The 2009 IEEE/RSJ International Conference on
Intelligent Robots and Systems
October 11-15, 2009 St. Louis, USA

Optimal Path Planning in the Workspace for Articulated Robots using Mixed Integer Programming

Hao Ding[†*], Mingxiang Zhou[*], and Olaf Stursberg[†]

Abstract— **This paper considers the task of path planning for articulated robots such that the end effector is driven optimally between two points in the workspace while collision with dynamic obstacles is avoided. Compared to path planning in the configuration space, approaches in the workspace save the computationally expensive step of mapping obstacles from the workspace into the configuration space. The method presented here builds on a problem formulation as a mixed-integer program considering time-varying constraints resulting from moving obstacles, as well as state and input constraints depending on the region of the work space. The method is applied to a two-link robot with static and moving obstacles and is evaluated for different situations.**

I. INTRODUCTION

The task of path planning for articulated robots means to find a feasible path from an initial to a goal configuration of the robot such that no collision with obstacles occurs. The latter may be considered to be fixed in their position or to have an own dynamic behavior. In addition to feasibility of path generation, the property of performance becomes important if, e.g., the time or usage of resources required to reach the goal configuration has to be minimized – however, optimizing performance in path and trajectory planning of articulated robots is much less investigated as the case where only the determination of feasible paths is to be achieved. For the latter case, several algorithms and approaches have been developed in the last decades, see e.g. [3], [16] for recent overviews of corresponding techniques. An important class are potential field methods which combine repulsive potentials of obstacles (to account for collision avoidance) with an attractive potential of the goal, see [8] for an early treatment of this idea. Using randomized methods for escaping from local equilibrium states, the robot can in many cases be driven to the goal without collision, however, obtaining a feasible solution is not guaranteed. (For techniques which certainly provide a feasible path if one exists, we use the term *complete method*, according to [5]). In addition, an optimal or high performance of the motion (in the above sense) is not considered.

In order to reduce the complexity of the planning problem and abstract it in a unified way, the configuration space (C-space) is often introduced, the dimension of which is equal to the number of degrees of freedom (DOF) of the robot. The planning problem for the whole robot body in the workspace can be reduced to path planning for a single point (the configuration) in the C-space. Several methods for path planning in this space have been suggested before, like e.g. the exact and approximate cell decomposition methods presented in [10]. The difficulty of planning in the C-space lies in the obstacle representation, in particular for robots with many DOF and for dynamic obstacles. As a remedy to this drawback, sampling-based algorithms have been developed, as rapidly-exploring random trees (RRT) in [9] and probabilistic roadmaps (PRM) in [7]. By random sampling in the C-space, a tree or a roadmap can be built for determining a connection of the start and the goal configuration (two points in the C-space). The validation of the samples for collision avoidance is checked by mapping the sample into the workspace using forward kinematics. Based on the tree or the roadmap, search algorithms like A* [14] or D* [17] can be used to find the path. However, the completeness of the sampling-based algorithms can only be guaranteed in a probabilistic sense [1], i.e. the methods are guaranteed to find a feasible solution (if one exists) only for the limit of infinitely many samples. Moreover, the path is not optimized with respect to a given performance criterion like minimum energy or time.

Therefore, Blackmore and Williams developed a complete algorithm for planning paths of robotic manipulators directly in the workspace [2]. The method extends the formulation for planning of mobile robots in [12], [13], [15] to articulated robots. The optimal trajectory is computed by disjunctive programming, taking into account the kinematic, dynamic, and obstacle constraints. The optimization with regard to, e.g., the minimization of the energy consumption is considered and the completeness of the method can be guaranteed. Furthermore, the costly mapping of obstacles from the workspace to the C-space is avoided. A lower-level controller, e.g. a PID controller employing computed torque principles, is used to track reference trajectories for the velocities and positions of each joint, where the trajectories represent the series of points obtained in the path planning step.

This paper exploits and extends the method presented in [2] in the following aspects:

- The approach presented here is extended to the case of varying constraints, i.e. state and input constraints differing between different regions of the workspace and, in particular, time-varying state constraints resulting from the requirement of collision avoidance with

[†] Hao Ding and Olaf Stursberg are with the Institute of Control and System Theory, Dept. of Electrical Eng. and Computer Science, University of Kassel, Germany. (hao.ding@tum.de, stursberg@uni-kassel.de)

[*] Mingxiang Zhou was with the Institute of Automatic Control Engineering, TU München, Germany.

dynamic obstacles[1]. The basic technique proposed for this task is mixed integer programming.
- The method is numerically evaluated with respect to the computational time for (a) varying numbers of the particles introduced for the robot links and (b) for different discrete time steps required to move the end effector into the goal. This investigation is carried out for the example of a two-link robot with static and moving obstacles.

This paper is organized as follows: the transformation of the path planning task into problems of mixed integer programming is described in Sec. II for the cases of static and dynamic obstacles. Section III contains the numerical results for the application to a two-link robot for different configurations, followed by a discussion and an outlook on future work in Sec. IV.

II. PATH PLANNING IN THE WORKSPACE USING MIXED INTEGER PROGRAMMING

The task of path planning considered in this paper can be summarized as follows: for optimizing a given performance criterion, like minimization of energy consumption or transition time, determine the best way of driving an articulated robot from a start configuration into a specified goal configuration while all relevant kinematic and dynamic constraints in the workspace are satisfied, including in particular the avoidance of collision with static and dynamic obstacles operating in the same space as the robot.

A. Collision Avoidance Formulation

To formulate the obstacle avoidance, we choose a scheme similar to the one presented in [2]. The formulation is first given for the case of a single point on a robot link and then extended to articulated robots with a set of points (particles) on the links and joints of the robot.

1) Collision Avoidance for a Single Point: A polyhedral obstacle can be regarded as the intersection of N halfspaces with outward normal vectors $c_1, \ldots, c_N$, $c_j \in \mathbb{R}^{1\times n}$ and position scalars $d_1, \ldots, d_N$ with $d_j \in \mathbb{R}$. Let x denote a point in the Euclidean workspace $X \subset \mathbb{R}^n$ with dimension n. The open region of X filled by the obstacle (possibly including a safety area surrounding it) can be expressed by:

$$\mathcal{P} = \{x \mid C \in \mathbb{R}^{N\times n}, d \in \mathbb{R}^{N\times 1} : C \cdot x < d\}. \quad (1)$$

To guarantee that a point lies outside the polyhedron at least one of the N inequalities must be violated. The sufficient condition to ensure a point outside the polyhedron can be expressed using the 'Big-M' method according to [19]:

$$C \cdot x \geq d + (b - \mathbf{1}) \cdot M, \quad (2)$$

where $b = (b_1, \ldots, b_N)^{\mathrm{T}}$ is a vector of binary variables $b_j \in \{0, 1\}$, $\mathbf{1} = 1^{N\times 1}$ a vector of ones, and M a large constant, e.g. ∞. An equality contained in (2) is relaxed when the corresponding b_j has the value zero and is enforced for $b_j = 1$. To achieve that a point x must lie outside the polyhedron specified by (1), at least one among the N inequalities must be satisfied leading to the following condition:

$$\sum_{j=1}^{N} b_j \geq 1. \quad (3)$$

If the formulation according to (2) and (3) is considered as a set of constraints for an optimization problem, it is enforced that the optimizer never leads to a state x (e.g. a joint position) which is contained in the region occupied by the obstacle. If more than one obstacle has to be considered, the same formulation with corresponding matrices C and d is introduced and considered in conjunction for the optimization.

2) Collision Avoidance for Manipulators: To appropriately describe the planning task for articulated robots, the extension of the robot body has to be represented in X. Selected points of the robot geometry (*particles*) are chosen for this purpose. Assuming that the links form straight lines between adjacent joints, the particles must satisfy the following equalities:

$$x_{sl}^t = \lambda \cdot x_q^t + (1 - \lambda) \cdot x_{q+1}^t, \quad (4)$$

where x_{sl}^t indicates the position of a point with index s on a link l of the robot arm at time t. The link with index l connects the two joints indexed by q and $q + 1$. For the indices applies that $l \in \{1, \ldots, L\}$ for a total number of L links and $q \in \{0, 1, \ldots, Q - 1\}$ for a number of $Q = L + 1$ joints including the end effector. By selecting $\lambda \in [0, 1]$, the position x_{sl} of the particle is defined. The constraints to enforce that each link of the robot lies outside of an obstacle are obtained by substituting (4) for any particle into (2). The number of particles has to be chosen such that the geometry of any link is sufficiently represented in relation to the shape of the obstacle(s). This means in particular that the distance between two adjacent points on a link l must be smaller than the size of every obstacle in any direction.

B. Kinematic Constraints

Again assuming that the Q joints of the robot are connected by straight links, the kinematic constraints imply that the distance between the joints q and $q + 1$ is equal to the length $r_{q,q+1}$ of the respective link, what can be expressed by:

$$(x_{q+1}^t - x_q^t)^T \cdot (x_{q+1}^t - x_q^t) = r_{q,q+1}^2 \quad (5)$$

with $q \in \{0, \ldots, Q - 1\}$ and $t \in \{1, 2, \ldots, H\}$ where H is the number of considered discrete time points.

If this constraint is used within an optimization which requires linear constraints (as is the case for mixed integer linear programming below), the quadratic form of the equation must be replaced. This can be achieved by using the approximation of a conjunction of a circumscribing polytope and an inscribing polytope of the circle defined by (5). By enforcing that $x_{q+1}^t - x_q^t$ lies in the region

[1]Within this paper, it is assumed that the obstacle dynamics is known for the planning task, i.e. the dynamics is identified from measured data or it is communicated by the obstacle itself.

defined by the conjunction, the kinematic constraints can be imposed approximately. The fact that $x_{q+1}^t - x_q^t$ lies inside the circumscribing polytope is expressed by:

$$C_{cs} \cdot (x_{q+1}^t - x_q^t) \leq d_{cs}, \tag{6}$$

where C_{cs} and d_{cs} specify the circumscribing polytope. Likewise, $x_{q+1}^t - x_q^t$ must lie outside the inscribing polytope what is formulated by (similar to (2)):

$$C_{is} \cdot (x_{q+1}^t - x_q^t) \geq d_{is} + (b^t - \mathbf{1}) \cdot M \tag{7}$$

and

$$\sum_{j=1}^{N_{is}} b_j^t \geq 1 \tag{8}$$

with N_{is} as the number of faces of the inscribing polytope. The conjunction of (6), (7), and (8) approximates the quadratic equality constraint (5) in form of linear inequalities.

C. Varying Dynamic Constraints in Different Regions

The dynamics of the robot is considered by specifying the velocities by which the joint positions are changed. More specifically, constraints for the joint velocities are formulated to describe physical limits for the joint actuation. Furthermore, to account for the safety of operation, the limits permitted for the velocities are reduced if the robot is close to obstacles. Otherwise, it moves faster towards the goal for time or energy efficiency. This leads to the problem of varying dynamic constraints in different regions of the workspace. For illustrating corresponding formulations, the simple example of varying velocity limits in two different regions is described here. The workspace is divided into two halfspaces by a hyperplane and different allowable maximal and minimal velocities are assigned to each region.

The halfspace $c \cdot x = d$ with $c \in \mathbb{R}^{1\times 2}$, $d \in \mathbb{R}$ partitions the workspace into two regions $R_1 = \{x \mid c \cdot x < d\}$ and $R_2 = \{x \mid c \cdot x \geq d\}$. To specify different velocity limits for the two halfspaces, binary variables b_1^t and b_2^t are introduced, and the constraints are formulated by the following two groups of inequalities:

$$\begin{aligned} c \cdot x_q^t &< d + b_1^t \cdot M \\ x_q^{t+1} - x_q^t &\leq V_{q,max1} \cdot \Delta t + b_1^t \cdot M \\ x_q^{t+1} - x_q^t &\geq V_{q,min1} \cdot \Delta t - b_1^t \cdot M \end{aligned} \tag{9}$$

$$\begin{aligned} c \cdot x_q^t &\geq d - b_2^t \cdot M \\ x_q^{t+1} - x_q^t &\leq V_{q,max2} \cdot \Delta t + b_2^t \cdot M \\ x_q^{t+1} - x_q^t &\geq V_{q,min2} \cdot \Delta t - b_2^t \cdot M \end{aligned} \tag{10}$$

where $[V_{q,min1}\ V_{q,max1}]$ and $[V_{q,min2}\ V_{q,max2}]$ are the permitted ranges of velocities of the joint q in region 1 and region 2, respectively, and Δt is the time interval.

If $b_j^t = 0$ for $j \in \{1, 2\}$, the corresponding group of inequalities is enforced, otherwise it is relaxed. At any time t, only one of the two groups of inequalities should be enforced, i.e.:

$$b_1^t + b_2^t = 1. \tag{11}$$

In case of varying constraints in more than two different regions, more binary variables have to be introduced.

D. Robot Interaction with Moving Obstacles

With little modification, the above formulation can cope also with moving obstacles. Assume that the dynamics of the moving obstacles is known a-priori. The corresponding characteristic parameters C^t and d^t of a moving obstacle become time-varying and have to be updated in any time step t based on the dynamics of the obstacle.

For particles chosen according to (4), the condition for obstacle avoidance can then be expressed as:

$$C^t \cdot x_{sl}^t \geq d^t + (b^t - 1) \cdot M \tag{12}$$

with binary variables defined for any t.

In addition, the following condition has to be fulfilled in each time step t (equivalent to (3)):

$$\sum_{j=1}^{N} b_j^t \geq 1. \tag{13}$$

E. Optimization

Different criteria for formulating the performance of the robot motion can be considered and encoded in linear, quadratic, or other arbitrary form. Here, the minimization of the average kinetic energy of the robot and the minimization of the time for reaching the goal, respectively, are used.

1) Minimization of the Average Kinetic Energy: Average kinetic energy of a robot arm over time can be expressed as a quadratic function as follows: let x_{cl}^t denote the position of the mass center of the link l at time t and let m_l denote the mass of the link l. Then the velocity of the mass center of the link l can be expressed as:

$$v_{cl}^t = \frac{x_{cl}^t - x_{cl}^{t-1}}{\Delta t}. \tag{14}$$

Hence, the average kinetic energy of the robot over the planning horizon is:

$$J = \frac{1}{H} \sum_{t=1,\ldots,H} \sum_{l=1,\ldots,L} \frac{1}{2} m_l (v_{cl}^t)^T v_{cl}^t, \tag{15}$$

where H is again the number of discrete time points considered for the path from the start to the goal.

2) Minimization of the Transition Time: An objective function for achieving time optimality can be formulated as a simple linear function. Before constructing the cost function, a binary variable a^t is introduced to indicate the first time step at which the goal is reached. The value of a^t is defined as[2]:

$$a^t = \begin{cases} 1 & x_q^t \in G_q \wedge x_q^{t-1} \notin G_q \\ 0 & else \end{cases}$$

[2]Assume that the horizon H (number of time steps from the start to the goal) is chosen sufficiently large such that the goal can be reached in H.

where $t = \{1, \ldots, H\}$, G_q denotes the goal region of the joint q. The above condition can be enforced by imposing the following constraints:

$$x_q^t \geq G_q^{'} - (1 - a^t) \cdot M \qquad (16)$$
$$x_q^t \leq G_q^{'} + (1 - a^t) \cdot M$$

where x_q^t denotes the position of the joint q at time t, and $G_q^{'}$ represents the goal value for the position of the joint q.

Now the specific cost function is defined as:

$$J = \sum_{t=1}^{H} a^t \cdot t, \qquad (17)$$

where its value encodes at which time step the goal is reached. By minimizing this cost function, the robot is forced to reach the goal within the minimal number of time steps. When the goal is reached the subsequent a^t must not add any contribution to the cost function, so an additional constraint on a^t is imposed:

$$\sum_{t=1}^{H} a^t = 1. \qquad (18)$$

Overall, the optimization is formulated by using (15) or (17) subject to: the obstacle avoidance constraints according to (2) and (3), the constraints for linear approximation of kinematic constraints[3] (6) and (7), the constraints for varying velocity limits in different regions (9) to (11), the constraints for avoiding the moving obstacles (12) and (13), as well as the terminal constraints (16) and (18). This optimization problem can be solved by existing tools for mixed integer linear or quadratic programming.

III. APPLICATION AND SIMULATION RESULTS

In this section, the method is tested and evaluated by the application to a two-link robot arm. Different scenarios are considered to account for the cases of static obstacles (with different velocity limits for the joint actuators and the end-effector in two different regions) and of moving obstacles. The optimality criteria of (15) and (17) are used.

A. Implementation

To model the mixed integer optimization problem, AMPL is employed, which is a widely applied optimization modeling language interfaced to a variety of solvers [4]. The parameters of the optimization and the description of the obstacles are defined in Matlab [11] and are converted into the AMPL data file through an AMPL-Matlab interface. After loading the model and data file, AMPL calls the mixed integer programming solver CPLEX [6] to perform the optimization. The result is saved as ASCII file and can be loaded in Matlab for visualization.

[3]In case of direct consideration of the quadratic constraints of (5) in the optimization, solvers for mixed integer nonlinear programs have to be used, but this option is not considered in the paper on hand.

B. Minimization of Average Kinetic Energy with Varying Velocity Limits and Static Obstacles

A scenario of the two-link robot with two static obstacles is considered here. Both obstacles are modeled as squares of size $0.05\,m \times 0.05\,m$. One is located at $[0.45,\ 0.15]^T\,m$, the other one at $[0.4,\ 0.35]^T\,m$. The transition time is discretized into $H = 10$ steps, and the duration of each step is set to $\Delta t = 0.1s$. To ensure the obstacle avoidance of the links, 10 particles are taken with equal distribution on each link, and their obstacle avoidance constraints are included in the optimization problem. The length of the two links are both $0.3\,m$ and their masses chosen to be $1\,kg$. The mass center of each link lies in its midpoint.

The base of the robot is rooted in the origin. The initial relative joint angles for the two joints are $[0,\ 0]^T\,rad$. At the end of the motion, it should reach $[\pi/2,\ \pi/4]^T\,rad$. The motion of the robot should conform to different velocity constraints in the two different regions shown in Fig. 1. In this application, the workspace is divided into two halfspaces by the plane $c \cdot x = 0$ with $c = [1, -1]$. In the halfspace $c \cdot x < 0$, the absolute values of V_{max} (velocity of the joint position) for the two joints and the end-effector are set to $[0,\ 0.7,\ 1]^T\,m/s$ and to $[0,\ 1,\ 1.5]^T\,m/s$ in the other halfspace given by $c \cdot x \geq 0$.

With respect to the optimality criterion specifying the average kinetic energy (15), the optimized path of the robot is shown in Fig. 1, where the two rectangles with solid lines mark the two obstacles and the two rectangles with dashed lines are the safety margins around the corresponding obstacles. Because of the discrete time setting used in the planning problem, the path connecting two subsequent steps may pass through the obstacle what would be safety critical. Hence, the obstacles are enlarged depending on the values of V_{max}, Δt, the number of particles, and the length of the links. The solver CPLEX for mixed integer linear programming found the solution in $133\,s$ on a 2 GHz CPU.

C. Minimization of the Transition Time and Static Obstacles

In the scenario of minimizing the transition time, namely the optimal criterion given by (17), two static obstacles are represented by the rectangles with solid lines surrounded by their safety margin (rectangles with dashed lines) as shown in Fig. 2. The optimal trajectory for this case (which does not include a variation of the velocity limits over different regions) are obtained for a time step $\Delta t = 0.1\,s$ and a planning horizon of 25 steps. The maximal velocities for the two joints and the end-effector are set to $[0, 0.4, 0.6]^T\,m/s$. The other parameters are set the same as in the previous scenario. The motion completed within 14 steps, namely $1.4\,s$.

D. Minimization of Average Kinetic Energy Considering Moving Obstacles

In this scenario, the method is applied to the case with a moving obstacle. The whole motion is discretized into $H = 20$ steps, with a constant time step interval of $\Delta t = 0.1\,s$. The initial and goal settings are the same as in the above

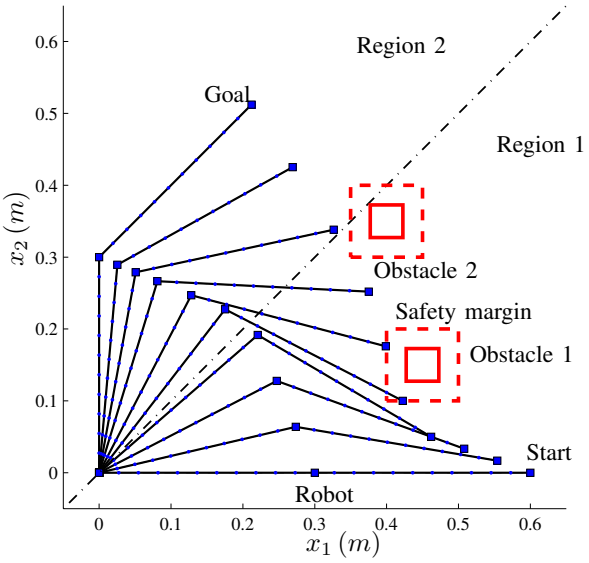

Fig. 1. Planning result of a 2-DOF robot among static obstacles for the minimization of the average kinetic energy.

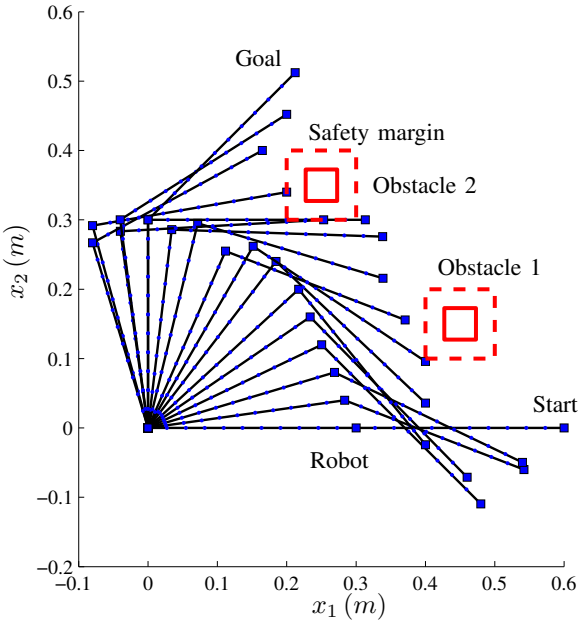

Fig. 2. Planning result of the 2-DOF robot with static obstacles for the minimization of the transition time.

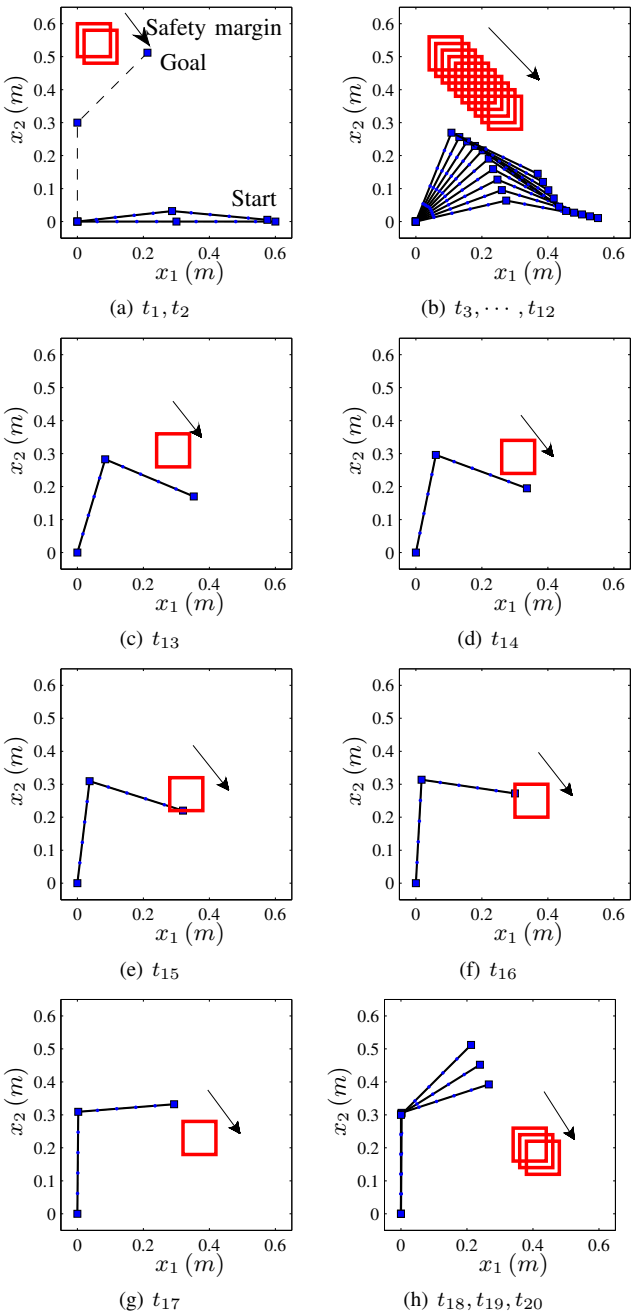

Fig. 3. Planning result of the 2-DOF robot with a moving obstacle.

case. During the motion of the robot, the safety margin of the obstacle, which is a $0.1\,m \times 0.1\,m$ square moves from the position $[0.05,\ 0.55]^T\ m$ to $[0.45,\ 0.15]^T\ m$ with uniform velocity. The maximal absolute velocities of the joints are set to $V_{max} = [0,\ 0.4,\ 0.6]^T\ m/s$ for the whole workspace. In order to save computational time, 4 equally distributed particles on each link are considered for obstacle avoidance. The other parameter settings are the same as in the first scenario.

The cost criterion is chosen as the average kinetic energy (15), and the optimal solution was found in $227\,s$. A series of plots in Fig. 3 shows the optimal path of the robot and the movement of the obstacle from the start time t_1 to the goal at t_{20}. (The shown rectangle marks again the safety area around the obstacle.)

E. Discussion

The two optimality criteria lead to significantly different optimal paths as shown in Fig. 1 and Fig. 2. For the case of minimizing the average kinetic energy, the computational times for different numbers of particles (which are equally distributed on the links) and for different discretization steps from the start to the goal are given in Tab. I. The results show that the computational time depends critically on varied

TABLE I

COMPUTATIONAL TIME (SECONDS) FOR DIFFERENT NUMBERS OF PARTICLES PER LINK AND FOR DIFFERENT NUMBERS OF DISCRETIZATION STEPS

Particles per link \ Discretization steps	10	12
2	15	23
4	52	121
6	46	77
8	332	265

parameters. In addition, the position of the particles also affect the computational effort, as can be concluded from the fact that the computational time with 4 equally distributed particles is longer than for the case with 6 equally distributed particles.

The following observations can be derived from the results in addition:

- The shape, the position, and the dynamics of the obstacles affect the computational time.
- For the optimality criterion of minimizing the average kinetic energy, the velocities of each joint are optimized with fixed overall time and discretization steps. In the case of a small range of velocity limits, the overall time and the discretized steps must be carefully selected in order to guarantee the feasibility of the problem.
- Since the method discretizes the planning problem, *jumping* behavior to circumvent a corner of the safety area are relatively often observed. This problem can be resolved by decreasing the maximal allowable velocity what increases the transition time and also the computational time.
- It can be observed that obstacles may collide with the robot in between two adjacent particles on the links if the distance between particles is chosen too large. In future work, it makes sense to develop a scheme for adaptive assignment of particles to the robot links.

IV. CONCLUSIONS AND FUTURE WORK

The path planning algorithm of articulated robots directly in the workspace for the case of static and moving obstacles, as well as with varying constraints over different regions was formulated as a constrained optimization problem with regard to different optimality criteria. The solution by mixed-integer linear or quadratic programming is complete and leads to the results which are (at least sub-)optimal with respect to the selected criterion. The application results show that the approach is suitable to compute off-line solutions for known dynamics of the obstacles.

A more ambitious goal is to identify the dynamics of the obstacles from measured data and compute the path and trajectory of the robot online – given the computation times in the experiments, the time has to be decreased still considerably to meet this goal. Thus, we currently investigate adaptive distribution of particles on the robot links as well as adaptive time steps in encoding the optimization problem. Further accelerations can be expected by using the moving horizon scheme known from model predictive control rather than optimizing at once over the complete time required to reach the goal (compare to [18]).

V. ACKNOWLEDGMENTS

This work was partially supported by the cluster of excellence *Cognition for Technical Systems* (CoTeSys), funded by the German Research Foundation (DFG) (see also www.cotesys.org).

REFERENCES

[1] J. Barraquand and J.C. Latombe, Nonholonomic multibody mobile robots: Controllability and motion planning in the presence of obstacles, *Algorithmica*, vol. 10, no. 2-4, 1991, pp. 121-155.

[2] L. Blackmore and B. Williams, "Optimal manipulator path planning with obstacles using disjunctive programming", *Proc. of the American Control Conference*, 2006, pp. 3200-3202.

[3] H. Choset, K. M. Lynch, S. Hutchinson, G. Kantor, W. Burgard, L. E. Kavraki, and S. Thrun, *Principles of robot motion*, MIT Press, 2005.

[4] R. Fourer, D.M. Gay, and B.W. Kernighan, *AMPL: A Modeling Language for Mathematical Programming*, 2nd ed., Duxbury Press, 2002.

[5] K. Goldberg, "Completeness in Robot Motion Planning", *Proc. 1st Workshop on the Algorithmic Foundations of Robotics*, 1994.

[6] ILOG, CPLEX Product Datasheet, http://www.ilog.com.

[7] L. Kavraki, P. Svestka, J. Latombe, and M. Overmars, Probabilistic roadmaps for path planning in high-dimensional configuration space, *IEEE Transactions on Robotics and Automation*, vol. 12, no. 4, 1996, pp. 566-580.

[8] O. Khatib, Real-time obstacle avoidance for manipulators and mobile robots, *Int. Journal of Robotics Research*, vol. 5, no. 1, 1986, pp. 90-98.

[9] J.J. Kuffner and S.M. LaValle, "RRT-Connect: An efficient approach to single-query path planning", *Proc. IEEE Int. Conf. on Robotics and Automation*, 2000, pp. 995-1001.

[10] J. Latombe, *Robot Motion Planning*, Kluwer, Boston, 1991.

[11] Matlab, http://www.mathworks.com/.

[12] A. Richards and J. How, "Aircraft trajectory planning with collision avoidance using mixed integer linear programming", *Proc. of the American Control Conference*, 2002, pp. 1936-1941.

[13] A. Richards and J. How, "Mixed-integer programming for control", *Proc. of the American Control Conference*, 2005, pp. 2676-2683.

[14] S.J. Russell and P. Norvig, *Artificial Intelligence: A Modern Approach*, 2nd ed., Prentice Hall, 2002.

[15] T. Schouwenaars, B. DeMoor, E. Feron, and J. How, "Mixed integer programming for multi-vehicle path planning," *European Control Conference*, 2001, pp. 2603-2608.

[16] B. Siciliano and O. Khatib, eds., *Springer Handbook of Robotics*, Springer, 2008.

[17] A. Stentz, "Optimal and efficient path planning for partially-known environments", *Proc. IEEE Int. Conf. on Robotics and Automation*, 1994, pp. 3310-3317.

[18] O. Stursberg and S. Engell, "Optimal Control of Switched Continuous Systems Using Mixed-Integer Programming", *15th IFAC World Congress*, 2002, ThA06-4.

[19] H. P. Williams, *Model building in mathematical programming*, 4th ed., WILEY, 2001.