# Constraint Task-Based Control in Industrial Settings

Claus Lenz, Markus Rickert, Giorgio Panin, Alois Knoll
Robotics and Embedded Systems, Department of Informatics, Technische Universität München

***Abstract*— Direct physical human-robot interaction has become a central part in the research field of robotics today. To use the advantages of the potential for humans and robots to work together as a team in industrial settings, the most important issues are safety for the human and an easy way to describe tasks for the robot. In this work, we present an approach of a hierarchical structured control of industrial robots for joint-action scenarios. Multiple atomic tasks including dynamic collision avoidance, operational position, and posture can be combined in an arbitrary order respecting constraints of higher priority tasks. The controller flow is based on the theory of orthogonal projection using nullspaces and constraint least-square optimization. To proof the approach, we present three collaboration scenarios between a human and an industrial robot.**

## I. Introduction

Currently, interaction of human and robot is mainly reduced to a master-slave scheme with a human tele-operating the robot or programing it off-line. To ensure safety, the workspaces of humans and robots are strictly separated in time or space in the industry. But while direct physical interaction with stunning new hard- and software developments [1], [2] and joint-action of humans and robots are emerging fast in the research field of robotics [3], [4], [5], [6], [7], [8], most of the robots used in these scenarios are self-made or lightweight to guarantee safety through mechanical methods. However, standard industrial robots used in the industry often have unknown dynamic parameters, the control is often only possible on position- or velocity-level, and due to their masses and their high velocities the risk of injuries is very high [9]. Therefore, these robots cannot fulfill crucial requirements needed for safe direct physical interaction.

Although industrial norms are changing and allow direct cooperation of human and industrial robot (ISO 10218-1), the demands on the robots include a speed limit of $250\,\mathrm{mm/s}$ at the tool center point (*TCP*), limits on the dynamically received power of $80\,\mathrm{W}$, and limited force on the tool center point of $150\,\mathrm{N}$. With these limitations, an efficient collaboration with current industrial robots is not possible, although the applications of such collaborative human-industrial robot teams cover a magnitude of tasks needed in the industry—including support in carrying heavy objects, virtual rails, or robot assistance in hybrid assembly scenarios as depicted in Fig. 1.

In the latter scenario, human and industrial robot assemble a product cooperatively. To keep the collaboration fast and efficient, robot and human need to share the same workspace. This enables physical support of the worker or direct handover of parts and tools from the robot to the human and vice versa. But of course, safety for the human counterpart needs to be a central part in this kind of scenarios to take advantages of the potential for humans and robots to work together as a team in industrial settings.

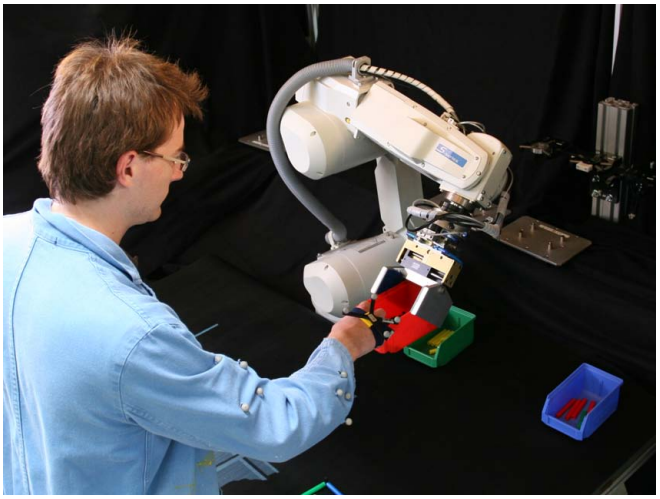


Fig. 1. *Hybrid assembly:* collaboration scenario of a human and an industrial robot. The markers on the upper, lower arm, and the palm are used to estimate the pose of the human co-worker.

Therefore, robots need to have different hierarchical high-level abilities that can be used according to the current situation to solve a given task jointly by a human and a robot. This can be solved using atomic action primitives (*tasks*) that can be joined to high-level actions.

In this paper, we propose an approach of a hierarchical structured control of industrial robots in joint-action scenarios. In this way, it is possible to combine multiple (sub-)tasks in an arbitrary order respecting constraints of higher priority tasks. A collision avoidance task can be easily set to have high priority in situations where a direct physical contact might be possible and has to be prevented. Although we only had a standard position controlled industrial robot available in our setup, a combination of the presented methodology along with a lightweight robot and its reactivity in case of actual collisions [10], [11] would enable an even more efficient way for human-robot collaboration.

The remainder of this work is organized as follows: Section II explains single robot tasks, including posture, operational position, and collision avoidance control, which can be stacked in a hierarchical way. Section III presents in detail three concrete applications of human-robot collaboration in an assembly scenario.

## II. Robot Tasks

According to [12], actions consist of several atomic tasks that are arranged in a task-oriented way. An encoding of priorities in the task-stacking hierarchy is of importance, because it needs to be prevented that contrary tasks interfere with each other and lead to uncontrollable or unwanted behavior of the robot.

Based on the syntax of [13], an action $A$ can be formulated as a composition of tasks $T_k$ with a projection rule $\triangleleft_k$ that ensures the behavior of the task:

$$A = \langle T \rangle_0^n = T_n \triangleleft_n T_{n-1} \triangleleft_{n-1} \dots \triangleleft_1 T_0. \tag{1}$$

As most industrial robots—including the one used in our experiments—are only controllable on the position- or velocity-level, we transform the task descriptions in terms of joint velocities ($\underline{\dot{q}}$):

$$\underline{\dot{q}}_{\text{Action}} = \underline{\dot{q}}_{T_n} \triangleleft_n \underline{\dot{q}}_{T_{n-1}} \triangleleft_{n-1} \dots \triangleleft_1 \underline{\dot{q}}_{T_0}. \tag{2}$$

According to the defined control structure, we need to take care of only a few points for each task: compute the velocity to solve the single task, set the constraints according to the current situation or use static constraints, and finally transform the lower priority task velocity into a safe subspace respecting the active constraints. For the latter we are using *nullspaces* with orthogonal projectors as in [12], [14] in the posture and the operational position task and a constraint least-square optimization for the collision avoidance task. With these projectors we are able to decouple the tasks from each other (see Figure 2).

### A. Respecting Hardware Limits

An important issue that needs to be respected are the limitations of joint angles, velocities and accelerations. The resulting velocity $\underline{\dot{q}}$—as well as intermediate results—needs to be in a certain bounding region $\underline{l} \leq \underline{\dot{q}} \leq \underline{u}$, that does not violate these limits. Therefore, we adaptively re-compute the bounding limits in every time step according to

$$\underline{l}_i = \max(\dot{q}_i^{\text{min}}, \dot{q}_i^{\text{lowerJL}}, \dot{q}_i^{\text{AL}}) \tag{3}$$

for the lower boundary and

$$\underline{u}_i = \min(\dot{q}_i^{\text{max}}, \dot{q}_i^{\text{upperJL}}, \dot{q}_i^{\text{DL}}) \tag{4}$$

for the upper boundary. The values for $\dot{q}_i^{\text{max}}$ and $\dot{q}_i^{\text{min}}$ are the maximum and minimum velocity for joint $i$ as defined by the manufacturer of the robot.

$$\dot{q}_i^{\text{lowerJL}} = \frac{q_i^{\text{min}} - q_i(t)}{\Delta t} \tag{5}$$

is the velocity that is needed to reach the lower joint limit of joint $i$ in the next time step $t+1$ and

$$\dot{q}_i^{\text{upperJL}} = \frac{q_i^{\text{max}} - q_i(t)}{\Delta t} \tag{6}$$

is the velocity to reach the upper joint limits respectively. These velocities converge to zero as the joint angle is approaching the joint limit.

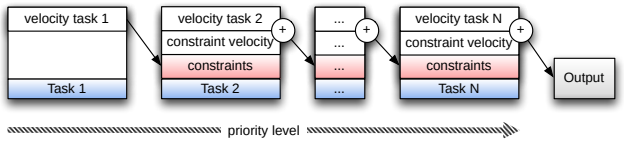



Fig. 2. Actions are defined through task compositions. The associated constraints are respected through projections into corresponding subspaces, with execution priorities ranging from lowest (left) to highest (right).

To respect that the acceleration and the deceleration abilities of the motors of the robot are also limited, the maximum possible velocities of the joints need to be constraint. Therefore, we approximate this factor with

$$\dot{q}_i^{\text{AL}} = \ddot{\underline{q}}_i^{\text{acc}} \cdot \Delta t + \dot{q}_i(t-1) \tag{7}$$

for the acceleration and

$$\dot{q}_i^{\text{DL}} = \ddot{\underline{q}}_i^{\text{dec}} \cdot \Delta t + \dot{q}_i(t-1) \tag{8}$$

for the deceleration of the joints.

With the limit information for each joint, a limit method $C(\underline{\dot{q}})$ can be defined that scales the velocity vector to respect all above mentioned limits *without* changing the trajectory of the motion:

$$C(\underline{\dot{q}}) = s(\underline{l}, \underline{u}) \cdot \underline{\dot{q}}. \tag{9}$$

### B. Task: Joint Position (Posture)

The goal of the posture task is to drive the robot to a certain joint configuration $\underline{q}_{goal}$. Using the *Posture*-controller, it is possible to give the robot a specific—e.g., "human friendly" or upright—posture.

The velocity for this task is calculated by

$$\underline{\dot{q}}_{po} = C\left(\frac{\underline{q}_{goal} - \underline{q}(t)}{\Delta t}\right). \tag{10}$$

To constrain certain degrees of freedom in joint space, that cannot be influenced by lower priority tasks, a $n \times n$ matrix $S_{po}$ is used to select them, with $n$ being the number of joints. This means the guaranteed velocity can be expressed as

$$\underline{\dot{q}}_{po}^* = S \cdot \underline{\dot{q}}_{po}. \tag{11}$$

Using the projector $N_{Po}$ of the selected constraints, we get as output for the projection of an input velocity $\underline{\dot{q}}_{in}$:

$$\underline{\dot{q}}^* = \underline{\dot{q}}_{po}^* + N_{po} \cdot \underline{\dot{q}}_{in} \tag{12}$$

with

$$N_{po} = I - S_{po}. \tag{13}$$

### C. Task: Operational Position

The operational position task drives the tool center point of the robot to a defined goal position and orientation $\underline{x}_{goal}$ in Cartesian coordinates according to:

$$\underline{\dot{x}} = \frac{\underline{x}_{goal} - \underline{x}(t)}{\Delta t}. \tag{14}$$

After the velocity in Cartesian space is calculated, constraints can be set using a diagonal selection $6 \times 6$ matrix

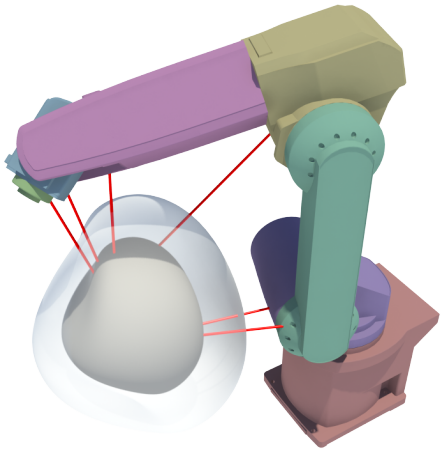


Fig. 3. Computing the repelling forces of an obstacle: The red lines illustrate the minimum distances of an obstacle to the body parts of the robot in a given joint configuration. If a distance is below a chosen security threshold (transparent bubble), the distance is used to compute virtual forces on the robot using potential fields.

$S_{Op}$ to select the degrees of freedom (in Cartesian space) that should not be influenced by lower priority tasks. Transformed into limited joint velocities using a singularity robust pseudo-inverse $J_e^\dagger$ of the Jacobian $J_e$ of the end-effector, we get

$$\underline{\dot{q}}_{op}^* = C(J_e^\dagger \cdot S_{op} \cdot \underline{\dot{x}}_{op}). \tag{15}$$

Using the nullspace $N_{op}$ of the selected constraints, we get as output for the orthogonal projection of an input velocity $\underline{\dot{q}}_{in}$:

$$\underline{\dot{q}}^* = \underline{\dot{q}}_{op}^* + N_{op} \cdot \underline{\dot{q}}_{in} \tag{16}$$

with

$$N_{op} = I - J_e^\dagger S_{op} J_e. \tag{17}$$

### D. Task: Collision Avoidance

Collisions need to be avoided for static (i.e., the workbench) and dynamic environment (i.e., the human and moving obstacles). In this task, the avoidance is done in a reactive way with dynamically updated collision scenes that is interfaceable with a variety of sensors.

The main challenge that arises here, is that the planned motion and the avoidance motion must be handled in a way where they do not interfere with each other. Therefore, we fuse potential field methodology to repel the robot from the obstacle with a constraint least-square optimization, that restricts the motion of the robot to safe orthogonal subspaces of the collision avoidance.

*1) Virtual forces:* To compute the velocity that repels the robot from surrounding obstacles, we need to compute the minimum distances of all objects in the environment model (including self-collision) to all body parts of the robot. Fig. 3 depicts the body parts of the used robot in different colors along with an example of the minimum distances $d_i$ (red lines) from an obstacle to a given joint configuration.

Opposite to simplified and only approximated models of manipulators (e.g., used in the skeleton algorithm presented in [15]), we measure the distances of arbitrary shapes to a convex version of the real CAD-model of the robot in order to reach a high precision of the virtual forces. With an efficient implementation of the GJK algorithm [16], we compute these distances faster than the update rate of the robot controller.

After calculating the minimum distance vectors $v_{x,i}$ in Cartesian space (i.e. the direction of the applied virtual force), we need to transform them to velocities in joint space and find the overall motion of the robot to avoid the collision. This is done according to

$$\underline{\dot{q}} = \sum_i^I \underline{\dot{q}}_i = \sum_i^I J_{P_r(i)}^T \cdot U_{rep,i}(q) \cdot v_{x,i}(q), \tag{18}$$

with $I$ being the number of bodies of the robot, the current joint configuration of the robot $q$, the Jacobian of the minimum distance point on the robot $J_{P_r(i)}$ and the repelling potential function $U_{rep,i}$:

$$U_{rep,i}(q) = \begin{cases} \frac{1}{2}\eta_i \left(\frac{1}{d_i(q)} - \frac{1}{Q^*}\right)^2 & \text{if } d_i(q) \leq Q^* \\ 0 & \text{if } d_i(q) > Q^* \end{cases},$$

with $Q^*$ being the distance at which the potential field function is applied (see transparent bubble in Fig. 3).

*2) Constraint least-square minimization:* To ensure that we *project* the lower priority task in an orthogonal subspace of the collision avoidance task, we use the mathematical framework of quadratic programming [17] to minimize the quadratic error between optimal velocity of the lower priority task subject and the constraints of the higher priority task. The low-priority task execution ($\underline{\dot{q}}_{in}$) is optimized regarding *must have* constraints of the higher priority task. The projection is described according to:

$$\min_{\underline{\dot{q}}_{in}} \| J_e \cdot \underline{\dot{q}}_{in} - \underline{\dot{x}}_t \|^2, \tag{19}$$

where $\underline{\dot{x}}_t$ is the ideal linear and angular velocity to solve the lower priority task subject to the linear constraints of the form

$$C^T \underline{\dot{q}}_t \geq 0 \tag{20}$$

with the constraint matrix

$$\underline{C}^T = \begin{pmatrix} \underline{\dot{q}}_1^T \\ \vdots \\ \underline{\dot{q}}_I^T \end{pmatrix} \tag{21}$$

and $\underline{\dot{q}}_i$ calculated according to (18). This means only those velocities are valid, that are orthogonal to the direction of the collision avoidance velocities or point in a direction that leaves the defined safety region. The output of the minimization process is then equal to the constrained joint velocity $\underline{\dot{q}}^*$. Quadratic Programming in combination with collision avoidance on the level of joint acceleration was used in [18], [19].

## III. Application Examples

As stated in the introduction, with the presented task-based hierarchical control structure of the industrial robot we can solve a magnitude of tasks that can be used in production scenarios to take advantage of collaboration of human and robot as a team. In the following sections, we present three applications used on our demonstration platform *JAHIR* (Joint-Action for Humans and Industrial Robots), [20] that is embedded in a factory setting [21].

In the used setup, the human is standing face to face with the industrial robot in front of a working table. Both, human and robot, can access the working table and therefore share the same workspace. The body of the human standing in front of the shared working table is secured by an approximated cylinder (see Fig. 6).

In the following applications, we have used human motion data and applied it to our dynamic 3D representation for distance measuring. The motion data was estimated using a marker-based infra-red tracking system with markers placed on the palm and on the lower and upper arm (see Fig. 1).

### A. Mobile Storage Box

In manual production, the efficiency of the current production step depends highly on the availability of parts. If different parts needed for a certain step are always within reach, the human can take them efficiently. On the other side, parts that are pre-assembled and not needed at the moment, need to be placed somewhere where they can be accessed easily when required.

In the first application scenario, we use the industrial robot as a mobile storage box. Parts can be placed in the box and the robot needs to guarantee they are not falling off. To be always within reach, the box follows the human hand, but avoids collision with it and the surrounding environment. Regarding these requirements, the controllers presented in Section II are arranged to compose action $A_1$ as follows:

$$A_1 = T_{\text{orientation}} \triangleleft T_{\text{avoidance}} \triangleleft T_{\text{position}} \triangleleft T_{\text{posture}}. \tag{22}$$

$T_{\text{orientation}}$ is the task with the highest priority, which takes care of keeping the box always in a horizontal orientation. The operational position controller is used here with the selection matrix

$$S_{\text{orientation}} = \text{diag}\,(0, 0, 0, 1, 1, 1) \tag{23}$$

to fix the orientation of the box. Task $T_{\text{avoidance}}$ avoids collisions with the surrounding environment and the human hand. The position task $T_{\text{position}}$ follows the human hand through updates of the hand tracking system to the goal position $x_{goal}$, that should be $0.1$ m in front and below of the hand. To keep the position fixed, the selection matrix

$$S_{\text{position}} = \text{diag}\,(1, 1, 1, 0, 0, 0) \tag{24}$$

is used. In the posture task, we defined that the robot should have an upright joint configuration. Because the posture task has the lowest priority, we can include all joints in the velocity calculation.

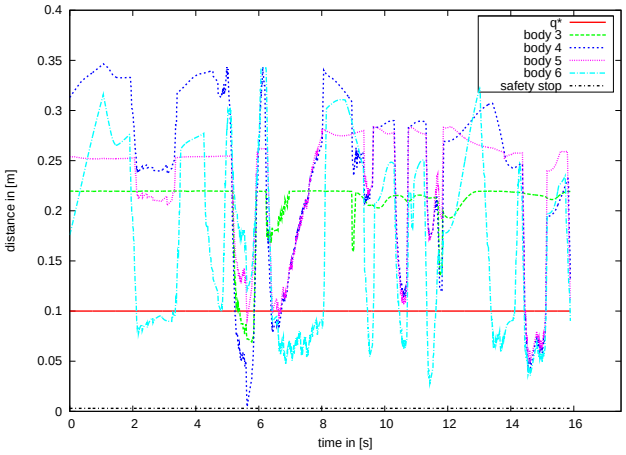



Fig. 5. Minimum distances of obstacles and robot bodies. The potential field is applied for distances that fall below $0.1$ m. If a critical minimum distance of $0.005$ m is reached, the robot stops its motion.

The result of this experiment is depicted in Fig. 4. The robot is carrying a red box with parts in its gripper, so that the human can grasp out of the box (Fig. 4(a)). To be able to grasp something, the motion of the robot needs to be stopped through the avoidance task that measures the distances. This is done in the controller with a defined distance, to stop the current motion, if the distance of robot and obstacle (i.e. hand) is touching or is below. Fig 5 shows the minimum distances of obstacles and robot with the two security distances $Q^*$, that enables the virtual repelling forces, and the *security stop* line.

### B. Direct Line-of-Sight

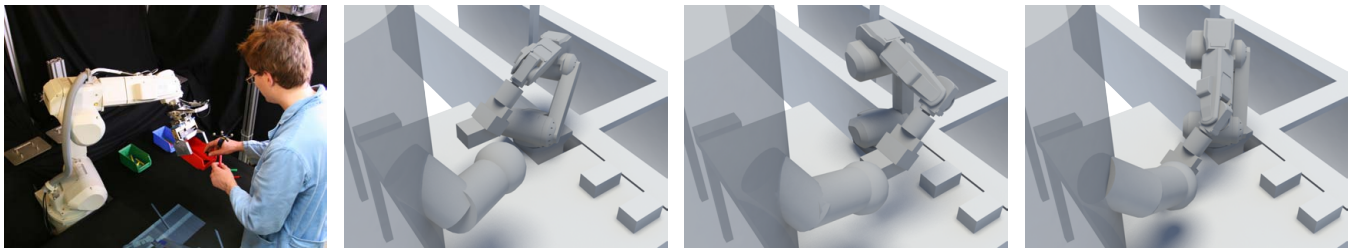
In our demonstration scenario, a top-mounted camera directed towards the working table is used to recognize, inspect and track objects lying on the table. To recognize objects reliably or to inspect objects according to defects, the camera needs direct line-of-sight for a certain amount of time. However, the robot should not be stopped, because it can fulfill other tasks in the meantime—including picking up an object at a position and placing it somewhere else or handing over tools needed for the next assembly steps to the human.

What needs to be considered here is the issue that the collision avoidance with the environment has to have a higher priority than to avoid the crossing of the line-of-sight of the camera with the robot. Therefore, two collision avoidance controllers with different collision scenes need to be specified.

If we describe the action of the robot again according to our controller scheme, we get action $A_2$:

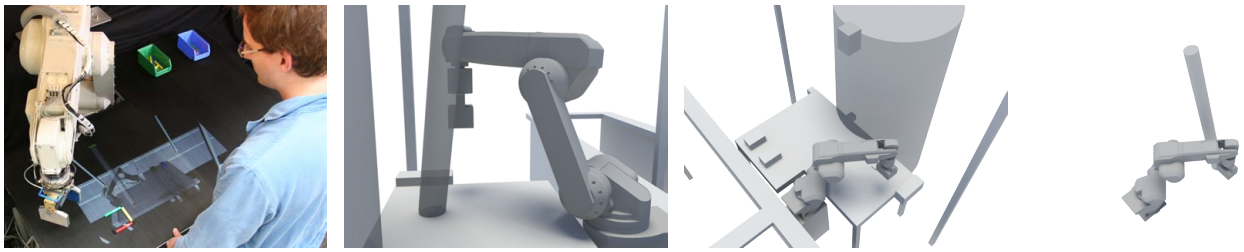
$$A_2 = T_{\text{environment}} \triangleleft T_{\text{orient.}} \triangleleft T_{\text{line-of-sight}} \triangleleft T_{\text{position}} \triangleleft T_{\text{posture}}. \tag{25}$$

$T_{\text{environment}}$ is the collision avoidance controller without the line-of-sight as depicted in Fig. 6(c). $T_{\text{orientation}}$ is the controller taking care of the orientation of the gripper with the same selection matrix as in the previous experiment. In the collision controller $T_{\text{line of sight}}$, the line-of-sight from the



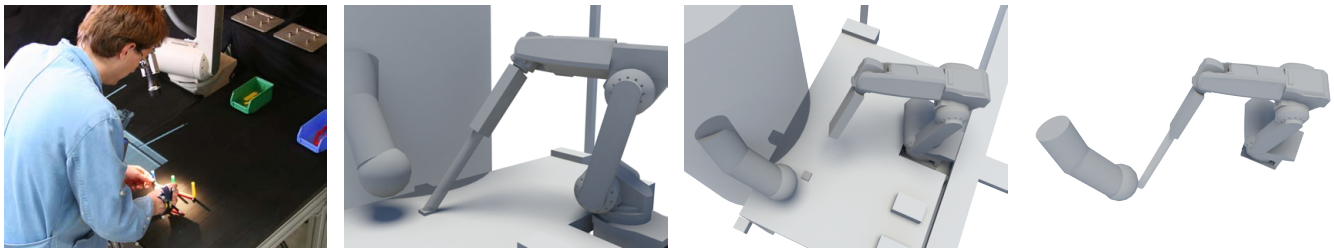
(a) (b) (c) (d)

Fig. 4. Mobile storage box: The robot has a storage box as tool which follows the human hand in safe distance and avoids collision with the hand and the surrounding, so that the human can pick up parts or place assembled products in there. (a) real scene; (b), (c) and (d) 3D representation with robot behavior.



(a) (b) (c) (d)

Fig. 6. Direct line-of-sight: The robot picks up an object at a position and needs to place it on another specified position. A top-down camera is inspecting another object on the table and needs to have always direct line-of-sight for this task. Therefore, the robot needs to find a way around the line from camera to object, respecting also the human working in the same workspace. From left to right: (a) real scene; (b) 3D representation; 3D scenes of the collision avoidance controllers to avoid collisions with the environment (c) and the line-of-sight of the camera (d).



(a) (b) (c) (d)

Fig. 7. Adaptive light source: In this application, the robot has a lamp mounted as tool that is used to illuminate a component lying on the table. Therefore, the position of the cone of light needs to stay fixed, but collisions with the human need to be avoided. From left to right: (a) real scene; (b) 3D representation; collision scenes used in the two collision avoidance controllers: (c) collisions with the environment (including the human) and (d) collisions of the light beam with the human.

camera is approximated by a cylinder from the camera to the object on the table as depicted in Fig. 6(d). The task $T_{\text{position}}$ drives the robot to the goal position. The posture task $T_{\text{posture}}$ is the same as in the previous experiment.

The images in Fig. 6 show the behavior of the robot moving from position $(0.1\ \text{m}, 0.2\ \text{m}, 0.3\ \text{m})$ (Fig. 6(a)) to the goal position $(0.8\ \text{m}, 0.3\ \text{m}, 0.1\ \text{m})$. As depicted in Fig. 6(c) and 6(d), the robot gets repelled by the potential field generated by different collision scenes.

If the human worker is busy fulfilling a complex assembly step on an object lying on the desk, we can also model the focus-of-attention of the human employing e.g. 3D face tracking algorithms as presented in [22]. The line-of-sight cylinder in this example would range from the human's head towards the direction his head is turned and the robot would avoid crossing the field of view of the human.

### C. Adaptive Light Source

In this application example, the robot has a light source mounted as its tool. With this lamp, an object lying on the table is to be illuminated. Therefore, the position of the light cone needs to stay fixed on the selected object, but collisions with the human need to be avoided. To enable this action, we describe the wanted action $A_3$ as follows:

$$A_3 = T_{\text{position}} \triangleleft T_{\text{environment}} \triangleleft T_{\text{posture}} \triangleleft T_{\text{light}}. \tag{26}$$

The same position task $T_{\text{position}}$ (with static goal position) has now the highest priority task, therefore assuring that the cone of light is directed towards the selected object. $T_{\text{environment}}$ is the avoidance controller with the environment model without

the cone of light, as depicted in Fig. 7(c). $T_{\text{posture}}$ tries to keep the robot in an upright position, with the constraint that the first joint should be at 0 degrees and joint 3 (connecting body 3 and 4—see Fig. 3) should form a right angle. The selection matrix for this task is

$$S_{\text{posture}} = \text{diag}\,(1, 0, 1, 0, 0, 0)\,. \tag{27}$$

The task $T_{\text{light}}$ to avoid collisions with the light beam has the lowest priority, because on the one hand it is not dangerous to cross a light beam with a hand and on the other side it does not damage the robot. For this controller, only the light beam and the arm position are modeled in the collision scene as shown in Fig. 7(d). The real set-up is depicted in Figure 7(a)) along with the dynamic 3D scene (Fig. 7(b)) used for visualization.

## IV. Conclusion and Future Work

In this paper, we have presented an approach to define robot actions in order to control an industrial robot through a hierarchical task-based control structure. To cope with any interference of competitive tasks, we employed nullspaces and constraint least-square optimization to project task velocities in safe subspaces of higher priority tasks. For the used industrial setup, we presented several examples built with this control structure. The robot changed its behavior with a simple rearrangement of basic tasks. The possibility to arrange atomic tasks, that are able to project into an orthogonal subspace including a collision avoidance task, opens new ways of programming robots in the presence of humans.

To increase the stability of the collision avoidance, more distances than only the minimum distance can be used to compute the virtual forces. Additional, more atomic tasks need to be added to the controller scheme including singularity and joint limit avoidance. To improve the security of the human, the whole human body pose needs to be estimated and tracked in adequate speed—at best without the use of any markers.

## V. Acknowledgements

We would like to thank Wolfgang Rösel for his help in setting up the real world scenarios. This work is supported by the DFG cluster of excellence "CoTeSys" (www.cotesys.org) and by the EU FP6 IST Cognitive Systems Integrated Project "JAST" (FP6-003747-IP) (www.euprojects-jast.net).

## References

[1] S. Wolf and G. Hirzinger, "A new variable stiffness design: Matching requirements of the next robot generation," in *Proceedings of the IEEE International Conference on Robotics and Automation*, 2008, pp. 1741–1746.

[2] D. Shin, I. Sardellitti, and O. Khatib, "A hybrid actuation approach for human-friendly robot design," in *Proceedings of the IEEE International Conference on Robotics and Automation*, 2008, pp. 1741–1746.

[3] M. Rickert, M. E. Foster, M. Giuliani, T. By, G. Panin, and A. Knoll, "Integrating language, vision and action for human robot dialog systems," in *Proceedings of the International Conference on Universal Access in Human-Computer Interaction, HCI International, Part II*, ser. Lecture Notes in Computer Science, vol. 4555. Springer, 2007, pp. 987–995.

[4] O. Schrempf, U. Hanebeck, A. Schmid, and H. Worn, "A novel approach to proactive human-robot cooperation," in *Proceedings of the IEEE International Workshop on Robot and Human Interactive Communication*, 2005, pp. 555–560.

[5] C. Burghart, R. Mikut, R. Stiefelhagen, T. Asfour, H. Holzapfel, P. Steinhaus, and R. Dillmann, "A cognitive architecture for a humanoid robot: a first approach," in *Proceedings of the IEEE-RAS International Conference on Humanoid Robots*, 2005, pp. 357–362.

[6] C. Breazeal, A. Brooks, J. Gray, G. Hoffman, C. Kidd, J. Lee, H.and Lieberman, A. Lockerd, and D. Mulanda, "Tutelage and collaboration for humanoid robots," *International Journal of Humanoid Robotics*, vol. 1, no. 2, pp. 315–348, 2004.

[7] G. Hoffman and C. Breazeal, "Effects of anticipatory action on human-robot teamwork efficiency, fluency, and perception of team," in *Proceedings of the ACM/IEEE International Conference on Human-Robot Interaction*, 2007, pp. 1–8.

[8] T. W. Fong, C. Kunz, L. Hiatt, and M. Bugajska, "The human-robot interaction operating system," in *Proceedings of the ACM/IEEE International Conference on Human-Robot Interaction*, 2006, pp. 41–48.

[9] S. Haddadin, A. Albu-Schäffer, M. Frommberger, and G. Hirzinger, "The role of the robot mass and velocity in physical human-robot interaction - part I: Non-constrained blunt impacts," in *Proceedings of the IEEE International Conference on Robotics and Automation*, 2008, pp. 1331–1338.

[10] A. D. Luca, A. Albu-Schäffer, S. Haddadin, and G. Hirzinger, "Collision detection and safe reaction with the DLR-III lightweight manipulator arm," in *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2006, pp. 1623–1630.

[11] S. Haddadin, A. Albu-Schäffer, M. Frommberger, and G. Hirzinger, "The role of the robot mass and velocity in physical human-robot interaction - part II: Constrained blunt impacts," in *Proceedings of the IEEE International Conference on Robotics and Automation*, 2008, pp. 1339–1345.

[12] L. Sentis and O. Khatib, "Task-oriented control of humanoid robots through prioritization," in *Proceedings of the IEEE-RAS/RSJ International Conference on Humanoid Robots*, 2004.

[13] Y. Yang and O. Brock, "Elastic roadmaps: Globally task-consistent motion for autonomous mobile manipulation in dynamic environments," in *Proceedings of the Robotics Science and Systems Conference*, 2006.

[14] L. Sentis, "Synthesis and control of whole-body behaviors in humanoid systems," Ph.D. dissertation, Standford University, 2007.

[15] A. De Santis, A. Albu-Schäffer, C. Ott, B. Siciliano, and G. Hirzinger, "The skeleton algorithm for self-collision avoidance of a humanoid manipulator," in *Proceedings of the IEEE/ASME International Conference on Advanced Intelligent Mechatronics*, 2007, pp. 1–6.

[16] G. van den Bergen, "A fast and robust GJK implementation for collision detection of convex objects," *Journal of Graphics Tools*, vol. 4, no. 2, pp. 7–25, 1999.

[17] R. Fletcher, "Quadratic programming," in *Practical Methods of Optimization*, 2nd ed. New York: Wiley, 1987, ch. 10, pp. 229–258.

[18] E. Freund, M. Schluse, and J. Rossmann, "Dynamic collision avoidance for redundant multi-robot systems," in *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2001, pp. 1201–1206.

[19] E. Freund and J. Rossman, "The basic ideas of a proven dynamic collision avoidance approach for multi-robot manipulator systems," in *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2003, pp. 1173–1177.

[20] C. Lenz, S. Nair, M. Rickert, A. Knoll, W. Rösel, J. Gast, and F. Wallhoff, "Joint-action for humans and industrial robots for assembly tasks," in *Proceedings of the IEEE International Symposium on Robot and Human Interactive Communication*, 2008, pp. 130–135.

[21] M. Zäh, M. Beetz, K. Shea, G. Reinhart, K. Bender, C. Lau, M. Ostgathe, W. Vogl, M. Wiesbeck, M. Engelhard, C. Ertelt, T. Rühr, and M. Friedrich, *Changeable and Reconfigurable Manufacturing Systems*. Springer, 2009, ch. The Cognitive Factory, pp. 355–371.

[22] G. Panin and A. Knoll, "Real-time 3D face tracking with mutual information and active contours," in *Proceedings of the International Symposium on Visual Computing*, 2007, pp. 1–12.