Prioritized Motion-Force Control of Multi-Constraints for Industrial Manipulators

Caixia Cai[1], Nikhil Somani[2], Markus Rickert[2], Alois Knoll[1]

***Abstract*— To synthesize whole-body behaviors interactively, multiple tasks and constraints need to be simultaneously controlled, including those that guarantee that the constraints imposed by the robot's structure and the external environment are satisfied. In this paper, we present a prioritized, multiple-task control framework that is able to control forces in systems ranging from humanoids to industrial robots. Priorities between tasks are accomplished through null-space projection. Several relevant constraints (i.e., motion constraints, joint limits, force control) are tested to evaluate the control framework. Further, we evaluate the proposed approach in two typical industrial robotics applications: grasping of cylindrical objects and welding.**

I. Introduction and Related Work

Control of the physical interaction between a robot manipulator and the environment is crucial for the successful execution of a number of practical tasks where the robot end-effector has to manipulate an object or perform some operation on a surface. Typical examples in industrial settings include polishing, grasping, machining, or assembly. During contact, the environment may set constraints on the geometric paths that can be followed by the end-effector or other constraints for safety. A robot must accomplish a global task while satisfying several constraints. At the same time, the additional redundancy must also be controlled, thus requiring a control framework for multiple tasks or constraints.

Several frameworks for multi-task control of rigid robots exist in the literature. Most frameworks presented in the 80s, 90s [1, 2, 3, 4] and recently by Smits et al. (iTaSC) [5] and [6, 7] work at kinematic level, computing the desired joint velocities ($\dot{q}$) or accelerations ($\ddot{q}$). These approaches are not suited for robots that interact with the environment, because they do not allow for force control or impedance control. This motivated a more recent trend of torque control strategies [8, 9, 10, 11, 12], which consider the dynamics of the robot and compute the desired joint torques (τ). This approach can also improve tracking, as it compensates for the dynamic coupling between the joints of the multi-body system.

As mentioned in [13], a framework is *sound* if the control action of any task does not affect the performance of any higher priority tasks. A framework is *optimal* if its control action minimizes the error of each task, under the constraint of being *sound*. Finally, a framework is *efficient* if its computational complexity is minimal. All the control frameworks that we are interested in are *sound*, which means they are prioritized. Being *optimal* and *efficient* is crucial for real-world applications in industrial field.

Since we are interested in controlling robots that interact with the environment, we focus on frameworks that allow for force control. Peters et al. [14] demonstrated that we can derive several of these well-known torque control laws under a unifying framework, allowing for force control by setting the joint space control torques. This approach is efficient but not optimal. The Whole-Body Control Framework (WBCF) [8] allows for force control while being optimal, but not efficient. The framework (TSID) presented by Prete et al. [13] is motivated by designing a control framework that is both optimal and efficient. However, it does not allow for inequality constraints, which are particularly important for modeling joint limits and motor torque bounds. These are very important for safety in industrial robotics.

The frameworks presented above are usually applied on humanoid robots. In this paper, we present a prioritized, multiple-task control framework that allows for force control for industrial robotics. Although the TSID framework is efficient, we choose the WBCF framework for our work because of the fact that there are only very few DOFs for industrial manipulators. For this case, both WBCF and TSID have similar computation times. The main reason for us to choose WBCF is that it is easy to integrate multiple tasks and model inequality constraints.

Our proposed framework is based on a composable structure where several constraints, each describing a robot task or behavior, can be combined with priorities. The framework supports several types of constraints, such as operational task (position and orientation), force (contacts), or inequality constraints (e.g., joint limits, collision avoidance). For safety and for efficient control, the framework establishes a control hierarchy among behaviors, which is exploited to establish control priorities among the different control categories, i.e., constraints, operational tasks, and postures. Constraints should always be guaranteed, while operational tasks should be accomplished without violating the acting constraints. The priorities are accomplished by null-space projections.

In Section II, we review the Whole-Body Control Framework that allows us to establish a control hierarchy for multiple priority tasks. Section III demonstrates the frame work by testing several types of constraints. Section IV presents the evaluation of our approach on typical industrial robotics tasks. Finally, Section V concludes the paper and presents directions for future work.

Authors Affiliation: [1] Robotics and Embedded Systems, Department of Informatics, Technische Universität München, Munich, Germany. [2] fortiss GmbH, An-Institut Technische Universität München, Munich, Germany.
Contact: caica@in.tum.de

II. Whole-Body Control Framework

In this section, we review the hierarchical task control framework based on projecting the control of lower priority tasks into the null-space of higher priority tasks. In this context, we distinguish three priority levels in the hierarchy: Constraints (such as contacts, joint-limits, self-collisions), operational tasks (i.e., position, orientation motion), and postures (i.e., the residual motion), which should be controlled with different priority assignments. They are treated as independent control entities.

A. Integration of Constraints

This subsection describe the WBCF presented by Sentis et al. [8]. This framework is based on the Operational Space Formulation [15], which was introduced to address the dynamic interaction between the robot's task space motion and force, defining a dynamically consistent task null-space. We first review the fundamental mathematics and begin by describing the robot's joint space dynamics in terms of joint coordinates q with

$$M(q)\ddot{q} + C(q,\dot{q}) + G(q) = \tau\,, \tag{1}$$

where τ is the set of joint torques, $M(q)$ is the joint inertia matrix, $C(q,\dot{q})$ is the Coriolis and centrifugal torque vector, and $G(q)$ is the gravity torque vector.

The Operational Space Formulation describes the torque level decomposition of an operational task ($task_1$) and a secondary control task ($task_2$) according to the torque equation

$$\tau = \tau_{\text{task}_1} + \tau_{\text{task}_2}\,. \tag{2}$$

Based on the control algorithm projecting the control of lower priority tasks into the task null-space of higher priority tasks, the torque decomposition can be represented by

$$\tau = J^{\text{T}}_{\text{task}_1} F_{\text{task}_1} + N^{\text{T}}_{\text{task}_1}\tau_{\text{task}_2}\,, \tag{3}$$

where J_{task_1} is the Jacobian of task 1, F_{task_1} is a vector of forces, and $N_{task_1} = (I - J^{\dagger}_{task_1} J_{task_1})$ is the dynamically-consistent null space associated with the $task_1$. $J^{\dagger}_{task_1}$ is the dynamically-consistent generalized inverse of the task Jacobian.

B. Hierarchical Extension

In this subsection, we propose a control hierarchy that extends the previous decomposition to multiple levels. This hierarchy integrates constraints and additional tasks according to desired priorities, while optimizing the execution of the global task. Given n tasks controlling the robot behavior at a given time, the multi-level hierarchy is represented as

$$\tau = \sum_{i=1}^{n} J^{\text{T}}_{p(i)}\, F_{p(i)} \tag{4}$$

$$F_{p(i)} = \Lambda_{p(i)}\{\ddot{x}^*_i - \dot{J}_i\dot{q} + J_i M^{-1}(h - \sum_{j=1}^{i-1} J^{\text{T}}_{p(j)} F_{p(j)})\} \tag{5}$$

$$J_{p(i)} = J_i N_{p(i)}\,, \tag{6}$$

where τ, $F_{p(i)}$ and $J_{p(i)}$ are prioritized controls, prioritized forces, and projected Jacobian, respectively. $\ddot{x}^*_i$ is a reference input at the acceleration level, J_i is the task Jacobian, and $h = C + G$. $\Lambda_{p(i)}$ is the task-space mass matrix and $N_{p(i)}$ is an extend null-space matrix containing the null-spaces of all preceding constraints and tasks:

$$\Lambda_{p(i)} = (J_{p(i)}\, M^{-1}\, J^T_{p(i)})^{-1} \tag{7}$$

$$N_{p(i)} = I - \sum_{j=1}^{i-1} J^{\dagger}_{p(j)}\, J_{p(j)}\,. \tag{8}$$

This prioritization strategy minimizes the error of each task under the constraint of not conflicting with any higher priority tasks.

C. Hybrid Control

The framework allows for hybrid position/force control by setting (5)

$$F_{p(i)} = \Omega_f f^*_i + \Lambda_{p(i)}\{\Omega_m \ddot{x}^*_i - \dot{J}_i\dot{q} + J_i M^{-1}(h - \sum_{j=1}^{i-1}\tau_{p(j)})\}\,, \tag{9}$$

where the selection matrices Ω_f and Ω_m split the control space into force and motion components, respectively, f^*_i represent the constraint forces.

III. Examples of Constraints

The control framework supports several types of constraints, such as motion (position and orientation), force (contacts), or inequality constraints (e.g., joint limits, collision avoidance). The robot must accomplish a global task while satisfying several constraints. As an example, in the typical industrial task of inserting a peg into a hole, multiple constraints need to be simultaneously controlled, including position constraint, force control, and the robot's configuration constraints (i.e., joint limits and self-collision avoidance). This section shows some examples of different constraints.

A. Motion Constraints

The basic operational tasks are position and orientation movements with constraints. In the Peg-in-Hole scenario, the first step is to move the robot end-effector to the plane where the hole is on, then the robot needs to search the hole on that plane. Therefore, we test these two cases for our control framework on a 6 *DOF* industrial manipulator in simulation. The robot kinematics, dynamics, and low-level robot control are simulated using the Robotics Library[1] by Rickert with a realistic robot model.

1) ***Case 1 – Move to a plane****:* In this case, the robot is commanded to reach a plane with its end-effector while controlling the movement only along the plane normal direction. The plane is defined by a point (p) and its normal vector (n). During task execution, the distance d between the robot and the plane converges to zero when it reaches the plane. The orientation of the robot end-effector is finally aligned with

[1] http://www.roboticslibrary.org/

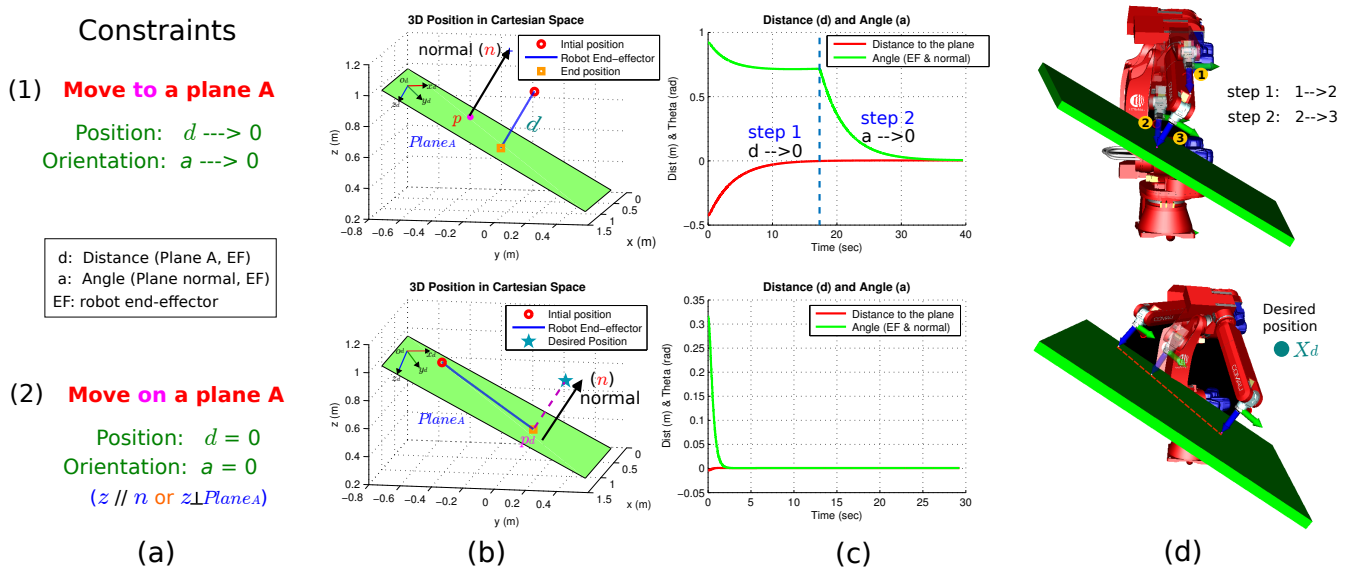

Fig. 1. Motion constraints tests for robot manipulators in two cases: (1) Move to a plane; (2) move on a plane. (a) Constraints for tests, (b) 3D position in the task space, (c) control parameters, (d) 3D visualization of coach for simulations.

the normal direction, which means the z axis of the end-effector is parallel to the plane normal n. The constraints are described in Fig. 1 (a). There are two steps:

Step 1: Move the robot to Plane$_A$ ($d \to 0$) with

$$\ddot{x}^*_{\text{position}} = \Omega_m (k_p\, d\, n - k_d\, \dot{X}_{\text{ef}}), \tag{10}$$

where n is the plane normal, d is the distance between the robot end-effector and Plane$_A$. $\dot{X}_{\text{ef}}$ is the robot linear velocity, and k_p, k_d are the positive control parameters. Moreover, we accomplish dynamic decoupling in the controllable directions according to the selection matrix Ω_m with

$$\Omega_m = R^d_o\, S_n\, R^o_d, \tag{11}$$

in which R^d_o is the transformation between the task frame O_d and the robot base frame O_o. In this case, we constrain the motion only in the n (z_d) direction, and the motion matrix S_n can be chosen as

$$S_n = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix}. \tag{12}$$

Step 2: After the robot has reached the plane, we rotate the end-effector until the z_d axis is perpendicular to Plane$_A$ (or $z_d // n$, which means angle (α) between z_d and n converges to zero) with

$$\ddot{x}^*_{\text{orientation}} = \Omega_m (k_p\, \theta - k_d\, \omega), \tag{13}$$

where ω is the robot angular velocity, $\theta = (0, 0, \alpha)^{\mathrm{T}}$ and α is the angle between the robot end-effector's z_d axis and the plane normal n.

These steps are demonstrated in Fig. 1(1)(d). Fig. 1(1)(b) and (c) show the simulation results, with the robot trajectory in the Cartesian space and the control parameters d, α. Both task errors converge to zero smoothly. In step 2, the position constraint is at a higher priority than the orientation. Due to the projection of the orientation task into the null-space of the position constraint, the orientation task does not affect the position task. Fig. 1(1)(d) shows that the position has not changed during the orientation task in step 2.

*2) **Case 2 – Move on a plane**:* In this case, we constrain the robot motion only on the plane with the robot end-effector perpendicular to the plane. Suppose we have a desired position X_d that does not lie on the plane, see Fig. 1(2)(d). The robot is commanded to reach the target position X_d, while staying on the plane ($x_d - y_d$ plane) and keeping the end-effector z axis parallel to normal n. The control law is a simple PD controller

$$\ddot{x}^*_{\text{position}} = \Omega_m \{k_p (X_d - X_{ef}) - k_d\, \dot{X}_{ef}\}, \tag{14}$$

where the selection matrix is

$$\Omega_m = R^d_o \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix} R^o_d. \tag{15}$$

In this task, the control of the position constraint is integrated by using the top-most priority level, while the secondary task is the orientation constraint. The orientation task is projected into the null space of the position constraint with

$$\tau = \tau_{\text{position}} + N^{\mathrm{T}}_{\text{position}}\, \tau_{\text{orientation}}, \tag{16}$$

where N_{position} is the constraint null-space matrix.

Fig. 1(2) shows the results for this test. During the task execution, the distance d and the angle α remain zero since the robot is constrained to move on the plane and keeps the orientation of the end-effector (Fig. 1(2)(c). In this test, since the task position X_d is out of the plane, the robot moves on the plane linearly until he reaches the projection point p_d, which is the projection point of the task position X_d, see Fig. 1(2)(b) and (d).

B. Position control under Joint-Limit Constraints

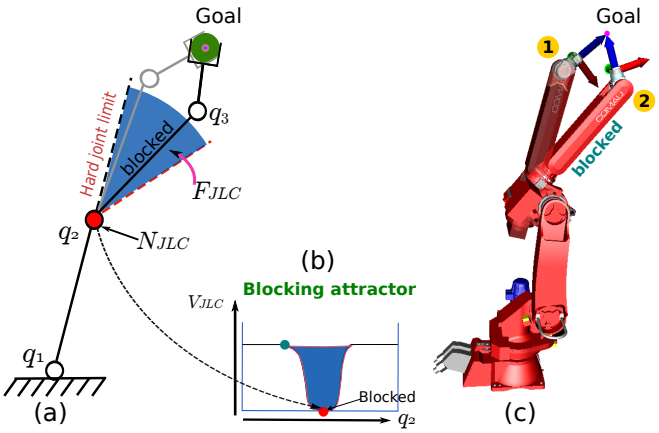

Fig. 2. End-effector position control under joint limit constraints: The robot's end-effector has been commanded to move toward a desired goal. The blue area defines a joint limit activation zone for the elbow joint. When this area is reached, a control approach is implemented to block the elbow joint while pursuing the goal (a). overview image. (b) depicts the attractor potential used to block the elbow joint inside the activation area. (c) shows the experiment results.

For industrial robotics, safety is a very important aspect. To guarantee the safety of the robot and its environment, safety related constraints (i.e., joint-limits, self-collisions) should always be guaranteed, and operational tasks should be accomplished without violating the acting constraints. Our approach handles joint-limit constraints as priority processes and executes operational tasks in the null space of joint-limits constraints.

To illustrate our approach, let us consider the control example shown in Fig. 2, where the robot's end-effector is commanded to move toward a target point X_d, while the controller handles joint-limit constraints. When no constraints are active, the end-effector is controlled using operational space control with

$$\tau = J_{\text{task}}^{\text{T}} F_{\text{task}}\,, \tag{17}$$

where τ is the vector of actuation torques, F_{task} is a control force to move the end-effector toward the desired goal, and J_{task} is the end-effector's Jacobian matrix.

When the elbow joint enters the activation zone (shown in blue), we project the task in the constraint-consistent motion manifold, decoupling the task from the constraint. At the same time, an artificial attraction potential is implemented to prevent the elbow from penetrating further into the activation area. The simultaneous control of constraints and operational tasks is expressed as

$$\tau = J_{\text{JLC}}^{\text{T}} F_{\text{JLC}} + N_{\text{JLC}}^{\text{T}} J_{\text{task}}^{\text{T}} F_{\text{task}}\,, \tag{18}$$

where N_{JLC} is the dynamically-consistent null space matrix of the constraint Jacobian, F_{JLC} is the vector of blocking forces (in the example a 1D joint space torque), J_{JLC} is the Jacobian of the violating joint (in the example it would be a constant matrix with zeros in non-violating joints and a 1 for the elbow joint). $\ddot{x}^*$ is controlled through a simple PD controller that includes velocity saturation. More details about the controller can be found in Sentis' dissertation [16].

The control of joint-limit constraints is integrated by using the top-most priority level as specified in (18), while the operational task is projected into the constraint null-space. Fig. 2 illustrates the results of the control. When the elbow joint enters the constraint activation area shown in Fig. 2(a), we apply blocking forces (F_{JLC}) to stop the elbow joint inside the activation area. To lock the joint, we use attraction fields as shown in Fig. 2(b). If there are no joint-limit constraints, the robot joints will be violated during motion, as can be seen in Fig. 2(c).

C. Hybrid control

In real applications, an accurate model of the environment is difficult to obtain, e.g., the Plane$_A$ in the *move to a plane* case cannot be known as model $(point, normal)$. Therefore, we require a force sensor and a dynamic contact to decide where the plane is located and what its normal direction n is. In this case, we use hybrid control to accomplish the task *move to a plane*.

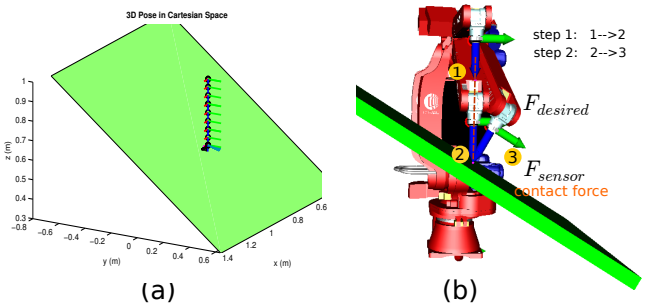

Fig. 3. Hybrid control: (a) 3D pose for the robot end-effector. (b) Two steps for the motion, including the desired force and the contact force from the force sensor.

Step 1: We apply a desired force F_{desired} to command the robot to move while the motion is constrained in the x and y axes. All constraints are combined together to guide the robot motion solely along the z axis (Fig. 3) until it reaches a plane, where the contact force F_{sensor} is provided by the force sensor. The control law is

$$F_{p(i)} = \Omega_f f_i^* + \Lambda_{p(i)}\{\Omega_m \ddot{x}_i^* - \dot{J}_i \dot{q} + J_i M^{-1}(h - \sum_{j=1}^{i-1} \tau_{p(j)})\}\,, \tag{19}$$

with the selection matrices as

$$\Omega_m = R_o^d \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix} R_d^o \tag{20}$$

and

$$\Omega_f = I - \Omega_m = R_o^d \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix} R_d^o\,. \tag{21}$$

The force control is defined as

$$f_i^* = k_s(F_{\text{desired}} - F_{\text{sensor}})\,, \tag{22}$$

where k_s is the contact stiffness. Finally, the robot stays on the plane when $\Omega_f f_i^* = 0$.

Step 2: After the robot reaches the plane, we rotate the end-effector until the contact force F_{sensor} only has the z axis component, which means that the z axis of the end-effector is perpendicular to the plane.

The results and motion are depicted in Fig. 3.

IV. Real-world Applications

In this section, we evaluate our approach on a selection of classical industrial robotics scenarios in real-world setups (using a 6-DOF industrial robot). We choose two typical industrial robotic applications for demonstrating and evaluating the proposed approach: grasping of cylindrical objects and welding.

A. Grasping of Cylindrical Objects

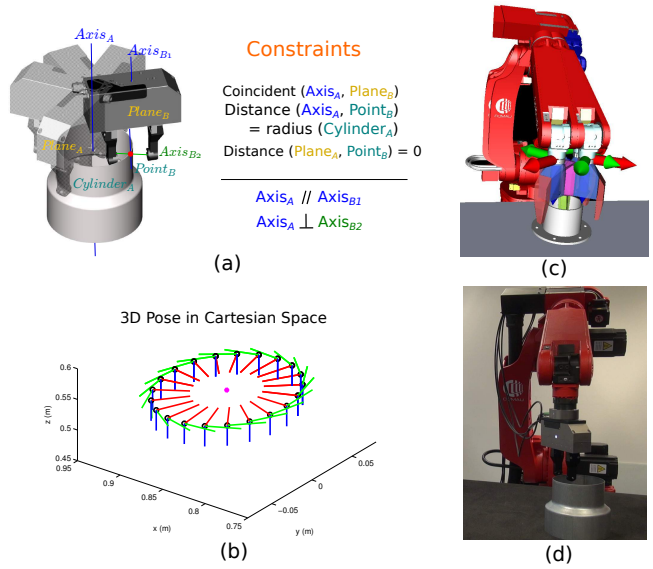

Fig. 4. Grasping of cylindrical objects at their rim in a robotic workcell: (a) Task constraints. (b) 3D pose in Cartesian space, shows the end-effector trajectory. (c) and (d) illustrate the snapshots in simulation and real experiment.

In this scenario, an industrial manipulator is supposed to grasp a cylindrical object at its rim using a parallel gripper (Fig. 4). The robot end-effector is commanded to grasp the a cylindrical object at any point along the object's rim (a valid grasping pose). The orientation of the gripper is adjusted in a way that it is tangential to the cylinder's rim.

This task can be defined with the following constraints:

- **Line-Plane Coincident Constraint:** Plane_B, which contains Axis_{B_1} and Axis_{B_2}, coincident with Axis_A.
- **Line-Point Distance Constraint:** Point_B, which is the point of intersection of Axis_{B_1} and Axis_{B_2}, at a distance (Cylinder_A) from Axis_A.
- **Plane-Point Distance Constraint:** Point_B is at a distance zero from Plane_A, which is the top plane of the object.
- **Orientation Constraint:** Axis_A is parallel to Axis_{B_1} and Axis_A is perpendicular to Axis_{B_2} of the gripper.

While the above constraints need to be fulfilled exactly, a residual degree of freedom is available as a path along the rim of the cylinder (Fig. 4(b)). We implemented this scenario in simulation and on an industrial robot platform (Fig. 4(c) and (d)), where we solve the constraints to obtain the target pose closest to the previous waypoint of the robot.

B. Welding

For the welding application, we test two cases: point welding and seam welding.

In the point welding scenario (Fig. 5(b)), the robot is supposed to weld an object at a user-specified point. This task fixes the position of the welding tool-tip. However, its orientation is not fixed and can be optimized during runtime. The tip of the welding gun must exactly coincide with the target point on the object with the position constraint:

- **Point-Point Coincident Constraint:** Point_A of workpiece is coincident with Point_B of welding gun tool (Fig. 5(a)).

In this example, the orientation of the welding gun should be adjusted by an operator.

In the seam welding scenario (Fig. 5(c)), the tip of the welding gun must lie on the target line on the object. The task constraints are

- **Point-Line Coincident Constraint:** Line_A of workpiece is coincident with Point_B of welding gun tool (Fig. 5(a)).

In this simplified experiment, we add one more constraint with constant velocities for the robot movement. This velocity constraint is applied in the null space of the position constraint, which means that the motion along the line is always satisfied. Fig. 4(d) illustrates the pose of the seam welding and the constant velocities of 0.04 m/s. Moreover, the robot can be jogged in the null-space to choose an orientation as required by other constraints from the workcell.

V. Conclusions and Future Work

In this work, we presented a prioritized, multiple-task control framework that allows for force control in industrial robotics. The proposed framework is based on a composable structure where several constraints, each describing a robot task or constraint, can be combined with a priority. The priorities are accomplished by null-space projections. We tested several types of constraints, i.e., operational task (position and orientation), force (contacts), or inequality constraints (joint limits). Moreover, two typical industrial robotics applications were chosen for demonstrating and evaluating the proposed approach.

Future work will focus on the force control applications. We are now in the process of testing the framework on typical industrial scenarios where the force control is a critical aspect, such as inserting a peg into a hole, turning a crank, or turning a screw.

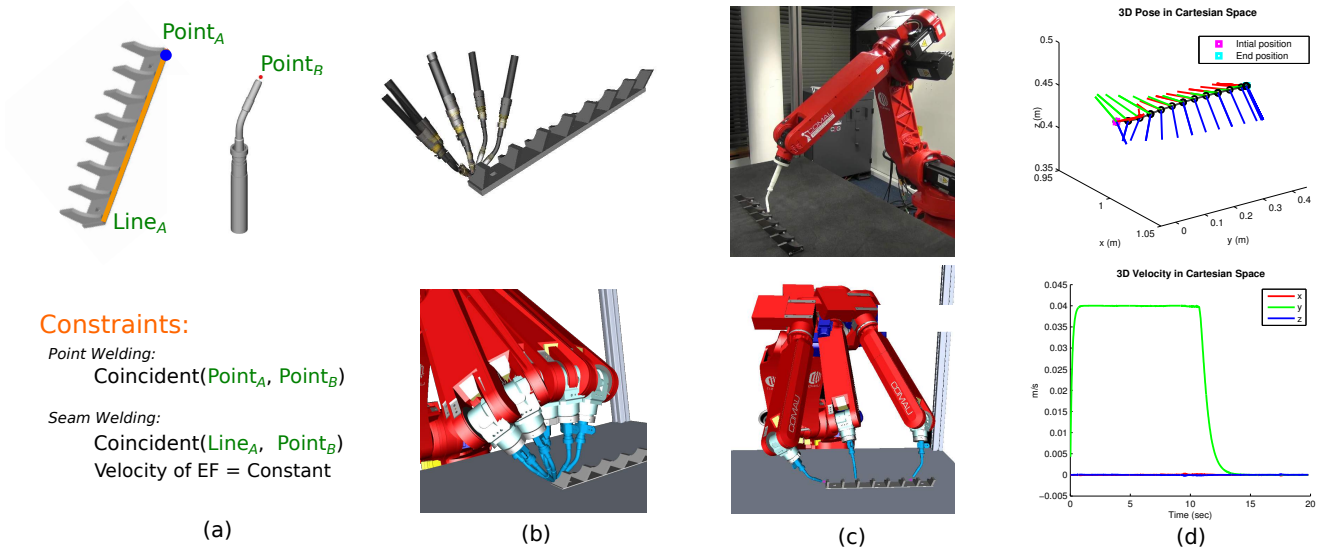

Fig. 5. Welding in a robotic workcell: (a) Task constraints, (b) point welding, (c) seam welding in simulation and real experiment, (d) 3D end-effector trajectory and velocities for robot end-effector.

REFERENCES

[1] Y. Nakamura, H. Hanafusa, and T. Yoshikawa, "Task-priority based redundancy control of robot manipulators," *The International Journal of Robotics Research*, vol. 6, no. 2, pp. 3–15, Jul. 1987.

[2] B. Siciliano and J. Slotine, "A general framework for managing multiple tasks in highly redundant robotic systems," in *Proceedings of the International Conference on Advanced Robotics (ICAR)*, Jun. 1991, pp. 1211–1216.

[3] S. Chiaverini, "Singularity-robust task-priority redundancy resolution for real-time kinematic control of robot manipulators," *IEEE Transactions on Robotics and Automation*, vol. 13, no. 3, pp. 398–410, Jun. 1997.

[4] P. Baerlocher and R. Boulic, "Task-priority formulations for the kinematic control of highly redundant articulated structures," in *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, vol. 1, Oct. 1998, pp. 323–329.

[5] R. Smits, T. De Laet, K. Claes, H. Bruyninckx, and J. De Schutter, "iTASC: A tool for multi-sensor integration in robot manipulation," in *Proceedings of the IEEE International Conference on Multisensor Fusion and Integration for Intelligent Systems*, Aug. 2008, pp. 426–433.

[6] C. Lenz, M. Rickert, G. Panin, and A. Knoll, "Constraint task-based control in industrial settings," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Oct. 2009, pp. 3058–3063.

[7] N. Somani, A. Gaschler, M. Rickert, A. Perzylo, and A. Knoll, "Constraint-based task programming with CAD semantics: From intuitive specification to real-time control," in *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Hamburg, Germany, Sep. 2015.

[8] L. Sentis and O. Khatib, "Synthesis of whole-body behaviors through hierarchical control of behavioral primitives," *International Journal of Humanoid Robotics*, vol. 2, no. 4, pp. 505–518, Dec. 2005.

[9] M. de Lasa and A. Hertzmann, "Prioritized optimization for task-space control," in *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Oct. 2009, pp. 5755–5762.

[10] L. Saab, N. Mansard, F. Keith, J.-Y. Fourquet, and P. Soueres, "Generation of dynamic motion for anthropomorphic systems under prioritized equality and inequality constraints," in *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, May 2011, pp. 1091–1096.

[11] M. Mistry and L. Righetti, "Operational space control of constrained and underactuated systems," in *Proceedings of Robotics: Science and Systems*, Los Angeles, CA, USA, Jun. 2011.

[12] J. W. Jeong and P. H. Chang, "A task-priority based framework for multiple tasks in highly redundant robots," in *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Oct. 2009, pp. 5886–5891.

[13] A. D. Prete, F. Nori, G. Metta, and L. Natale, "Prioritized motion-force control of constrained fully-actuated robots: Task space inverse dynamics," *Robotics and Autonomous Systems*, vol. 63, pp. 150–157, 2015.

[14] J. Peters, M. Mistry, F. Udwadia, J. Nakanishi, and S. Schaal, "A unifying framework for robot control with redundant dofs," *Autonomous Robots*, vol. 24, pp. 1–12, 2008.

[15] O. Khatib, "A unified approach for motion and force control of robot manipulators: The operational space formulation," *IEEE Journal of Robotics and Automation*, vol. 3, pp. 43–53, Feb. 1987.

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