Spline-Based Motion Planning in an Obstructed 3D Environment ⋆

Ruben Van Parys * **Goele Pipeleers** *

* *MECO research group, Department of Mechanical Engineering, KU Leuven, Belgium. (e-mail: ruben.vanparys@kuleuven.be).*

Abstract: This paper presents a motion planning approach that steers systems in an optimal way through an obstructed 3D environment. The motion trajectories are parameterized as polynomial splines and by exploiting the properties of B-spline basis functions, constraints on the trajectories are efficiently enforced. The approach is applied on two relevant cases. The first one elaborates a pick and place task for a Cartesian robot which is validated experimentally on an industrial plate transportation system. Depending on the task, the proposed method can reduce the motion time with 10 - 30% with respect to the currently applied trajectories. In a second case the approach is applied on the navigation of Unmanned Aerial Vehicles (UAVs) flying in an uncertain dynamic environment. This problem is formulated in a receding-horizon fashion which can update trajectories with a rate of 2.5 Hz. A supporting software toolbox is provided that implements the proposed approach and facilitates its use.

Keywords: Trajectory planning, obstacle avoidance, splines, Cartesian manipulators, Unmanned Aerial Vehicles

1. INTRODUCTION

Motion planning considers the problem of steering a motion system from an initial state to a terminal state as fast, energy-efficient... as possible while obeying the system limitations as well as geometric constraints. Geometric constraints can, for instance, represent workspace restrictions and collision avoidance constraints with nearby obstacles. Solving motion planning problems efficiently is key to unleashing the true potential of autonomous systems in industrial applications as well as in our daily lives.

This paper focuses on two applications where optimal motion planning in a three dimensional obstructed environment naturally comes to mind. The first one deals with the optimal control of Cartesian robots, which are omnipresent in industry. Examples are pick and place systems, 3D printers and CNC milling machines. Cartesian robots consist of three perpendicular linear axes and supply a natural way of transporting an object in a three dimensional space. As collisions with other machine parts should be avoided at all time, steering these systems is a non-trivial job. Collision avoidance is commonly achieved by using safe heuristics which introduce sub-optimality. Using a systematic approach for generating optimal collision-free trajectories would substantially increase the flexibility of these systems as well as their productivity and economic profit. The second application focuses on the autonomous navigation of Unmanned Aerial Vehicles (UAVs) in an obstructed environment. UAVs are commonly used in operations that are too dull or dangerous for humans such as surveillance, search and rescue operations or the delivery of supplies in inaccessible regions. The autonomous navigation of UAVs hinges upon the efficient solution of motion planning problems. A crucial issue in a UAV's task is the uncertain environment. It is therefore beneficial to solve the motion planning online to cope with sudden changes in the environment. Incorporating predictions concerning the motion of obstacles can further improve the UAV's navigation.

Motion planning problems naturally translate into optimal control problems. The resulting problems are however hard to solve. One challenge lies in the non-convexity of the problem coming from obstacle avoidance requirements and possible non-linear system dynamics. Due to the complexity of this problem, a decoupled approach is often adopted, which solves the motion planning problem in two stages (Shin and McKay, 1985; Bobrow et al., 1985; Verscheure et al., 2009). In a first step a geometric path is determined that accounts for obstacle avoidance and other geometric constraints. In the subsequent path following stage, an optimal trajectory along the geometric path is determined taking into account the dynamic constraints. Although this approach requires lower computational effort, it often results in sub-optimal solutions. Moreover it is not straightforward to incorporate a dynamic environment. A coupled approach that solves the optimal control problem at once, is more suited for those cases. When solving the motion planning online, an efficient formulation of this problem is however crucial in order to retrieve a small-scale problem that can be solved in a reasonable time.

⋆ This work benefits from KU Leuven-BOF PFV/10/002 Centre of Excellence: Optimization in Engineering (OPTEC), from the Belgian Programme on Interuniversity Attraction Poles, initiated by the Belgian Federal Science Policy Office (DYSCO), from the project G0C4515N of the Research Foundation-Flanders (FWO-Flanders) and the KU Leuven Research project C14/15/067: B-spline based certificates of positivity with applications in engineering. Ruben Van Parys is a PhD fellow of FWO-Flanders. The MECO Research Group is an associated research lab of Flanders Make.

Another challenge is the infinite dimensionality of the resulting problem, which holds in general for all optimal control problems. Indeed, the variables of an optimal control problem are input and state trajectories and geometric and dynamic constraints are imposed at each time instance. A final dimensional variable set is typically retrieved by parameterization of the trajectories as e.g. polynomials or splines. A classical approach for reformulating trajectory constraints in a finite way is time gridding (Louembet et al., 2009; Milam et al., 2000). However, this approach does not guarantee constraint satisfaction in between the grid points. When imposing safety critical constraints such as collision avoidance, this is unacceptable. The grid spacing should therefore be sufficiently fine to avoid constraint violations, which results in a large number of constraints.

The method used in this work is a coupled approach. It uses the idea from (Van Loock et al., 2015) for retrieving a finite dimensional small-scale problem with guaranteed constraint satisfaction by using a B-spline parameterization of the motion trajectories and an efficient enforcement of constraints on these trajectories. This paper builds on previous work presented in (Mercy et al., 2016) and demonstrates the capability of the framework when used for complex motion planning tasks in a three dimensional obstructed environment. It validates the approach experimentally on an industrial Cartesian robot and applies the method on a challenging non-linear quadrotor model. As a complement to the paper, a software toolbox is provided that implements the proposed approach and that forms a user-friendly interface for modeling, simulating and embedding spline-based motion planning problems (Van Parys and Mercy, 2016). It provides a library of different system models and contains an extensive list of illustrative examples.

Section 2 gives an overview of the spline-based motion planning approach and applies it to the optimal control of a Cartesian robot and the navigation of a UAV. Results on both cases are presented in Section 3. Finally, Section 4 draws concluding remarks.

2. SPLINE-BASED MOTION PLANNING

This section briefly recapitulates the used methodology which is adopted from early work described in (Van Loock et al., 2015). The subsequent subsections apply this approach on two three-dimensional motion planning cases: point-to-point trajectory generation for a Cartesian robot and optimal UAV navigation.

2.1 General methodology

The motion planning problem considered in this work searches for motion trajectories $q(\cdot) \in \mathbb{R}^{n_q}$, in order to steer a system from an initial condition, at $t = 0$, to a terminal condition, at $t = T$. Both conditions are expressed as conditions on q and its derivatives $q^{(j)}$. Optimal trajectories are obtained by minimizing an objective J while respecting constraints h over the considered time horizon $[0, T]$. These represent system limitations and collision avoidance constraints. This problem generally translates into an optimization problem of the following form:

$$\begin{aligned} \underset{q(\cdot)}{\text{minimize}} \quad & J(q) \\ \text{subject to} \quad & q^{(j)}(0) = q_0^{(j)}\,, \quad j \in \{0,\dots,r\}\,, \\ & q^{(j)}(T) = q_T^{(j)}\,, \quad j \in \{0,\dots,r\}\,, \\ & h(q,t) \geq 0\,, \quad \forall t \in [0,T]\,. \end{aligned} \tag{1}$$

Problem (1) is infinite dimensional, comprising both infinitely many optimization variables and constraints, as the optimization variables $q(\cdot)$ are functions and the constraints h are enforced at all time instances. To cope with the infinitely many optimization variables, the trajectories $q(\cdot)$ are approximated as piecewise polynomials and are parameterized in a B-spline basis (Boor, 2001):

$$\hat{q}(t) = \sum_{l=1}^{n} \mathsf{q}_l b_l(t)\,,$$

with B-spline basis $b = (b_1, \dots, b_n)$ and B-spline coefficients $\mathsf{q} = (\mathsf{q}_1, \dots, \mathsf{q}_n)$, which become the new optimization variables. The main reason for adopting the B-spline basis is the so-called convex hull property: as the B-splines are positive and sum up to 1, a spline is always contained in the convex hull of its B-spline coefficients. This way, bounds on a spline function can be enforced by imposing them on the coefficients:

$$\mathsf{q} \geq 0 \Rightarrow \hat{q}(t) \geq 0\,, \forall t \in [0,T]\ .$$

Because derivatives, anti-derivatives and any polynomial function of a spline are splines, also polynomial constraints on spline trajectories and its derivatives and anti-derivatives can be relaxed in the same way.

Using this approach requires finding a set of trajectories $q(\cdot)$ that characterizes the motion of a system and from which state and input trajectories can be derived. Furthermore, it should be possible to reformulate trajectory constraints as polynomial constraints in q, its derivatives and anti-derivatives. This is possible for many two-dimensional cases as indicated in (Van Loock et al., 2015; Mercy et al., 2016) and is illustrated for two three-dimensional motion planning cases below.

The presented optimal control problem (1) can be further adapted to a receding-horizon formulation in order to account for disturbances or a dynamic environment. Hence, at every sample period, an optimal control problem needs to be solved over an updated control horizon, starting from estimates of the current system state and input. As in the optimal control approach the solution of the optimization problem is formulated as a spline, care must be taken in transferring the solution of the previous iteration to a hot-start for the subsequent iteration. In (Van Parys and Pipeleers, 2016) an update scheme is presented that overcomes this problem.

2.2 Point-to-point motion for a Cartesian robot

A Cartesian robot is composed of three perpendicular independently driven linear axes that control the motion of an object in the three dimensional space. This work considers an industrial case illustrated in Figure 1. It examines the transportation of a plate by a Cartesian robot from an initial storage place (A) towards a table (B) of a punching machine. After a punching process the

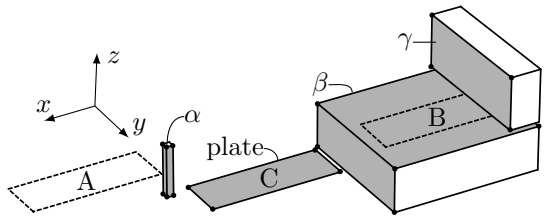


Fig. 1. Simplified representation of the plate transportation case. The plate is translated from an initial storage place (A) towards the table (B) of a punching machine. After the punching process the plate is transported to a final storage plate (C). The Cartesian robot carrying the plate is omitted in the figure for reasons of clearness.

plate is transported to a final storage place (C). The transportation should always happen as fast as possible.

The Cartesian robot can freely translate the plate in the three dimensional space and the system has therefore three degrees of freedom which are chosen as the position (x, y, z) of the plate center. An available controller can track given reference trajectories for the plate's position provided that these are sufficiently smooth and that they do not violate actuator limitations. In order to accomplish this the trajectories should be continuous up to acceleration level and kinematic constraints are provided for velocity, acceleration and jerk. In order to meet the continuity requirement, the position trajectories $q = (x, y, z)$ are parameterized as cubic splines. Kinematic constraints are applied by constraining the time derivatives of q.

The transportation of the plate happens in an obstructed environment as illustrated in Figure 1. Apart from bounds on the position q that prevent to cross machine boundaries, extra constraints should be included to avoid collisions with the obstacles α, β and γ. These are constructed by implying the existence of a separating plane between the plate and an obstacle. Note that this construction can only separate convex shapes (Boyd and Vandenberghe, 2004). When both the plate and an obstacle i are represented by convex polyhedra with vertices v_j and $w_{i,k}$, respectively, the anti-collision constraints between the plate and obstacle i are expressed as

$$\begin{aligned} a_i(t)^T v_j(q(t)) - b_i(t) &\geq r_v\,, \quad \forall j \in \{1,\dots,n_v\}\,, \\ a_i(t)^T w_{i,k} - b_i(t) &\leq -r_{w_i}\,, \quad \forall k \in \{1,\dots,n_{w_i}\}\,, \\ a_i(t)^T a_i(t) &\leq 1\,, \end{aligned} \tag{2}$$

where n_v and n_{w_i} indicate the number of vertices for plate and obstacle. The parameters r_v and r_{w_i} are positive numbers that express the minimal distance between the separating plane and respectively the plate and obstacle. These can be used as safety margins for collision avoidance. The separating plane $\{x \in \mathbb{R}^3 | a_i^T x = b_i\}$ can change over time. Both $a_i(\cdot)$ and $b_i(\cdot)$ are introduced as new time-dependent variables in the optimization problem and are also parameterized as splines. The set of constraints (2) should be introduced for each pair of plate and obstacle. These can however be simplified by only including the vertices of the obstacle's faces the plate can collide with. Because of the limited motion freedom (e.g. the plate can not tilt) and bounds on q, the plate can only collide with the faces indicated in gray on Figure 1. The vertices introduced in the anti-collision constraints are indicated by the black dots. The plate itself is modeled as a rectangular polygon where the corresponding r_v parameter expresses its thickness and an extra safety margin. Note that the Cartesian robot itself is not included in the collision avoidance as this is resolved by workspace restrictions and the collision avoidance constraints of the plate.

The generated trajectories should be time-optimal. Therefore the objective is chosen as the motion time T. In order to make the B-spline basis functions independent of the variable T, the splines are expressed with respect to a scaled time $\tau = \frac{t}{T}$ running from 0 to 1. The variable T is then introduced in constraints on the time derivatives $q^{(j)}$.

Finally, the plate transportation case is expressed as the following optimal control problem:

$$\begin{aligned} \underset{q(\cdot),a_i(\cdot),b_i(\cdot),T}{\text{minimize}} \quad & T \\ \text{subject to} \quad & q(0) = q_0\,,\ q(1) = q_T\,, \\ & q^{(j)}(0) = 0\,,\ q^{(j)}(1) = 0\,,\ \forall j \in \{1,2\}\,, \\ & T^j q^{(j)}_{\min} \leq q^{(j)}(\tau) \leq T^j q^{(j)}_{\max}\,,\ \forall j \in \{0,\dots,3\}\,, \\ & \text{constraints } (2)\,,\ \forall i \in \{\alpha,\beta,\gamma\}\,, \\ & \forall \tau \in [0,1]\,. \end{aligned}$$

All constraints are polynomial in $q(\cdot)$, $a_i(\cdot)$, $b_i(\cdot)$ and their derivatives. The spline parameterization and constraint enforcement of Section 2.1 is used to translate this to a numerical tractable problem.

2.3 Optimal UAV navigation

This case considers the optimal navigation of an Unmanned Aerial Vehicle (UAV) from an initial position towards a destination position in an obstructed dynamic environment. In order to account for disturbances, model-plant mismatch and changes in the environment, this problem is formulated in a model predictive control (MPC) fashion. The update scheme for the control horizon is adopted from (Van Parys and Pipeleers, 2016).

This work focuses on the navigation of a three dimensional quadrotor as illustrated in Figure 2. Simplified quadrotor dynamics are used, which are derived in (Beard, 2008), by fixing the yaw angle, assuming small roll and pitch angles and neglecting Coriolis forces. This quadrotor model is controlled by three inputs, the total thrust acceleration $f = \frac{F}{m}$ and the rolling and pitch torque τ_ϕ and τ_θ. The quadrotor has five degrees of freedom, the position x, y, z and the roll and pitch angle ϕ and θ. The equations of motion are

$$\begin{aligned} \ddot{x} &= f\cos\phi\sin\theta\,, \\ \ddot{y} &= -f\sin\phi\,, \\ \ddot{z} &= f\cos\phi\cos\theta - g\,, \\ \ddot{\phi} &= \frac{1}{J_x}\tau_\phi\,, \\ \ddot{\theta} &= \frac{1}{J_y}\tau_\theta\,, \end{aligned} \tag{3}$$

where g denotes the gravitational acceleration and J_x and J_y the inertia around the axes of the quadrotor's body frame.

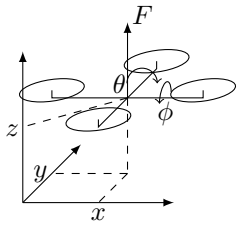


Fig. 2. Representation of a three-dimensional quadrotor.

In the considered navigation problem the quadrotor is subject to bounds on its thrust acceleration, roll and pitch angles and their derivatives:

$$\begin{aligned} &f_{\min} \le f \le f_{\max}\,, \\ &\phi_{\min} \le \phi \le \phi_{\max}\,, \quad \theta_{\min} \le \theta \le \theta_{\max}\,, \\ &\dot\phi_{\min} \le \dot\phi \le \dot\phi_{\max}\,, \quad \dot\theta_{\min} \le \dot\theta \le \dot\theta_{\max}\,. \end{aligned} \tag{4}$$

The quadrotor is steered from an initial condition towards a terminal condition which are expressed as the following equality constraints:

$$\begin{aligned} &\xi(0) = \xi_0\,, \quad \phi(0) = \phi_0\,, \quad \theta(0) = \theta_0\,, \\ &\dot\xi(0) = \dot\xi_0\,, \quad \dot\phi(0) = \dot\phi_0\,, \quad \dot\theta(0) = \dot\theta_0\,, \\ &\xi(T) = \xi_T\,, \quad \phi(T) = \phi_T\,, \quad \theta(T) = \theta_T\,, \\ &\dot\xi(T) = 0\,, \quad \dot\phi(T) = 0\,, \quad \dot\theta(T) = 0\,, \end{aligned} \tag{5}$$

where $\xi = (x, y, z)$ represents the quadrotor's position.

The objective of the navigation problem is to reach the target as fast as possible. Here an alternative for the approach of Section 2.2 is proposed which allows a fixed motion time T and therefore a reduction of the constraints' complexity. The objective is formulated as

$$J = \int_0^T \|\xi(t) - \xi_T\|_1 dt\,, \tag{6}$$

which will steer the quadrotor as close to the destination ξ_T as possible during the control horizon T. As the L^1-norm puts relatively large weight on small residuals, this objective results in a motion with low overshoot (Boyd and Vandenberghe, 2004).

As obstacles can be present in the quadrotor's airspace, collision constraints similar as (2) are imposed. In general these can depend on the quadrotor's orientation ϕ and θ. However, in order to simplify the collision constraints, the quadrotor's shape is modeled by a sphere with radius r_q and thus independent of ϕ and θ. To account for disturbances and model-plant mismatch it is desired to stay further away from obstacles than strictly necessary for avoiding collisions. Otherwise trajectory following errors due to e.g. disturbances could result in collisions. To trade-off time-optimality against the safety margin to obstacles, an extra spline variable $\delta_i(\cdot)$ is introduced and an extra term

$$\gamma_i \int_0^T \delta_i(t) dt$$

is added to the objective, while the anti-collision constraints are adapted to:

$$\begin{aligned} a_i(t)^T \xi(t) - b_i(t) &\ge r_q + d - \delta_i(t)\,, \\ a_i(t)^T w_{i,k}(t) - b_i(t) &\le -r_{w_i}\,, \quad \forall j \in \{1, \dots, n_{w_i}\}\,, \\ a_i(t)^T a_i(t) &\le 1\,, \\ 0 \le \delta_i(t) &\le d\,. \end{aligned} \tag{7}$$

This construction motivates the quadrotor to fly an extra distance d away from an obstacle. Note that the vertices $w_{i,k}$ can depend on time t. As the environment is dynamic, it is beneficial to incorporate predictions concerning an obstacle's motion. In this way an obstacle's vertices are represented as (known) splines.

In order to use the approach described in Section 2.1, a set of trajectories $q(\cdot)$ is chosen from which the states and inputs of the quadrotor can be derived and such that constraints (4) and (7) can be reformulated as polynomial constraints in q, its derivatives and anti-derivatives. This is achieved by introducing the variables $z_\phi = \tan\frac{\phi}{2}$ and $z_\theta = \tan\frac{\theta}{2}$ and using relations

$$\cos\phi = \frac{1 - z_\phi^2}{1 + z_\phi^2}\,, \quad \sin\phi = \frac{2 z_\phi}{1 + z_\phi^2}$$

and similar relations for $\cos\theta$ and $\sin\theta$. When choosing the trajectories $q = (q_1, q_2, q_3)$ as

$$q_1 = \frac{f}{(1 + z_\phi^2)(1 + z_\theta^2)}\,,\ q_2 = z_\phi\,,\ q_3 = z_\theta\,, \tag{8}$$

the first three lines of (3) are reformulated as

$$\begin{aligned} \ddot x &= 2 q_1 (1 - q_2^2) q_3\,, \\ \ddot y &= -2 q_1 q_2 (1 + q_3^2)\,, \\ \ddot z &= q_1 (1 - q_2^2)(1 - q_3^2) - g\,, \end{aligned} \tag{9}$$

and constraints (4) are translated to polynomial constraints in q and $\dot q$:

$$\begin{aligned} f_{\min} &\le q_1 (1 + q_2^2)(1 + q_3^2) \le f_{\max}\,, \\ \tan\frac{\phi_{\min}}{2} &\le q_2 \le \tan\frac{\phi_{\max}}{2}\,, \\ \tan\frac{\theta_{\min}}{2} &\le q_3 \le \tan\frac{\theta_{\max}}{2}\,, \\ (1 + q_2^2)\dot\phi_{\min} &\le 2\dot q_2 \le (1 + q_2^2)\dot\phi_{\max}\,, \\ (1 + q_3^2)\dot\theta_{\min} &\le 2\dot q_3 \le (1 + q_3^2)\dot\theta_{\max}\,. \end{aligned} \tag{10}$$

By taking the anti-derivatives of (9), expressions are retrieved for $\dot\xi$ and ξ that are polynomial in q such that collision avoidance constraints (7) become also polynomial in q. Trajectories q are parameterized as splines and spline constraints (7) and (10) are replaced by constraints on the coefficients.

The resulting optimization problem is however complex and takes a rather long time to solve. As these problems are solved repeatedly in an MPC framework, this is not desired. One main reason of the problem's complexity is the inclusion of the position ξ in the constraints (5) and (7) and objective (6) which itself is formulated as the 2nd anti-derivative of expression (9). In order to come up with a numerically more lightweight problem, trajectories $\hat\xi(\cdot)$ are introduced as slack spline variables in the problem. These are expressed in a lower dimensional basis than required for equalities (9) to hold. An approximated equality is imposed between $\hat\xi(\cdot)$ and $\xi(\cdot)$, the position derived as the 2nd anti-derivative of expression (9):

$$-\epsilon \le \hat\xi(t) - \xi(t) \le \epsilon\,, \quad \forall t \in [0, T]\,,$$

where ϵ is a small positive number. The position ξ is replaced by $\hat\xi$ in (5), (6) and (7).

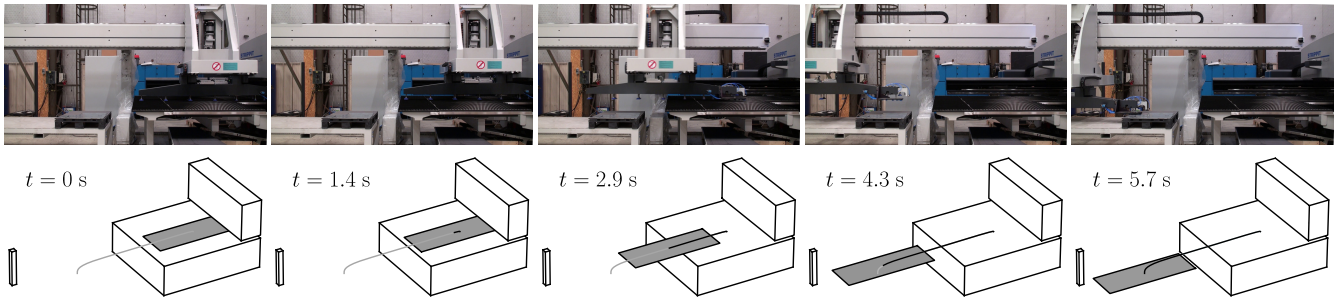


Fig. 3. Generated motion for the plate unloading task, validated on an industrial plate transportation system.

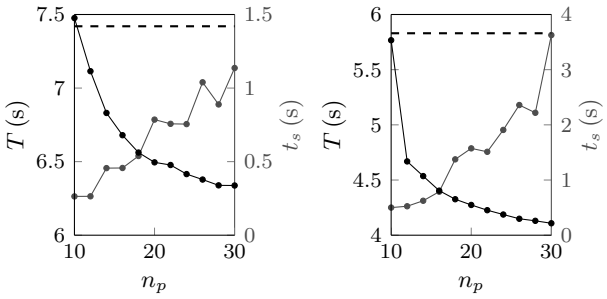
3. RESULTS

3.1 Validation on a Cartesian plate transportation system

This subsection describes the results obtained for the time-optimal plate transportation by a Cartesian robot as described in Section 2.2. Optimal trajectories are generated for two tasks: the loading of the plate in the punching machine (A to B in Figure 1) and the unloading of the machine (B to C in Figure 1). These trajectories were validated experimentally on an industrial transportation system as illustrated in Figure 3 for the loading task. A video illustrating both tasks is found on the `readme` file of the supporting toolbox (Van Parys and Mercy, 2016).

For both cases different trajectories are computed with a varying number n_p of polynomial sub-pieces for the splines. The polynomial intervals are chosen equidistant over the total motion time. A higher n_p implies a higher motion freedom and results in more time-optimal trajectories. In general this is at the expense of a higher computational cost. This trade-off is illustrated in Figure 4. The motion time T of the original trajectories is indicated in dashed line. Even with a rather coarse spline parameterization the presented approach generates more time-optimal trajectories. The reason for this lies in the heuristic approach for avoiding collisions used in the generation of the original trajectories. This is clearly visible in Figure 5, which illustrates the velocity trajectories for the loading task. The dashed curves indicate the original trajectories. In order to avoid collisions, the x axis is not moving while the y and z axis are. By incorporating collision avoidance constraints in the proposed approach, these axes can be steered simultaneous resulting in a lower total motion time. These trajectories are represented by the black solid lines

The presented method is further compared with an approach where trajectory constraints are implied by time gridding. For this comparison the Greville points are a natural choice of grid points as they result in as many constraints as the presented approach and each polynomial interval contains a Greville point. As expected this approach leads to constraint violations. Figure 5 illustrates the resulting velocity trajectories in gray. Constraint violation occurs both on the lower bound of $\dot{x}$ and upper bound on $\dot{y}$, which are represented by the dotted line. This further motivates the use of the constraint enforcement described in Section 2.1 which guaranteers constraint satisfaction over the whole time domain.



(a) Loading task (b) Unloading task

Fig. 4. Motion time T and solving time t_s for varying number of knot intervals n_p [1]. The motion time of the original trajectory is indicated in dashed line.

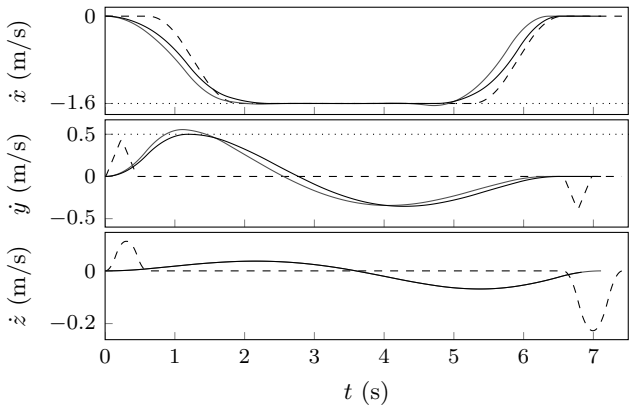


Fig. 5. Velocity trajectories for the plate loading task. The original trajectories are indicated in dashed line, while the black lines represent the time-optimal trajectories for $n_p = 12$. The gray lines are trajectories determined by time gridding of the constraints.

3.2 Quadrotor navigation in a dynamic environment

This subsection illustrates a simulation result of a quadrotor navigating in a dynamic environment. The resulting motion is illustrated in Figure 6. The quadrotor should avoid two vertical walls in order to reach its destination. At $t = 1.5$ s the right wall starts moving downwards with a velocity of 0.6 m/s. By solving the motion planning problem in receding-horizon the quadrotor is able to adapt its trajectory in order to avoid the obstacle. In the motion

[1] Optimization problems are solved with Ipopt and modeled in CasADi. All computations are performed on a notebook with Intel Core i5-4300M CPU @ 2.60 GHz x 4 processor and 8 GB of memory.

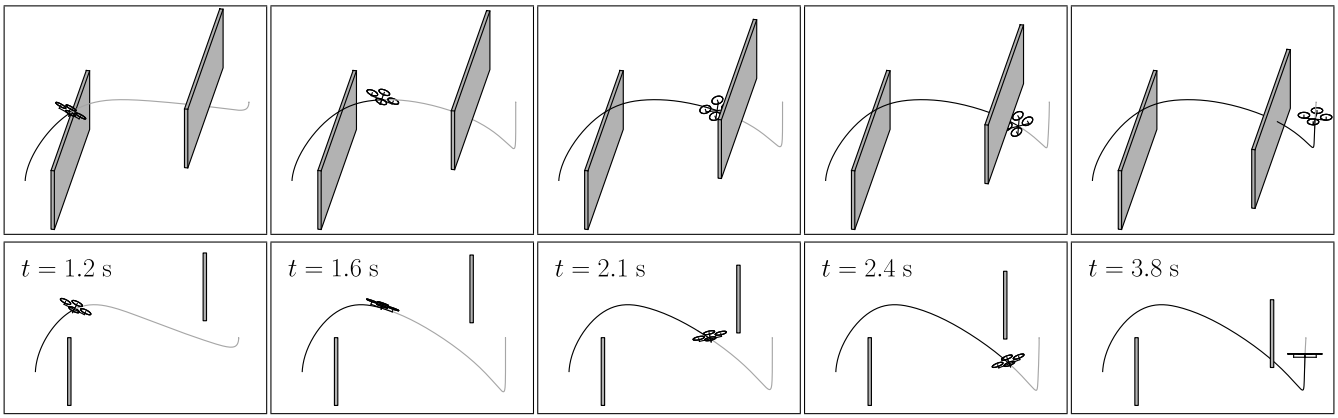


Fig. 6. Quadrotor flying in an obstructed environment. At $t = 1.5$ s the right wall starts moving downwards. The gray and black line indicate respectively the predicted and covered trajectories.

planning an exact prediction of the wall's velocity is used. The trajectories (8) are parameterized as quadratic splines with 10 polynomial intervals over a control horizon of $T = 5$ s. The average time for solving the motion planning equals 350 ms. The problem is solved repeatedly with a rate of 2.5 Hz. An approximated position $\hat{\xi}$ is introduced as a 6^{th} degree spline with bound $\epsilon = 1\,\text{mm}$. Without this approximation the average solving time equals 810 ms.

3.3 OMG-tools

The spline-based motion planning approach is implemented as part of software toolbox with the name Optimal Motion Generation-tools (Van Parys and Mercy, 2016). This toolbox is written in Python and uses CasADi (Andersson, 2013) as symbolic framework and interface to solvers. Apart from the examples presented in this paper, the toolbox provides more illustrations of the capabilities of the presented method. For a quick overview, the reader is referred to the `readme` file that contains various illustrative animations.

4. CONCLUSION AND FUTURE WORK

This paper presents a motion planning approach for efficiently steering systems through an obstructed environment. By using a spline parameterization for the motion trajectories and an efficient constraint enforcement on these trajectories, the problem is formulated as a small-scale optimization problem. This paper specifically treats two three dimensional cases. In the first case the method is used to steer an industrial Cartesian Robot time-optimal. This was validated experimentally on a plate transportation system. The second case handles the navigation of a quadrotor in a dynamic environment. In order to cope with a changing environment and with disturbances in general, the motion planning is solved repeatedly online. All code is made available via a user-friendly toolbox (Van Parys and Mercy, 2016). Future work includes the experimental validation of the quadrotor navigation and investigating approaches for further reducing the update time in an online setting.

ACKNOWLEDGEMENTS

The authors would like to thank Tim Mercy for his support during the experimental validation and Gianluca Rossetti for transferring his expertise on UAV modeling.

REFERENCES

J. Andersson. *A general-purpose software framework for dynamic optimization*. PhD thesis, PhD thesis, Arenberg Doctoral School, KU Leuven, Department of Electrical Engineering (ESAT/SCD) and Optimization in Engineering Center, Kasteelpark Arenberg 10, 3001-Heverlee, Belgium, 2013.

R. Beard. Quadrotor dynamics and control rev 0.1. 2008.

J. E. Bobrow, S. Dubowsky, and J. S. Gibson. Time-optimal control of robotic manipulators along specified paths. *The international journal of robotics research*, 4(3):3–17, 1985.

C. De Boor. A practical guide to splines, revised edition, vol. 27 of applied mathematical sciences, 2001.

S. Boyd and L. Vandenberghe. *Convex optimization*. Cambridge university press, 2004.

C. Louembet, F. Cazaurang, A. Zolghadri, C. Charbonnel, and C. Pittet. Path planning for satellite slew manoeuvres: a combined flatness and collocation-based approach. *IET control theory & applications*, 3(4):481–491, 2009.

T. Mercy, W. Van Loock, and G. Pipeleers. Real-time motion planning in the presence of moving obstacles. In *Proceedings of the 2016 European Control Conference*, pages 1586–1591, July 2016.

M. B. Milam, K. Mushambi, and R. M. Murray. A new computational approach to real-time trajectory generation for constrained mechanical systems. In *Decision and Control, 2000. Proceedings of the 39th IEEE Conference on*, volume 1, pages 845–851. IEEE, 2000.

K. Shin and N. McKay. Minimum-time control of robotic manipulators with geometric path constraints. *IEEE Transactions on Automatic Control*, 30(6):531–541, 1985.

W. Van Loock, G. Pipeleers, and J. Swevers. B-spline parameterized optimal motion trajectories for robotic systems with guaranteed constraint satisfaction. *Mechanical Sciences*, 6(2):163–171, 2015.

R. Van Parys and T. Mercy. OMG-tools. `https://github.com/meco-group/omg-tools`, 2016.

R. Van Parys and G. Pipeleers. Online distributed motion planning for multi-vehicle systems. In *Proceedings of the 2016 European Control Conference*, pages 1580–1585, July 2016.

D. Verscheure, B. Demeulenaere, J. Swevers, J. De Schutter, and M. Diehl. Time-optimal path tracking for robots: A convex optimization approach. *IEEE Transactions on Automatic Control*, 54(10):2318–2327, 2009.