). We first calculate the unconstrained optimal control input u_{ef}, then map it to the set of safe control. The final control input u_{safe} is marked in blue in Fig.3a. The calculation complexity is greatly reduced in this case, while the performance is not affected. The parameters in the simulation are shown in Table.I. Animations are available in the accompanying video.

A. Non-Human Involved Simulation

1) Learning Performance: Fig.4 shows the learning performance of the human behavior with and without a forgetting effect ($\lambda = 1$ or 0.98). The simulated human changed control law at $k = 500$. The first plot is the learning gain versus time, while the second and third plots are the normed errors between the estimated and the true A_i^* and B_i^*

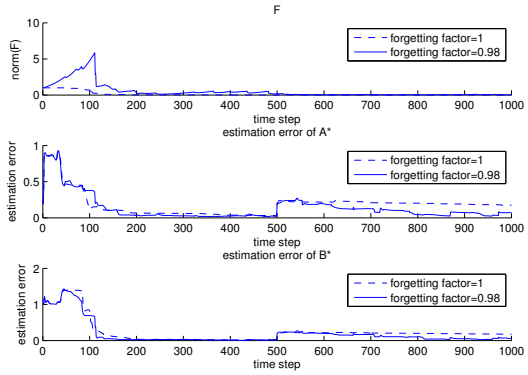


Fig. 4: Learning performance under different forgetting factor with simulated human

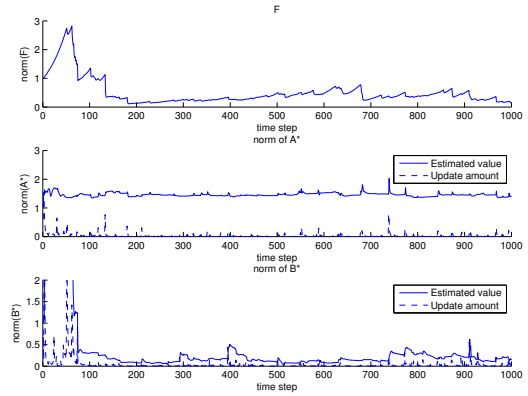


Fig. 5: Learning performance under $\lambda = 0.98$ with human control

matrices respectively. The learning gain is higher for smaller forgetting factor. The estimation performance before the 500-th time step is good in both cases. But the error convergence rate is faster under $\lambda = 0.98$ for $k > 500$. Thus the EM algorithm with a forgetting factor is a good learning tool of time-varying behavior.

2) Safety Performance: Fig.6 shows the human-avoidance behavior of the robot. At the starting point, the robot (red circle) and the human (blue circle) were close to each other's goal. When the simulation started, the robot was not quite sure how the human would behave. But the robot did slow down when observing that the human was approaching, since a large relative velocity towards each other violated the safety constraint. Then the robot detoured a little bit. Just at the moment the robot started to detour, the simulated human started to approach his next goal. Thus the robot followed the human in a safe distance and reached its goal successfully.

B. Human Involved Simulation

1) Learning Performance: Fig.5 shows the learning performance with $\lambda = 0.98$ under human control. The first plot is the learning gain, while the second and third plots are the norms of the learned $\hat{A}_i^*(k)$ and $\hat{B}_i^*(k)$ matrices and their

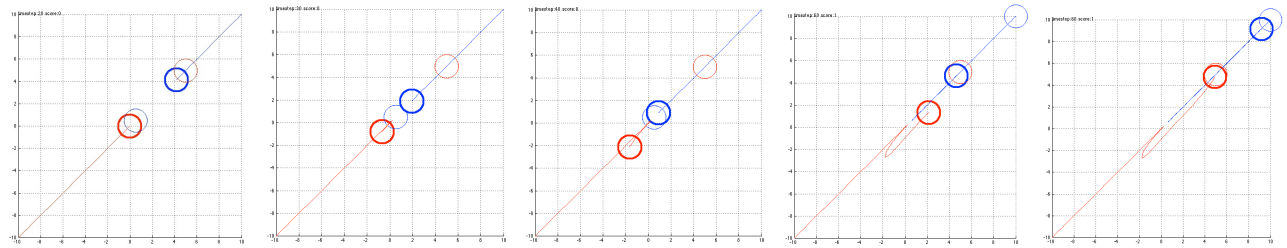


Fig. 6: Safety performance - human avoidance by detouring

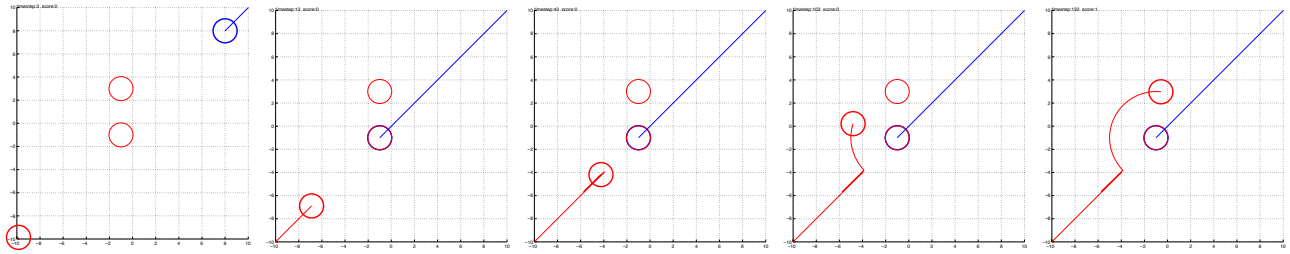


Fig. 7: Safety performance - human avoidance by changing goal point

updating amount at each time step. The estimated values are consistent.

2) Safety Performance: Fig.7 shows the safety performance when the human mistakenly stayed on one of the robot's goals. In this simulation, there are two goals for the robot. At the starting point, the robot was to go to the lower goal. Then the human occupied that goal unexpectedly. The robot wandered around for a while. After learning that the human would not move, the robot changed its goal to the upper one and reached it successfully. This behavior demonstrates the effectiveness of the safety measure at the decision-making level.[4]

VI. Conlusion

In this paper, we proposed a two-layer interaction model for a workspace sharing human-robot team. The human's closed loop dynamics is linearized locally and learned by an online EM algorithm. Motion control for the robot is done in a dynamic safe set (dependent on the relative distance and velocity). An integrated design method for the controller and the decision-making center concerning inference on the human's goal, online learning of the human's closed loop behavior, optimal goal generation for the robot and receding horizon control in the safe set were proposed. The validity of the algorithm was confirmed by simulation. In the future, this algorithm will be implemented on a real robot. Problems such as sensor deficiency and multi-rate sensing and control are to be solved.

Appendix

Proof of Proposition 1: Suppose the proposition statement is not true. Thus $U = \{t : \phi(t) \leq 0, d(t) < d_{min}\}$ is not empty, which implies that $\exists t_2$, s.t. $d_{min}^2 - k_\phi \dot{d}(t_2) \leq d^2(t_2) < d_{min}^2$. Since d is a continuous function of t, $F = d^{-1}([d(t_2), d_{min}]) \cap [t_0, t_2]$ is a closed set in the time horizon $[t_0, t_2]$. Pick the largest connected set in F containing t_2, and denote it by $T = [t_1, t_2]$. Thus $d(t_1) = d_{min}$, and $\dot{d}(t) \leq d_{min}$, $\forall t \in (t_1, t_2]$. In order to satisfy $\phi(t) \leq 0$, $\dot{d}(t)$ needs to satisfy $\dot{d}(t) \geq 0$, $\forall t \in (t_1, t_2]$. So $d(t_2) = d(t_1) + \int_{t_1}^{t_2} \dot{d}(t)dt \geq d(t_1) = d_{min}$, which is a contradiction. Thus $\{d : \phi(t) \leq 0, \forall t > t_0, d(t_0) > d_{min}\} \subset \{d : d(t) \geq d_{min}, \forall t > t_0\}$.

[4]The robot may go to the upper goal directly if it has already learned that the human would stay on the lower goal for a while. However, in our simulation, the robot does not have the prior knowledge of the human behavior. Thus we still consider the robot's action as efficient.

Acknowledgement

The authors gratefully acknowledge the reviewer's comments and also thank Wenlong Zhang, Kevin Haninger and Chi-Shen Tsai for their precious suggestions for this paper.

References

[1] At Volkswagen, Robots Are Coming Out Of Their Cages. *Co.Exist.* N.p., 9 Sept. 2013.

[2] Our Friends Electric. *The Economist*. The Economist Newspaper, 7 Sept. 2013.

[3] G.Charalambous. Human-Automation Collarboration in Manufacturing: Identifying Key Implementation Factors. In *Contemporary Ergonomics and Human Factors 2013*, p. 59. Taylor & Francis, 2013.

[4] J.Krger, T. K. Lien, and A. Verl. Cooperation of human and machines in assembly lines. *CIRP Annals-Manufacturing Technology* 58, no. 2 (2009): 628-646.

[5] X.Na, and David J. Cole. Linear quadratic game and non-cooperative predictive methods for potential application to modelling driverAFS interactive steering control. *Vehicle System Dynamics* 51, no. 2 (2013): 165-198.

[6] P.Trautman, and A.Krause. Unfreezing the robot: Navigation in dense, interacting crowds. In *Intelligent Robots and Systems (IROS), 2010 IEEE/RSJ International Conference on,* pp. 797-803. 2010.

[7] N.Du Toit , and Joel W. Burdick. Robot motion planning in dynamic, uncertain environments. *Robotics, IEEE Transactions on* 28, no. 1 (2012): 101-115.

[8] L.Gracia, F.Garelli, and A.Sala. Reactive Sliding-Mode Algorithm for Collision Avoidance in Robotic Systems. *Control Systems Technology, IEEE Transactions on*, in press.

[9] P.Liang, and D.Klein. Online EM for unsupervised models. In *Proceedings of human language technologies 2009*, pp. 611-619. 2009.