A hierarchical approach to minimum-time control of industrial robots

Saed Al Homsi*, Alexander Sherikov†, Dimitar Dimitrov† and Pierre-Brice Wieber†
*Adept Technology France
saed.al-homsi@adept.com
†INRIA Rhône-Alpes, 38334 Montbonnot Cedex, France
{alexander.sherikov, dimitar.dimitrov, pierre-brice.wieber}@inria.fr

Abstract— **A novel approach to minimum-time control is presented. It is stated in terms of a hierarchical optimization problem, which is standard in the field of robotics. This is advantageous as already existing tools can be used to approach its solution. Our formulation is applied to the online generation of trajectories for industrial robots performing pick and place operations in the presence of obstacles. Model predictive control is used in order to achieve reactive system behavior and to obtain accurate local approximations of the collision avoidance constraints (which are nonconvex). Our approach has the capacity to suppress high frequency chattering in the control signal in the presence of noise: a common drawback of aggressive control strategies. Experiment using two SCARA robots that share the same working environment is used to evaluate the presented approach.**

I. Introduction

This work addresses the problem of online trajectory generation for an industrial manipulator performing pick and place operations in the presence of dynamic obstacles. Since in many mechatronic applications the control input cost is less important than the task execution time [1], we focus on fast transitions by attempting to achieve time-optimality. The user input for the proposed scheme is simply the desired endpoints without the need to specify an intermediate trajectory. This can simplify greatly deployment of industrial technology, leading to decreased cost and thus may have impact on various industrial applications [2]. Accounting for the full-body dynamics when generating this intermediate trajectory is usually not essential as most industrial robots are position controlled. That is why we model the evolution of the joint positions and velocities of the manipulator using a discrete-time linear dynamical system while accounting for input and state constraints.

Since the collision avoidance constraints are in general nonconvex, we employ a Sequential Quadratic Programming (SQP) type of approach [4] where a sequence of linearized sub-problems is solved. Each sub-problem[1] identifies a minimum-time trajectory from the current state of the robot with respect to local linear approximations of the collision avoidance constraints. While such a sequence of problems is not guaranteed to converge to a time-optimal solution for the original nonconvex problem, it provides a practical way of generating locally optimal solutions, which is sufficient for most applications [5], [6]. Due to its local nature, our approach scales well with the number of manipulators and their degrees of freedom. In particular, we avoid the "curse of dimensionality" of global approaches (which usually rely on offline computations) [7], [8]. Our approach is applied in a Model Predictive Control (MPC) setting, which not only improves reactivity of the system but presents a possibility to obtain accurate local linear approximations of the collision avoidance constraints.

[1]For clarity of presentation, and due to computational restrictions, we consider only one such sub-problem per control sampling time, even though being able to solve multiple sub-problems may lead to improved results.

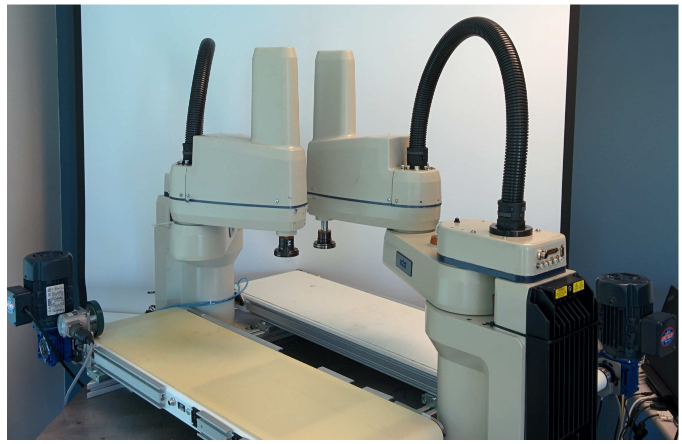


Fig. 1: Experimental setup. Two Adept Cobra SCARA [3] robots sharing the same working environment.

Our main contribution is the introduction of a hierarchical formulation [9] which guarantees time-optimal trajectories for the above mentioned SQP sub-problems. In more abstract terms, we address the problem of driving the state of an arbitrary discrete-time linear dynamical system to the origin in minimal time in the presence of linear constraints on inputs and states. Even though, in special cases, analytical solutions to this problem have been proposed [10], [11], [12], the general case that we consider here necessitates the use of numerical techniques. In contrast to other numerical approaches [13], we pose a problem that: (i) does not rely on the ad hoc selection of weighting factors (which is highly non-trivial), (ii) does not lead to any approximation and results in time-optimal behavior for arbitrary linear constraints (iii) and yet it is tractable in real-time. Our formulation hinges on recent developments of efficient hierarchical solvers in the field of robotics [14], [15] and can be integrated seamlessly in existing hierarchical control frameworks.

Apart from introducing our hierarchical formulation to minimum-time trajectory generation, we discuss practical issues related to its application. One such issue is the high frequency chattering in the control signal in the presence of noise when the setpoint has been reached [16]: a common drawback of aggressive control strategies. Following the ideas in [1] we formulate our controller in a way that leads to smooth behavior in the vicinity of the goal state.

We present an experimental evaluation of the proposed approach using a typical industrial setup where two manipulators share the same working environment (see Fig. 1). Each manipulator has its own controller and considers the other manipulator as a potential obstacle. This is a problem of practical interest and presents a very good test bed for our approach due to the limited computational resources (the underlying optimization problem for each manipulator is solved on a CPU of 400 MHz under the constraint that not more than half of the CPU power can be utilized).

The paper is organized as follows. Section II reviews a classical approach to the minimum-time control problem and introduces our hierarchical formulation. Section III includes a numerical comparison with an analytical solution in a simplified setting. Section IV discusses the effect of noise in the state estimates and how it can be ameliorated. Section V considers the nonconvex collision avoidance constraints and their linearization. Finally, Section VI presents the experimental evaluation.

II. The minimum-time problem

A. A classical formulation

We consider a discrete-time linear dynamical system

$$x_{k+1} = Ax_k + Bu_k,$$

to model the evolution of the joint angles q of a manipulator system, where $x_k \in \mathbb{R}^{n_x}$ and $u_k \in \mathbb{R}^{n_u}$ are the state variables and control inputs, respectively. The system matrices $A \in \mathbb{R}^{n_x \times n_x}$ and $B \in \mathbb{R}^{n_x \times n_u}$ could be arbitrary (however, we assume that the origin is reachable). In this paper we consider $x_k = (q_k, \dot{q}_k)$ and $u_k = \ddot{q}_k$, *i.e.,* a double integrator. Note that the use of alternative dynamical systems might be beneficial, *e.g.,* a triple integrator [12], [17]. Transferring a given initial state $x^{(c)}$ (at discrete sampling time c) to the origin in minimal time can be achieved by solving [1], [18]

$$\begin{aligned} &\text{minimize } N \\ &\text{subject to } x_{k+1} = Ax_k + Bu_k \\ &\qquad x_0 = x^{(c)} \\ &\qquad x_N = 0 \\ &\qquad u_k \in \mathcal{U}_k \\ &\qquad g(x_1, \ldots, x_N) \geq 0 \\ &\qquad N \in \{N^{\min}, \ldots, N^{\max}\}, \end{aligned} \tag{1}$$

with $k = 0, \ldots, N-1$ and $\mathcal{U}_k$ being a closed and bounded set containing zero in its interior (we assume it to be convex). The decision variables are $x_1, \ldots, x_N$, $u_0, \ldots, u_{N-1}$ and the number of discrete-sampling intervals N. Note that, by design, we are not interested in reaching the origin faster than $N^{\min}$ sampling intervals in order to avoid aggressive behavior near the origin (as we will discuss in Section IV). $g(x_1, \ldots, x_N) \geq 0$ includes collision avoidance constraints, which are in general nonconvex, as well as possibly other state related constraints, *e.g.,* joint position and velocity limits. Note that this is a mixed integer programming problem. We will use $N_c^\star$ to denote the value of N at the solution of (1) (the subscript emphasizes the dependence on $x^{(c)}$).

B. A hierarchical formulation

The approach introduced in this paper is based on an equivalent reformulation of (1) as a hierarchical optimization problem: a standard multi-objective problem, where objectives can be assigned with different levels of priority. Hierarchical formulations are popular in robotics because they ensure that objectives with lower priority are optimized as far as they do not interfere with the optimization of objectives with higher priority [19], [20].

Let us consider $N^{\max} \geq N_c^\star$, and define a sequence of states $x = (x_1, \ldots, x_{N^{\max}})$ and control inputs $u = (u_0, \ldots, u_{N^{\max}-1})$. We introduce the following hierarchical problem

$$\begin{aligned} &\underset{x,u}{\text{lex minimize}} \; v = (\|x_{N^{\max}}\|^2, \ldots, \|x_{N^{\min}}\|^2) \\ &\text{subject to } x_{k+1} = Ax_k + Bu_k \\ &\qquad x_0 = x^{(c)} \\ &\qquad u \in \mathcal{U} \\ &\qquad g(x) \geq 0, \end{aligned} \tag{2}$$

with $k = 0, \ldots, N^{\max}-1$ and $\mathcal{U}$ being a closed and bounded set containing zero in its interior. The "lex minimize" operator is standard and implies that the vector v is to be minimized according to lexicographic order [21], that is, minimizing v_i (in a least-squares sense) is infinitely more important than minimizing v_j, for $i < j$. We will use $\mathbb{P}_c$ to refer to (2) when we want to emphasize the dependence on the initial state $x^{(c)}$.

The novelty of formulation (2) is in the particular choice of lexicographic objective. It states that the most important thing, after satisfying the constraints, is to reach the origin in $N^{\max}$ number of sampling intervals. Then, if possible, try to reach the origin in $N^{\max} - 1$ sampling intervals, and so on until $N^{\min}$ intervals. This formulation ensures that each state $x_{N^{\min}}, \ldots, x_{N^{\max}}$ would be as close as possible to the origin (in Euclidean norm), and once the origin has been reached, the states $x_{N_c^\star}, \ldots, x_{N^{\max}}$ would remain there. Note that we have chosen the origin as the target state only for convenience. An arbitrary target state can be used by a simple change of variable [22]. Furthermore, if necessary, target regions can be considered by using a similar formulation.

As already discussed in the Introduction, problem (2) is nonconvex due to the collision avoidance constraints

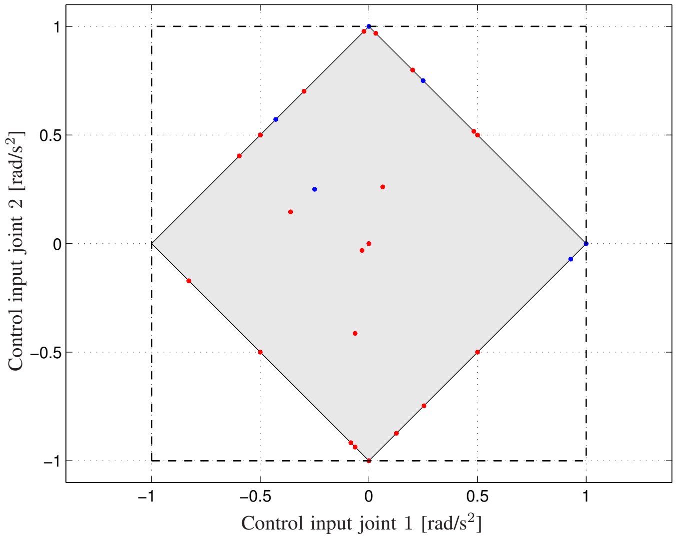


Fig. 2: Coupling constraint (diamond) and box constraints with dashed line. Blue and red dots depict the control profiles from Fig. 4.

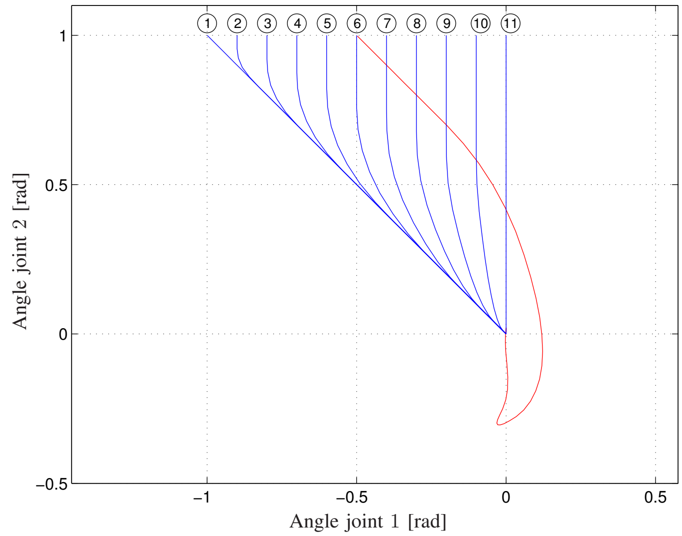


Fig. 3: Eleven joint-space trajectories. Each trajectory starts with zero joint velocity and converges to the origin.

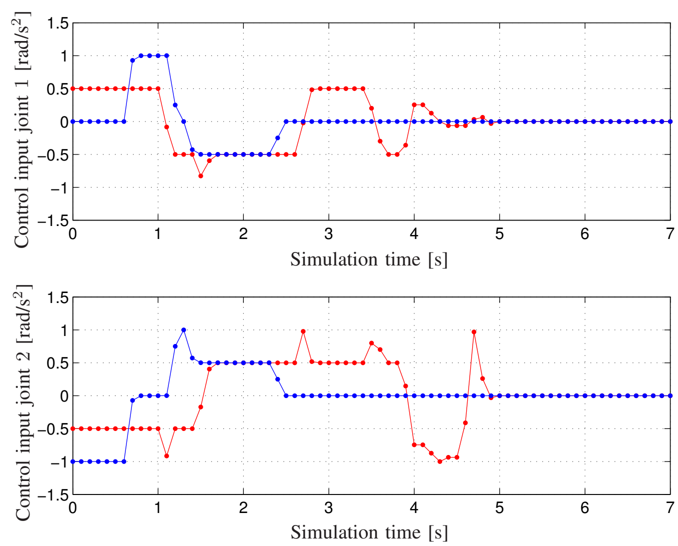


Fig. 4: Evolution of the control inputs for trajectory number 6 in Fig. 3. Blue and red correspond to formulation (4) and heuristics (5), respectively.

$g(x) \geq 0$. We approach its solution by adopting an SQP type of scheme in an MPC context. That is, problem (2) with linearized collision avoidance constraints is solved during each control sampling interval c and only $u^{(c)} = u_0^\star$ is applied to propagate the state from $x^{(c)}$ to $x^{(c+1)}$ (more details on the linearization are provided in Section V). Note that by a proper choice of $N^{\min}$ and $N^{\max}$ (which will be discussed in Section IV) one can ensure time-optimality for the linearized sub-problem. Each sub-problem can be integrated seamlessly in existing hierarchical control frameworks in robotics. Furthermore, its solution can be approached using already existing tools [15], [14], [23].

III. Comparison with the Analytical Solution for a Double Integrator

In special cases the minimum-time problem for discrete-time linear dynamical systems subject to linear constraints has an analytical solution. One such case is when using a double integrator subject to simple bounds on the accelerations [10], [11]. Here, numerical results from our hierarchical formulation are compared to this analytical solution. The purpose of this comparison is not so much to demonstrate the equivalence (which should be apparent from the analysis in Section II-B) but to emphasize the potential advantages of using numerical techniques for approaching the solution of the minimum-time problem. We consider the joint-space behavior of a two Degree of Freedom (DoF) manipulator and omit the constraint $g(x) \geq 0$.

Let the optimal policy from [11] be denoted by $u_k^\star = \pi(x_k)$. Using this policy is attractive because: (i) for any given state x_k it gives us the control actions that ensure time-optimal transition towards the goal and (ii) evaluating $\pi(x_k)$ is computationally very cheap. Assuming that $x_0 = x^{(c)}$ and $N^{\max} \geq N_c^\star$, the recursion

$$x_{k+1} = Ax_k + B\pi(x_k), \quad k = 0, \dots, N^{\max} - 1 \tag{3}$$

would reach the origin in minimal time (while taking into account the simple bounds on the controls) and remain there. The optimal sequence of control actions

$$u^\star = (u_0^\star, \dots, u_{N^{\max}-1}^\star)$$

generated from (3) coincides with the solution of

$$\begin{aligned} \underset{x,u}{\text{lex minimize}} \;\; & v \\ \text{subject to} \;\; & x_{k+1} = Ax_k + Bu_k \\ & x_0 = x^{(c)} \\ & u \in \mathcal{U}, \end{aligned} \tag{4}$$

for an appropriately chosen $\mathcal{U}$.

Note that, in the above setting, the simple bounds on u essentially decouple the joint motions. In our envisioned

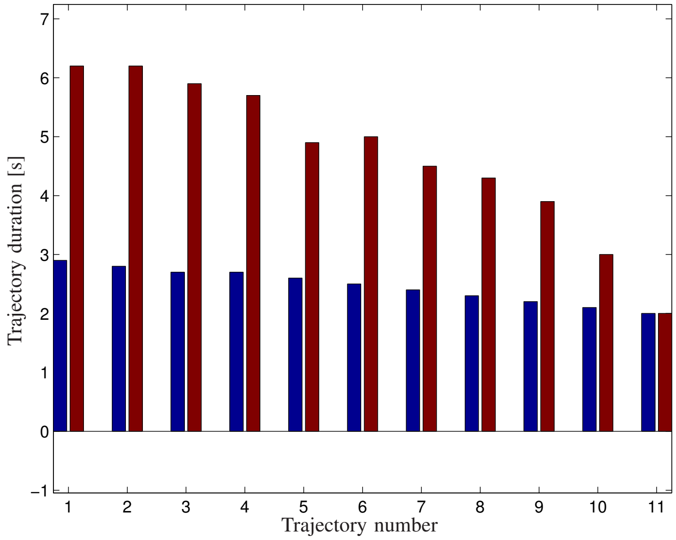


Fig. 5: Duration of each of the eleven trajectories in Fig. 3. The minimal time as computed using (4) is depicted in blue, while the time required when using the heuristics (5) is depicted in red (a more than two times difference can be observed).

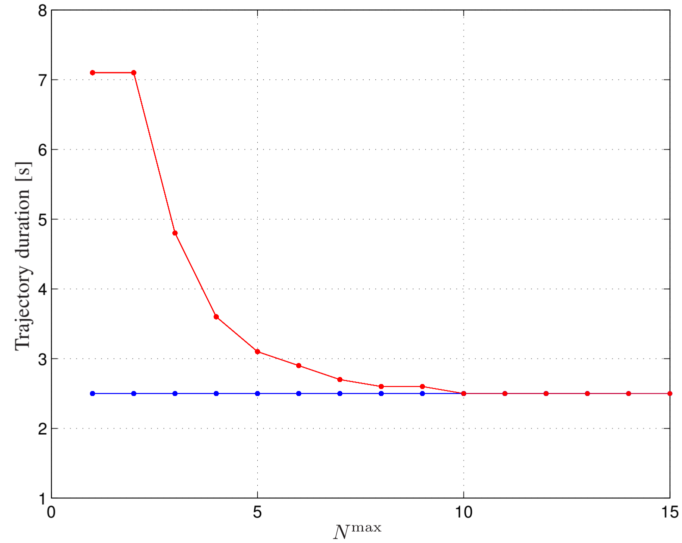


Fig. 6: Duration of trajectory number 6 in Fig. 3 for varying $N^{\max}$. Similar pattern can be observed across all trajectories. Note that $N_6^\star = 25$, hence even though in theory time-optimality is guaranteed only for $N^{\max} \geq N_6^\star$, it appears that in practice satisfactory results can be obtained with much smaller $N^{\max}$.

scenario, however, we expect to have additional linear constraints coupling the motion of the links (*e.g.,* due to the linearization of the collision avoidance constraints). One possible option for still using the analytical solution would be to find $u_k^\star$ by solving

$$\begin{aligned} &\underset{u_k}{\text{minimize}} \ \ \|u_k - \pi(x_k)\|^2 \\ &\text{subject to } u_k \in \mathcal{U}_k. \end{aligned} \tag{5}$$

The motivation behind (5) is to stay as close as possible (in Euclidean norm) to $\pi(x_k)$ while respecting the additional constraints defined by $\mathcal{U}_k$. In order to evaluate the performance of (5) we compare it to (4) on a simple example, with more restrictive constraints on u_k that couple the motion of the joints.

Figure 2 depicts these constraints as a gray diamond (contained in the box defined by the simple bounds). Figure 3 depicts in blue eleven minimum-time joint-space trajectories converging to the origin generated using (4). The effect of using heuristics (5) for the 6-th trajectory can be seen in red (the corresponding control inputs are given in Fig. 4). For all trajectories we have used $N^{\min} = 1$ and $N^{\max} = 29 \geq N_i^\star$, $i = 1, \dots, 11$ (*e.g.,* $N_6^\star = 25$), with a control sampling time $\Delta t = 0.1$ s. The duration of each trajectory is depicted in Fig. 5. As can be seen, using the heuristics (5) may result in more than twice slower transitions.

Based on these results we could conclude that even small modification of the constraints may render the analytical solution unsatisfactory. Since finding an analytical solution for arbitrary linear constraints is not straightforward it is beneficial to consider the numerical approach introduced here.

IV. Choosing $N^{\min}$ and $N^{\max}$

Formulation (2) involves the parameters $N^{\min}$ and $N^{\max}$ which should be specified by the user. The choice of $N^{\max}$ reflects the length of the preview horizon and thus can be used to influence the reactivity of the system to dynamic obstacles. If it satisfies $N^{\max} \geq \hat{N} = \max(N_1^\star, N_2^\star, \dots)$, time-optimality would be guaranteed withing each SQP subproblem. Even though $\hat{N}$ is not known beforehand, a reasonable guess for an upper bound can be made based on the particular industrial setting (*e.g.,* by considering factors like types of obstacles, sampling time, velocity and acceleration limits). Note, however, that $N^{\max}$ should not be chosen to be too large as it directly impacts the size of the problem to be solved.

Figure 6 depicts the influence of $N^{\max}$ on the duration of joint-space trajectory number 6 from Fig. 3 (for which $N_6^\star = 25$ with corresponding time of 2.5 s). As can be seen, in this particular case, time-optimality is achieved even for values considerably smaller than $N_6^\star$. Even $N^{\max} \in [7, 8, 9]$ appears to be acceptable, as the impact on the trajectory duration is rather small. We have observed that such behavior is very common even when additional state constraints are considered.

The choice of $N^{\min}$ has an impact on the behavior of (2) in the vicinity of the setpoint when state measurement noise is present. On one hand, using $N^{\min} = 1$ results in a rather aggressive controller that always attempt at reaching the setpoint in one step. In the presence of noise this would result in high frequency chattering in the control signal. On the other hand, a too high value for $N^{\min}$ might have a significant impact on the time optimal behavior. Finding

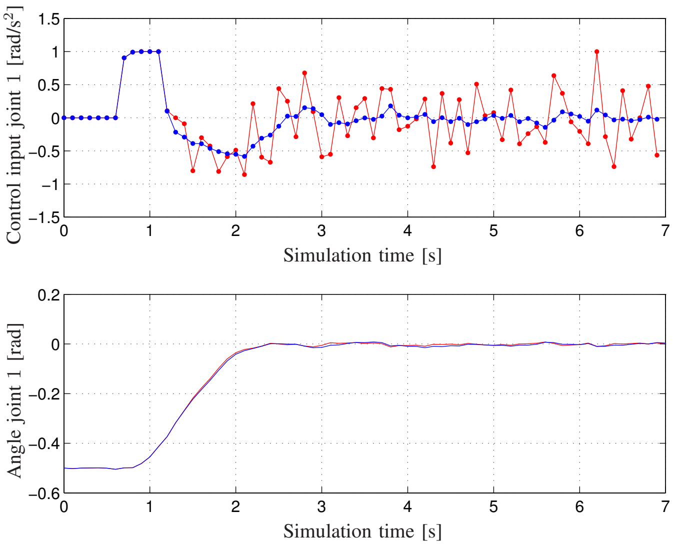


Fig. 7: Test with joint-space trajectory 6 from Fig. 3 when the state measurement is corrupted by Gaussian noise with zero mean and standard deviation 0.005. The blue and red curves represent cases with $N^{\min} = 6$ and $N^{\min} = 1$, respectively ($N^{\max} = 29$).

a proper trade-off has been considered as an important problem [16], [18].

Note that when the setpoint can be reached in m sampling intervals, using $N^{\min} > m$ leads to redundancy (the solution of (2) is not unique) which can be exploited to optimize additional criteria (that can be used to formulate a desired trade-off). This can be achieved by simply adding more hierarchical levels to (2).

Figure 7 depicts the influence of $N^{\min}$ on joint-space trajectory number 6 from Fig. 3, when the state measurement is corrupted by Gaussian noise with zero mean and standard deviation 0.005. The objective of (4) is modified to

$$\text{lex minimize } (\|x_{N^{\max}}\|^2, \ldots, \|x_{N^{\min}}\|^2, \|u\|^2),$$

i.e., an additional optimization criterion is introduced.

The blue and red curves represent cases with $N^{\min} = 6$ and $N^{\min} = 1$, respectively. The top plot illustrates the profile of the control input of joint 1. As can be seen, the minimization of $\|u\|^2$ has a filtering effect on the high frequency chattering (which is desirable in practice). The lower plot depicts the resultant profiles of the angle of joint 1: they are hardly distinguishable. This implies that a proper choice of $N^{\min}$ can have a smoothing effect on the control profiles without degrading the time-optimal behavior significantly.

In summary, the parameters $N^{\min}$ and $N^{\max}$ can be used to achieve a trade-off between time-optimality, problem size and smoothness of the solution (in the vicinity of the setpoint).

V. Collision avoidance constraints

Collision avoidance constraints $g(x) \geq 0$ can be defined in terms of various primitive shapes [24], [25]. We consider a standard model that approximates the shape of the robot and the obstacles using a composition of spheres and swept sphere lines [24]. Due to the nature of the envisioned application, the collision avoidance constraints are dynamically changing *i.e.,* not known in advance, and are moreover nonconvex. The MPC scheme that we have adopted here can be used to address both issues. Not only it increases the reactivity of the controller but also it can be used to develop accurate local linear approximations of $g(x) \geq 0$. This last point is precised next.

For clarity, first we consider collision avoidance constraints between a given link of the manipulator and a static circular obstacle. Suppose that the obstacle is centered at position $h \in \mathbb{R}^3$. Let $p_k^{(c)}$ be the point on the link that is closest to the obstacle during the k-th sampling interval of the preview associated with $\mathbb{P}_c$. Then, in order to avoid collision, the Euclidean distance between $p_k^{(c)}$ and h:

$$d_k^{(c)} = a_k^{(c)} \cdot \left(p_k^{(c)} - h\right), \quad a_k^{(c)} = \frac{p_k^{(c)} - h}{\left\|p_k^{(c)} - h\right\|},$$

must remain greater than a minimal safety distance d_s:

$$d_k^{(c)} \geq d_s. \tag{6}$$

This is a nonconvex constraint and accounting for it explicitly can be computationally costly. That is why, we approximate it by observing that $\mathbb{P}_c$ is closely related to $\mathbb{P}_{c-1}$. This fact is heavily used in the field of predictive control not only to formulate simple and expressive constraints but to warm-start each optimization process with an adequate initial guess [26]. Following the exposition in [27], we use an approximation:

$$a_k^{(c)} \approx \frac{p_{k-1}^{(c-1)} - h}{\left\|p_{k-1}^{(c-1)} - h\right\|}, \quad p_k^{(c)} \approx p_{k-1}^{(c-1)} + J_{k-1}^{(c-1)} \dot{q}_k^{(c)},$$

where $J_{k-1}^{(c-1)}$ is the Jacobian matrix associated with $p_{k-1}^{(c-1)}$. This way, the constraint (6) can be approximated using

$$a_k^{(c)} \cdot \left(J_{k-1}^{(c-1)} \dot{q}_k^{(c)} + p_{k-1}^{(c-1)}\right) \geq d_s, \tag{7}$$

which is linear in $\dot{q}_k^{(c)}$ (a part of the decision variables of $\mathbb{P}_c$). Alternatively one can use

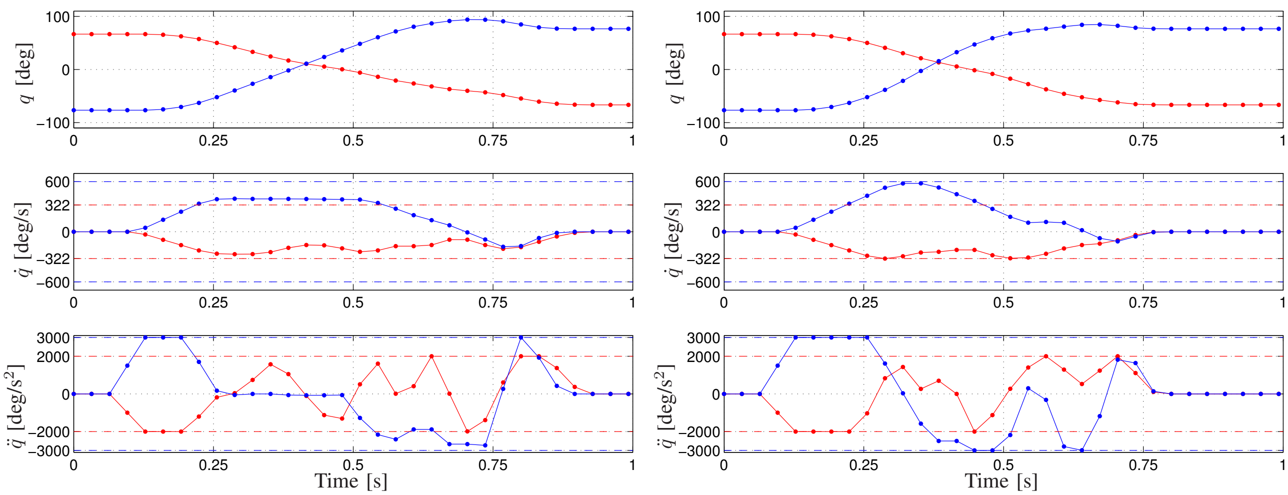
$$\dot{q}_k^{(c)} = \frac{q_k^{(c)} - q_{k-1}^{(c-1)}}{\Delta t},$$

with Δt being the sampling time, to reformulate (7) in terms of $q_k^{(c)}$. Approximating $g(x) \geq 0$ by using linear constraints like (7) for each link of the manipulator for $k = 1, \ldots, N^{\max}$, renders problem (2) with only linear constraints and a lexicographic least-squares objective, which is a class of problems commonly solved in robotics.

The only modification needed in case of a dynamic obstacle, assuming that its position over the preview horizon is known, is that one has to consider a time-varying h in the above derivations. Using other primitive shapes instead of a sphere to model obstacles is readily possible (this would only alter how the closest point is computed [28]). Note that

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

Fig. 8: Snapshots from a typical pick and place operation (a) → (b) → (c) → (d).



(a) (b)

Fig. 9: (a) Typical joint profiles of one pick and place cycle from the experiment in the video (for one of the robots) with $N^{max} = 5$. (b) A simulation result for the same endpoints as in (a), however, with $N^{\max} = 7$ (the associated snapshots are depicted in Fig. 8). Profiles of joints 1 and 2 are depicted using red and blue, respectively.

	joint 1	joint 2
angle $[deg]$	[-105, 105]	[-150, 150]
velocity $[deg/s]$	[-322, 322]	[-600, 600]
acceleration $[deg/s^2]$	[-2000, 2000]	[-3000, 3000]

TABLE I: Bounds on joint angles, velocities and accelerations.

generating collision avoidance constraints between a given manipulator link and all present obstacles is not necessary, as indicated by state of the art collision detection approaches.

VI. Experimental verification

We consider the industrial setup in Fig. 1. Two Adept Cobra s600 SCARA robots are performing pick and place operations while sharing the same working environment. Each manipulator has a dedicated controller and considers the other manipulator as a dynamic obstacle. Snapshots from a typical operation are depicted in Fig. 8.

In such industrial applications, the typical approach is for a programmer to specify intermediate paths between a large number of endpoints (for each robot). On these paths, acceleration profiles must then be defined. This process requires a lot of experience and takes considerable time and effort which prevents many manufacturers from using multi-robot systems. In contrast, the approach proposed here requires simply the desired endpoints to be specified by the user, while the intermediate trajectory is generated online.

In the experiment presented here, in addition to bounding the joint accelerations, constraints on the joint angles and velocities were imposed (see Table I). The underlying optimization problem for each manipulator was solved on a PowerPC CPU of 400 MHz under the constraint that not more than half of the CPU power can be utilized (32 ms control sampling time was used). This poses a challenge to our numerical approach (the hierarchical problems were solved using an implementation of the method in [14]).

Collision avoidance constraints were formed by considering each link of one of the manipulators (modeled using a swept sphere lines) as an obstacle for the other. We followed the linearization procedure described in Section V. Since there was no notable state estimation noise, $N^{\min} = 1$ was used. Although we were aiming at having a preview length $N^{\max} = 7$ (which was verified to lead to very satisfactory results in a simulation study), due to the hardware limitations, $N^{\max} = 5$ was considered.

Figure 9 (a) depicts typical joint profiles of one pick and place cycle from the experiment (for one of the robots).

The actual experiment can be seen in the accompanying video. The results demonstrate that online generation of fast manipulator motions with the proposed hierarchical approach is readily possible even with limited resources. Although our choice of $N^{\max}$ makes online computations feasible it, however, leads to an undesired "velocity saturation". Note how the velocity of joint 2 saturates at approximately 400 deg/s during the interval $[0.25, 0.5]$. This is a good indicator that by increasing $N^{\max}$ one can expect to achieve faster transitions. The results with $N^{\max} = 7$ (obtained in a simulation) confirm this. As can be seen on Fig 9 (b), the velocities of both joints are very close to the actual limits and, in our experience, increasing further $N^{\max}$ leads to only a marginal gain. The resultant transition duration is 30% faster compared to the case with $N^{\min} = 5$. Our current efforts are in the direction of reducing this gap by means of improving our numerical tools so that a larger $N^{\max}$ can be used or by enhancing our formulation. For example, we are investigating the effects of non-uniform sampling of the preview window and alternative warm-starting techniques.

In order to emphasize the online generation of the trajectories, the accompanying video includes a variant of the above industrial setup where the targets are moving on conveyors.

VII. Conclusion

This paper introduced a hierarchical approach to minimum-time control. It is applied in the context of online trajectory generation for industrial robots performing pick and place operations in the presence of dynamic obstacles. In particular, we presented experimental evaluation using two SCARA robots that share the same working environment. The proposed formulation simplifies greatly the deployment of industrial technology, as it does not rely on the tedious and time consuming task of manually specify paths between a large number of endpoints. We achieve a reactive behavior by using model predictive control, and our approach has the capacity to suppress high frequency chattering in the control signal in the presence of noise: a common drawback of aggressive control strategies. An important advantage of our hierarchical formulation is that the solution of the underlying problem can be approached using already existing tools in robotics.

References

[1] L. Van den Broeck, M. Diehl, and J. Swevers, "A model predictive control approach for time optimal point-to-point motion control," *Mechatronics*, vol. 21, no. 7, pp. 1203 – 1212, 2011.

[2] H. Andreasson, A. Bouguerra, M. Cirillo, D. Dimitrov, D. Driankov, L. Karlsson, and et al., "Autonomous transport vehicles: where we are and what is missing," *IEEE Robotics and Automation Magazine (IEEE-RAM)*, vol. 22, no. 1, pp. 64–75, 2015.

[3] "Omron Adept Technologies, Inc." http://www.adept.com.

[4] J. Nocedal and S. Wright, *Numerical Optimization*. Springer, 2006.

[5] D. Kim and J. Turner, "Near-minimum-time control of asymmetric rigid spacecraft using two controls," vol. 50, no. 8, pp. 2084 – 2089, 2014.

[6] H. Kim, S. Lim, C. Iuraşcu, F. Park, and Y. Cho, "A robust, discrete, near time-optimal controller for hard disk drives," *Precision Engineering*, vol. 28, no. 4, pp. 459 – 468, 2004.

[7] D. Bertsekas, *Dynamic Programming and Optimal Control*. Athena Scientific, 1995.

[8] S. Karaman and E. Frazzoli, "Sampling-based algorithms for optimal motion planning," *Int. Journal of Robotics Research*, vol. 30, no. 7, pp. 846 – 894, 2011.

[9] D. Dimitrov, P.-B. Wieber, and A. Escande, "Multi-objective control of robots," *Journal of the Robotics Society of Japan*, vol. 32, no. 6, pp. 512–518, 2014.

[10] Z. Gao, "On discrete time optimal control: a closed-form solution," in *American Control Conference*, pp. 52 – 58, 2004.

[11] R. Zanasi, C. Lo Bianco, and A. Toniellic, "Nonlinear filters for the generation of smooth trajectories," *Automatica*, vol. 36, no. 3, pp. 439 – 448, 2000.

[12] R. Zanasi and R. Morselli, "Discrete minimum time tracking problem for a chain of three integrators with bounded input," *Automatica*, vol. 39, no. 9, pp. 1643 – 1649, 2003.

[13] D. Chen, L. Bako, and S. Lecoeuche, "The minimum-time problem for discrete-time linear systems: a non-smooth optimization approach," in *IEEE International Conference on Control Applications (CCA)*, pp. 196 – 201, 2004.

[14] D. Dimitrov, A. Sherikov, and P.-B. Wieber, "Efficient resolution of potentially conflicting linear constraints in robotics," *IEEE Transactions on Robotics (under review)*.

[15] A. Escande, N. Mansard, and P.-B. Wieber, "Hierarchical quadratic programming: Fast online humanoid-robot motion generation," *The International Journal of Robotics Research (IJRR)*, vol. 33, no. 7, pp. 1006–1028, 2014.

[16] V. Zanotto, A. Gasparetto, A. Lanzutti, P. Boscariol, and R. Vidoni, "Experimental validation of minimum time-jerk algorithms for industrial robots," *Journal of Intelligent & Robotic Systems*, vol. 64, no. 2, pp. 197 – 219, 2011.

[17] O. Gerelli and C. Bianco, "A discrete-time filter for the on-line generation of trajectories with bounded velocity, acceleration, and jerk," in *IEEE International Conference on Robotics and Automation*, pp. 3989 – 3994, 2010.

[18] F. Borrelli, A. Bemporad, and M. Morar, *Predictive Control for linear and hybrid systems*. version from June, 7, 2015.

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

[20] O. Khatib, L. Sentis, J. Park, and J. Warren, "Whole-body dynamic behavior and control of human-like robots," *International Journal of Humanoid Robotics*, vol. 1, no. 1, pp. 29–43, 2004.

[21] H. Isermann, "Linear lexicographic optimization," *Operations Research Spektrum*, vol. 4, no. 4, pp. 223–228, 1982.

[22] H. Khalil, *Nonlinear Systems*. Prentice Hall, 2002.

[23] F. Flacco, A. De Luca, and O. Khatib, "Control of redundant robots under hard joint constraints: Saturation in the null space," *IEEE Transactions on Robotics*, vol. 31, no. 3, pp. 637–654, 2015.

[24] H. Sugiura, M. Gienger, H. Janssen, and C. Goerick, "Real-time collision avoidance with whole body motion control for humanoid robots," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 2053 – 2058, 2007.

[25] F. Flacco, T. Kroger, A. De Luca, and O. Khatib, "A depth space approach to human-robot collision avoidance," in *IEEE International Conference on Robotics and Automation (ICRA)*, pp. 338 – 345, 2012.

[26] Y. Wang and S. Boyd, "Fast model predictive control using online optimization," *IEEE Transactions on Control Systems Technology*, vol. 18, no. 2, pp. 267 – 278, 2010.

[27] J. Schulman, Y. Duan, J. Ho, A. Lee, I. Awwal, H. Bradlow, J. Pan, S. Patil, K. Goldberg, and P. Abbeel, "Motion planning with sequential convex optimization and convex collision checking," *The International Journal of Robotics Research (IJRR)*, vol. 33, no. 9, pp. 1251 – 1270, 2014.

[28] C. Ericson, *Real-Time Collision Detection*. Elsevier, 2004.