Time-Optimal Trajectory Generation for Path Following with Bounded Acceleration and Velocity

Tobias Kunz and Mike Stilman
School of Interactive Computing, Center for Robotics and Intelligent Machines
Georgia Institute of Technology
Atlanta, GA 30332
Email: tobias@gatech.edu, mstilman@cc.gatech.edu

***Abstract*—This paper presents a novel method to generate the time-optimal trajectory that exactly follows a given differentiable joint-space path within given bounds on joint accelerations and velocities. We also present a path preprocessing method to make nondifferentiable paths differentiable by adding circular blends. We introduce improvements to existing work that make the algorithm more robust in the presence of numerical inaccuracies. Furthermore we validate our methods on hundreds of randomly generated test cases on simulated and real 7-DOF robot arms. Finally, we provide open source software that implements our algorithms.**

I. Introduction

To deal with the complexity of planning a robot motion, the problem is often subdivided in two or more subproblems. The first subproblem is that of planning a geometric path through the environment, which does not collide with obstacles. In an additional step this path is converted into a time-parametrized trajectory that follows the path within the capabilities of the robot. Preferably we are looking for a near-optimal trajectory according to some optimality criterion. In practice the capabilities of the robot cannot be expressed exactly. Thus, the capabilities of the robot are normally approximated by using some model of the robot and limiting certain quantities. One option is to model the dynamics of the robot and limit the torques that can be applied by the joints. In this paper we are using limits on joint accelerations and velocities which are often available.

This paper presents a method to generate the time-optimal trajectory along a given path within given bounds on accelerations and velocities. The path can be given in any arbitrary configuration space. However, we assume that the acceleration and velocity of individual coordinates are limited. Thus, the configuration coordinates need to be chosen such that they match the quantities that need to be limited. For a robot manipulator the coordinates are typically chosen to match joints of the robot. Thus, we will refer to the limits as joint acceleration and joint velocity limits. We assume that the path is differentiable with respect to some path parameter s. This is a weak assumption. If the path was not differentiable at a point, the robot would have to come to a complete stop at that point in order to follow the path. Then the path could be split at that point and the method presented here could be applied to each part of the path. We also assume that the path curvature is piecewise-continuous and that for every piece the path is coplanar.

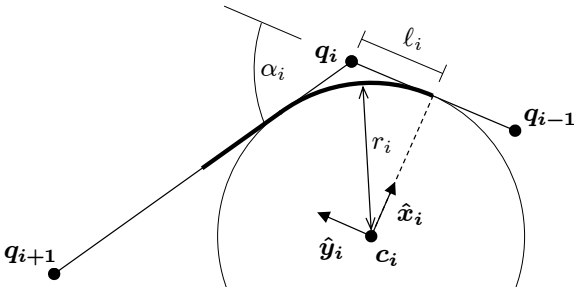


Fig. 1. Circular blend around waypoint

The output of a typical geometric path planner is a path in configuration space consisting of continuous straight line segments between waypoints. Such a path is not differentiable at the waypoints. Thus, we also present a preprocessing step to make such a path differentiable by adding circular blends.

II. Related Work

One approach described in standard textbooks [1, 2] to generate a trajectory that satisfies acceleration and velocity constraints from a list of waypoints is to use linear segments with parabolic blends. However, the approach is not directly applicable to automatically generated paths with potentially dense waypoints. They assume that the timing between the waypoints and thus the velocities for the linear segments are already known. However, a path normally does not include timing for the waypoints. Choosing the timing is not trivial. Generally we prefer to move fast, but if we move too fast, neighboring blend segments might overlap and render the trajectory invalid.

In [3] a very simple method for choosing the timing is presented. This method tries to resolve overlaps of blend segments locally by slowing down neighboring linear segments until no blend segments overlap. Yet, this method can lead to very slow trajectories, especially if the waypoints are close together.

The general approach used in this paper was introduced by Bobrow et al. [4] and by Shin and McKay [5]. We build on the later one. They propose to convert all joint constraints into constraints on the acceleration and velocity along the

path, which reduces the problem to one dimension and then search in the two-dimensional phase plane for the optimal trajectory. They consider not only acceleration constraints but use a more general model. They are able to limit torque in the presence of robot dynamics or even to limit the voltage in the presence of motor dynamics. Partially due to this generality, their derivation is more complex than ours. We are therefore able to more thoroughly consider all possible special cases. [5] falsely claims that at every point along the limit curve there is only a single feasible acceleration. It also does not consider limits on joint velocities. Finally, Shin searches for switching points solely numerically, which is less efficient and makes the algorithm more likely to fail, especially in the presence of discontinuities in the path curvature.

The following papers have built on top of this general approach. All of them assume torque constraints. A couple of papers analyse conditions for switching points in order to be able to calculate them explicitly instead of searching for them numerically.

[6] notes that points where the limit curve is non-differentiable are potential switching points and gives a necessary condition for identifying them. It also states that there is more than one valid acceleration at these switching points.

[7] lists all possible cases for switching points and gives necessary conditions for some of them that can be used to explicitly calculcate switching point candidates. However, they do not give sufficient conditions for switching points. Also, one of the cases still requires a numerical search. We will show that the case that requires a numerical search in [7] cannot happen in our case with acceleration limits instead of torque limits and a piecewise planar path. We only use a numerical search for switching points caused by joint velocity limits.

[8] deals with those switching points along the limit curve where the acceleration is not uniquely determined. However, we will show that the acceleration they claim to be optimal at such a point is incorrect.

Finally, [9] considers joint velocity constraints in addition to torque constraints.

None of the previous work deals with numerical issues or reports how robustly their algorithm performs.

III. Contribution

Our algorithms build on existing work. The novel contributions of this paper are the following:

- Combining path-following with a preprocessing step that converts the output of typical path planners to a differentiable path.
- A more thorough derivation of the constraints in the phase plane than [5]. At the same time our derivation is simpler since we only cover a special case of [5].
- We show that for the case of constraints on only acceleration and piecewise-planar paths, the limit curve is never continuous and differentiable at a switching point. Thus, we can calculate all switching points along the acceleration limit curve explicitly. Previous work relied at least partially on numerical search, but also handled the more general case of torque constraints.
- [8] applies an incorrect optimal acceleration at switching points where the limit curve is non-differentiable. We demonstrate this and provide an alternative solution.
- We provide sufficient conditions for switching points instead of just necessary ones.
- An improvement to the algorithm that makes it more robust in the presence of numerical inaccuracies.
- Our algorithm is demonstrated to be sufficiently robust to follow over 100 randomly generated paths without failing.
- We provide open-source software, available for download at http://www.golems.org/node/1570.

IV. Path Preprocessing

Common geometric path planners like PRMs or RRTs usually output the resulting path as a list of waypoints, which are connected by straight lines in configuration space. At a waypoint the path is changing its direction instantaneously and thus is not differentiable. In order to follow such a path exactly, the robot would have to come to a complete stop at every waypoint. This would make the robot motion very slow and look unnatural. Therefore, we add circular blends around the waypoints, which make the path differentiable. If the path is already differentiable, the preprocessing described in this section can be omitted.

We are looking for a circular segment that starts tangential to the linear path segment before the waypoint and ends tangential to the linear path segment after the waypoint. We also need to ensure that the circular segment does not replace more than half of each of the neighboring linear segments. Otherwise we might not get a continuous path. In addition, we allow the enforcement of a maximum deviation from the original path.

First, we define some quantities that are helpful to define the circle (see also Figure 1). The unit vector $\hat{\mathbf{y}}_\mathbf{i}$ pointing from waypoint $\mathbf{q_{i-1}}$ to $\mathbf{q_i}$ is given by

$$\hat{\boldsymbol{y}}_{\boldsymbol{i}} = \frac{\boldsymbol{q_i} - \boldsymbol{q_{i-1}}}{|\boldsymbol{q_i} - \boldsymbol{q_{i-1}}|} \tag{1}$$

The angle α_i between the two adjoining path segments of waypoint q_i is given by

$$\alpha_i = \arccos\left(\hat{\boldsymbol{y}}_{\boldsymbol{i}} \cdot \hat{\boldsymbol{y}}_{\boldsymbol{i+1}}\right) \tag{2}$$

The distance ℓ_i between waypoint $\boldsymbol{q_i}$ and the points where the circle touches the linear segments is given as

$$\ell_i = \min\left\{ \frac{||\boldsymbol{q_i} - \boldsymbol{q_{i-1}}||}{2}, \frac{||\boldsymbol{q_{i+1}} - \boldsymbol{q_i}||}{2}, \frac{\delta \sin\frac{\alpha_i}{2}}{1-\cos\frac{\alpha_i}{2}} \right\} \tag{3}$$

where the first two elements give the maximum possible distances such that the circular segment does not replace more than half of the adjoining linear segments and the last element limits the radius to make sure the circlular segment stays within a distance δ from waypoint $\boldsymbol{q_i}$.

Given the quantities above, we can now define the circular segment. The circle is defined by its center $\boldsymbol{c_i}$, its radius r_i and

two vectors $\hat{\boldsymbol{x}}_{\boldsymbol{i}}$ and $\hat{\boldsymbol{y}}_{\boldsymbol{i}}$ spanning the plane in which the circle lies. The vectors $\hat{\boldsymbol{x}}_{\boldsymbol{i}}$ and $\hat{\boldsymbol{y}}_{\boldsymbol{i}}$ are orthonormal. $\hat{\boldsymbol{x}}_{\boldsymbol{i}}$ points from the center of the circle to the point where the circle touches the preceding linear path segment. $\hat{\boldsymbol{y}}_{\boldsymbol{i}}$ is the previously defined direction of the preceding linear segment.

$$r_i = \frac{\ell_i}{\tan\frac{\alpha_i}{2}} \tag{4}$$

$$\boldsymbol{c_i} = \boldsymbol{q_i} + \frac{\hat{\boldsymbol{y}}_{\boldsymbol{i+1}} - \hat{\boldsymbol{y}}_{\boldsymbol{i}}}{||\hat{\boldsymbol{y}}_{\boldsymbol{i+1}} - \hat{\boldsymbol{y}}_{\boldsymbol{i}}||} \cdot \frac{r_i}{\cos\frac{\alpha_i}{2}} \tag{5}$$

$$\hat{\boldsymbol{x}}_{\boldsymbol{i}} = \frac{\boldsymbol{q_i} - \ell_i\hat{\boldsymbol{y}}_{\boldsymbol{i}} - \boldsymbol{c_i}}{||\boldsymbol{q_i} - \ell_i\hat{\boldsymbol{y}}_{\boldsymbol{i}} - \boldsymbol{c_i}||} \tag{6}$$

Given these quantities specifying the circle, we can calculate the robot configuration $\mathbf{q}$ for any point on the circular segment as a function $\boldsymbol{f}(s)$ of the arc length s traveled from the start of the path. As we are currently only considering the current circular path segment, we assume $s_i \leq s \leq s_i + \alpha_i r_i$, where s_i specifies the start of the circular segment. Similarly we can calculate the first and second derivatives of the function $\boldsymbol{f}(s)$. Note that these are not time-derivatives but derivatives by s. We will make use of these derivatives later.

$$\boldsymbol{q} = \boldsymbol{f}(s) = \boldsymbol{c_i} + r_i\left(\hat{\boldsymbol{x}}_{\boldsymbol{i}}\cos\left(\frac{s}{r_i}\right) + \hat{\boldsymbol{y}}_{\boldsymbol{i}}\sin\left(\frac{s}{r_i}\right)\right) \tag{7}$$

$$\boldsymbol{f'}(s) = -\hat{\boldsymbol{x}}_{\boldsymbol{i}}\sin\left(\frac{s}{r_i}\right) + \hat{\boldsymbol{y}}_{\boldsymbol{i}}\cos\left(\frac{s}{r_i}\right) \tag{8}$$

$$\boldsymbol{f''}(s) = -\frac{1}{r_i}\left(\hat{\boldsymbol{x}}_{\boldsymbol{i}}\sin\left(\frac{s}{r_i}\right) + \hat{\boldsymbol{y}}_{\boldsymbol{i}}\cos\left(\frac{s}{r_i}\right)\right) \tag{9}$$

V. Reduction to One Dimension

The approach we are using was originally proposed in [5], which finds a minimum-time trajectory that satisfies torque limits on the joints. In constrast, we assume acceleration and velocity limits on the joints. Acceleration limits are a special case of torque limits. Using acceleration instead of torque limits results in a simpler problem and a simpler derivation of the following equations than in [5]. Unlike [5] and like [9], we are also considering velocity limits on the joints.

Because the solution is constrained to follow a given path exactly, the problem can be reduced to one dimension: choosing the velocity $\dot{s} = \frac{ds}{dt}$ for every position s along the path.

A path of length s_f is given as a function $f : [0, s_f] \to \mathbb{R}^n$. The configuration q at a point s along the path is given by

$$\boldsymbol{q} = \boldsymbol{f}(s) \quad 0 \leq s \leq s_f \tag{10}$$

where s can be an arbitrary parameter. We will assume it is the arc length traveled since the start of the path. We can also define the joint velocities and accelerations with respect to the parameter s.

$$\dot{\boldsymbol{q}} = \frac{\mathrm{d}}{\mathrm{d}t}\boldsymbol{f}(s) = \frac{\mathrm{d}}{\mathrm{d}s}\boldsymbol{f}(s)\,\frac{ds}{dt} = \boldsymbol{f'}(s)\,\dot{s} \tag{11}$$

$$\ddot{\boldsymbol{q}} = \boldsymbol{f'}(s)\,\ddot{s} + \boldsymbol{f''}(s)\,\dot{s}^2 \tag{12}$$

If s is the arc length, $\dot{s}$ and $\ddot{s}$ are the velocity and acceleration along the path, then $\boldsymbol{f'}(s)$ is the unit vector tangent to the path and $\boldsymbol{f''}(s)$ is the curvature vector. [8]

The constraints in the high-dimensional joint space need to be converted into constraints on the scalar path velocity $\dot{s}(s)$ and path acceleration $\ddot{s}(s, \dot{s})$. The constraints on joint accelerations result in constraints on the acceleration and velocity along the path as shown in Section V-A. The constraints on joint velocities result in constraints on the velocity along the path as presented in Section V-B.

A. Joint Acceleration Limits

We have constraints on the joint accelerations given as

$$-\ddot{q}_i^{\max} \leq \ddot{q}_i \leq \ddot{q}_i^{\max} \qquad \forall i \in [0, ..., n] \tag{13}$$

where $\ddot{q}_i$ is the ith component of vector $\ddot{q}$. Although the universal quantifier is ommited in the following, all the following inequalities have to hold for all $i \in [0, ..., n]$.

$$(13) \Leftrightarrow -\ddot{q}_i^{\max} \leq f_i'(s)\,\ddot{s} + f_i''(s)\,\dot{s}^2 \leq \ddot{q}_i^{\max} \tag{14}$$

If $f_i'(s) > 0$:

$$(14) \Leftrightarrow \frac{-\ddot{q}_i^{\max}}{f_i'(s)} - \frac{f_i''(s)\,\dot{s}^2}{f_i'(s)} \leq \ddot{s} \leq \frac{\ddot{q}_i^{\max}}{f_i'(s)} - \frac{f_i''(s)\,\dot{s}^2}{f_i'(s)} \tag{15}$$

$$\Leftrightarrow \frac{-\ddot{q}_i^{\max}}{|f_i'(s)|} - \frac{f_i''(s)\,\dot{s}^2}{f_i'(s)} \leq \ddot{s} \leq \frac{\ddot{q}_i^{\max}}{|f_i'(s)|} - \frac{f_i''(s)\,\dot{s}^2}{f_i'(s)} \tag{16}$$

If $f_i'(s) < 0$:

$$(14) \Leftrightarrow \frac{-\ddot{q}_i^{\max}}{f_i'(s)} - \frac{f_i''(s)\,\dot{s}^2}{f_i'(s)} \geq \ddot{s} \geq \frac{\ddot{q}_i^{\max}}{f_i'(s)} - \frac{f_i''(s)\,\dot{s}^2}{f_i'(s)} \tag{17}$$

$$\Leftrightarrow \frac{\ddot{q}_i^{\max}}{|f_i'(s)|} - \frac{f_i''(s)\,\dot{s}^2}{f_i'(s)} \geq \ddot{s} \geq \frac{-\ddot{q}_i^{\max}}{|f_i'(s)|} - \frac{f_i''(s)\,\dot{s}^2}{f_i'(s)} \tag{18}$$

If $f_i'(s) = 0$ and $f_i''(s) \neq 0$:

$$(14) \Leftrightarrow \frac{-\ddot{q}_i^{\max}}{|f_i''(s)|} \leq \dot{s}^2 \leq \frac{\ddot{q}_i^{\max}}{|f_i''(s)|} \tag{19}$$

$$\Leftrightarrow \dot{s} \leq \sqrt{\frac{\ddot{q}_i^{\max}}{|f_i''(s)|}} \tag{20}$$

If $f_i'(s) = 0$ and $f_i''(s) = 0$, Eq. 14 is always satisfied.

Eq. 16 and Eq. 18 are equivalent. Thus, the limits on the path acceleration $\ddot{s}$ are

$$\ddot{s}^{\min}(s, \dot{s}) \leq \ddot{s} \leq \ddot{s}^{\max}(s, \dot{s}) \tag{21}$$

with

$$\ddot{s}^{\min}(s, \dot{s}) = \max_{\substack{i\in[1,...,n]\\ f_i'(s)\neq 0}} \left(\frac{-\ddot{q}_i^{\max}}{|f_i'(s)|} - \frac{f_i''(s)\dot{s}^2}{f_i'(s)}\right) \tag{22}$$

$$\ddot{s}^{\max}(s, \dot{s}) = \min_{\substack{i\in[1,...,n]\\ f_i'(s)\neq 0}} \left(\frac{\ddot{q}_i^{\max}}{|f_i'(s)|} - \frac{f_i''(s)\dot{s}^2}{f_i'(s)}\right) \tag{23}$$

The limit on the path acceleration also constrains the path velocity, because in order for a path velocity to be feasible we need $\ddot{s}^{\min}(s, \dot{s}) \leq \ddot{s}^{\max}(s, \dot{s})$. We now derive the path velocity limit $\dot{s}_{\text{acc}}^{\max}(s)$ caused by acceleration constraints. We get rid of the $\min$ and $\max$ functions in Eq. 22 and 23 by requiring the inequality to hold for all possible combinations of arguments to the $\min$ and $\max$ functions.

$$\ddot{s}^{\min}(s,\dot{s}) \leq \ddot{s}^{\max}(s,\dot{s}) \tag{24}$$

$$\Leftrightarrow \left(\frac{\ddot{q}_i^{\max}}{|f_i'(s)|} - \frac{f_i''(s)\dot{s}^2}{f_i'(s)}\right) - \left(\frac{-\ddot{q}_j^{\max}}{|f_j'(s)|} - \frac{f_j''(s)\dot{s}^2}{f_j'(s)}\right) \geq 0$$
$$\forall i,j \in [1,...,n], f_i'(s) \neq 0, f_j'(s) \neq 0 \tag{25}$$

$$\Leftrightarrow -\left(\frac{f_i''(s)}{f_i'(s)} - \frac{f_j''(s)}{f_j'(s)}\right)\dot{s}^2 + \left(\frac{\ddot{q}_i^{\max}}{|f_i'(s)|} + \frac{\ddot{q}_j^{\max}}{|f_j'(s)|}\right) \geq 0$$
$$\forall i,j \in [1,...,n], f_i'(s) \neq 0, f_j'(s) \neq 0 \tag{26}$$

$$\Leftrightarrow -\left|\frac{f_i''(s)}{f_i'(s)} - \frac{f_j''(s)}{f_j'(s)}\right|\dot{s}^2 + \left(\frac{\ddot{q}_i^{\max}}{|f_i'(s)|} + \frac{\ddot{q}_j^{\max}}{|f_j'(s)|}\right) \geq 0$$
$$\forall i \in [1,...,n], j \in [i+1,...,n], f_i'(s) \neq 0, f_j'(s) \neq 0 \tag{27}$$

This gives a set of downward-facing parabolas in $\dot{s}$ horizontally centered around the origin. Each parabola is positive within an interval around 0, which is the interval the feasible velocities may lie in. The positive bound of the interval can be found by setting the parabola equation to zero.

$$-\left|\frac{f_i''(s)}{f_i'(s)} - \frac{f_j''(s)}{f_j'(s)}\right|\dot{s}^2 + \left(\frac{\ddot{q}_i^{\max}}{|f_i'(s)|} + \frac{\ddot{q}_j^{\max}}{|f_j'(s)|}\right) = 0 \tag{28}$$

$$\Leftrightarrow \dot{s} = \sqrt{\frac{\frac{\ddot{q}_i^{\max}}{|f_i'(s)|} + \frac{\ddot{q}_j^{\max}}{|f_j'(s)|}}{\left|\frac{f_i''(s)}{f_i'(s)} - \frac{f_j''(s)}{f_j'(s)}\right|}} \tag{29}$$

The interval of feasible path velocities $\dot{s}$ is defined by the intersection of the feasible intervals of all parabolas. Thus, the upper bound for the path velocity is the minimum of all upper bounds given in Eq. 29. Combining this with the case from Eq. 20, the constraint on the path velocity caused by joint acceleration limits is given by

$$\dot{s} \leq \dot{s}_{\text{acc}}^{\max}(s) \tag{30}$$

with

$$\dot{s}_{\text{acc}}^{\max}(s) = \min \left\{ \begin{array}{l} \min\limits_{\substack{i\in[1,...,n] \\ j\in[i+1,...,n] \\ f_i'(s)\neq 0 \\ f_j'(s)\neq 0 \\ \frac{f_i''(s)}{f_i'(s)} - \frac{f_j''(s)}{f_j'(s)} \neq 0}} \sqrt{\frac{\frac{\ddot{q}_i^{\max}}{|f_i'(s)|} + \frac{\ddot{q}_j^{\max}}{|f_j'(s)|}}{\left|\frac{f_i''(s)}{f_i'(s)} - \frac{f_j''(s)}{f_j'(s)}\right|}}, \min\limits_{\substack{i\in[1,...,n] \\ f_i'(s)=0 \\ f_i''(s)\neq 0}} \sqrt{\frac{\ddot{q}_i^{\max}}{|f_i''(s)|}} \end{array} \right\} \tag{31}$$

B. Joint Velocity Limits

Constraints on the joint velocities are given as

$$-\dot{q}_i^{\max} \leq \dot{q}_i \leq q_i^{\max} \qquad \forall i \in [1,...,n] \tag{32}$$

Plugging Eq. 11 into Eq. 32 yields

$$-\dot{q}_i^{\max} \leq f_i'(s)\dot{s} \leq q_i^{\max} \qquad \forall i \in [1,...,n] \tag{33}$$

If $f_i'(s) = 0$, then Eq. 33 is always satisfied. Otherwise, because $\dot{s} > 0$, Eq. 33 is equivalent to

$$\dot{s} \leq \frac{q_i^{\max}}{|f_i'(s)|} \qquad \forall i \in [1,...,n] \tag{34}$$

Thus, the constraint on the path velocity caused by limits on the joint velocities is given by

$$\dot{s} \leq \dot{s}_{\text{vel}}^{\max}(s) \tag{35}$$

with

$$\dot{s}_{\text{vel}}^{\max}(s) = \min_{\substack{i\in[1,...,n] \\ f_i'(s)\neq 0}} \frac{\dot{q}_i^{\max}}{|f_i'(s)|} \tag{36}$$

We will make use of the slope of this limit curve in the phase plane, which is the derivative by s given by

$$\frac{d}{ds}\dot{s}_{\text{vel}}^{\max}(s) = -\frac{\dot{q}_i^{\max} f_i''(s)}{f_i'(s)\,|f_i'(s)|} \qquad i = \arg\min_{\substack{i\in[1,...,n] \\ f_i'(s)\neq 0}} \frac{\dot{q}_i^{\max}}{|f_i'(s)|} \tag{37}$$

VI. Algorithm

Figure 2 shows the s-$\dot{s}$ phase-plane. The start point is in the bottom-left corner and the end point in the bottom-right corner. The two limit curves limiting the path velocity, which are caused by joint acceleration and joint velocity constraints respectively, are shown. The trajectory must stay below these two limit curves. We want to find a trajectory that maximizes path velocity $\dot{s}$ at every point along the path. While not on the limit curve, the path acceleration must be at one of its limits, i.e. $\ddot{s}_{\min}$ or $\ddot{s}_{\max}$. Thus, at every point on the phase-plane below the limit curve there are two possible directions to move in: one applying minimum acceleration and the other one applying maximum acceleration. The algorithm needs to find the switching points between minimum and maximum acceleration. In Figure 2 switching points are marked by arrows. Switching points from minimum to maximum acceleration must be on the limit curve, because otherwise we could find a faster trajectory above the solution trajectory [5].

The high-level algorithm is described below. It differs slightly from [5]. We integrate backward from the end point as the very last step. This makes the algorithm slightly simpler to implement, because while integrating forward we can never intersect with another trajectory part and while integrating backward we can never reach the limit curve. Section VIII gives some more implementation details. The numbers in Figure 2 correspond to the steps of the algorithm.

1) Start from the start of the path, i. e. $s = 0$ and $\dot{s} = 0$.
2) Integrate forward with maximum acceleration $\ddot{s} = \ddot{s}^{\max}(s,\dot{s})$ until one of the following conditions is met.
   - If $s \geq s_f$, continue from the end of the path, i. e. $s = s_f$ and $\dot{s} = 0$ and go to step 5.
   - If $\dot{s} > \dot{s}_{\text{acc}}^{\max}(s)$, go to step 4.
   - If $\dot{s} > \dot{s}_{\text{vel}}^{\max}(s)$, go to step 3.

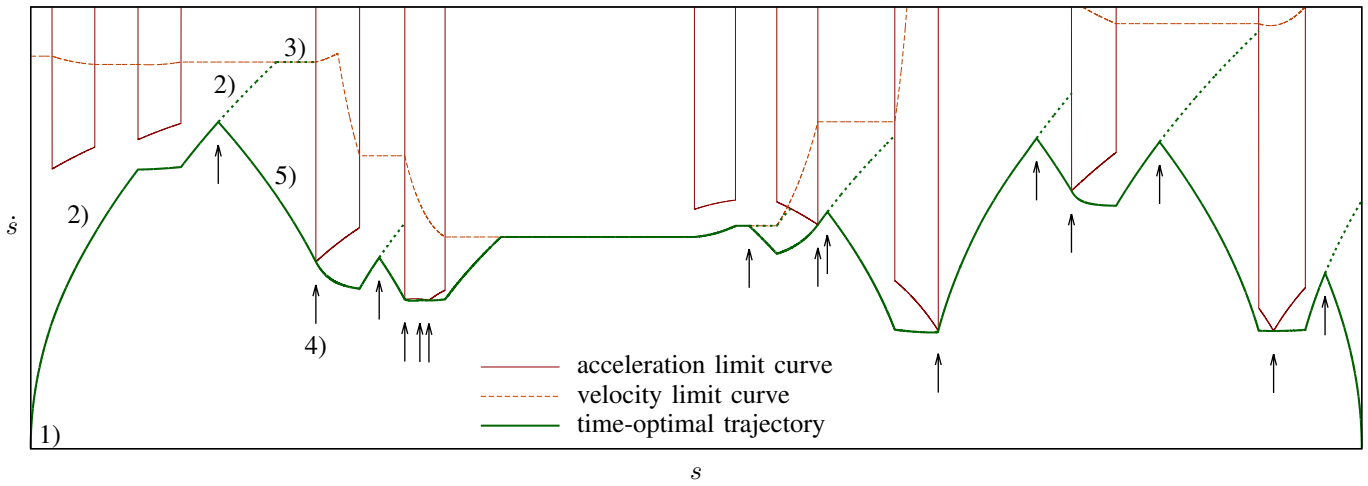


Fig. 2. Phase-plane trajectory

3) Follow the limit curve $\dot{s}^{\max}_{\text{vel}}(s)$ until one of the following conditions is met.
   - If $\frac{\ddot{s}^{\max}(s,\dot{s}^{\max}_{\text{vel}}(s))}{\dot{s}} < \frac{d}{ds}\dot{s}^{\max}_{\text{acc}}(s)$, go back to step 2.
   - If $\frac{\ddot{s}^{\min}(s,\dot{s}^{\max}_{\text{vel}}(s))}{\dot{s}} > \frac{d}{ds}\dot{s}^{\max}_{\text{acc}}(s)$, go to step 4.
4) Search along the combined limit curve for the next switching point. See section Section VII.
   - Continue from the switching point and go to step 5.
5) Integrate backward with minimum acceleration until the start trajectory is hit. (If the limit curve is hit instead, the algorithm failed. We used this condition to detect failures shown in Table I.) The point where the start trajectory is intersected is a switching point. Replace the part of the start trajectory after that switching point with the trajectory just generated.
   - If we transitioned into this step from step 2, halt. The start trajectory reached the end of the path with $s = s_f$ and $\dot{s} = 0$.
   - Otherwise, continue from the end of the start trajectory, which is the switching point found in step 4, and go to step 2.

VII. Switching Points

With the exception of a finite number of points, at every point on the acceleration limit curve there is only a single feasible acceleration. If this acceleration leads into the feasible region, the point on the limit curve is called a trajectory source [6]. If this acceleration leads out of the feasible region, the point is called a trajectory sink. We define trajectory source and sink similarly for the velocity limit curve if all feasible accelerations lead into or out of the feasible region. If the interval of feasible accelerations allows following the velocity limit curve, we call the point singular.

A switching point along a limit curve, is a point where the limit curve switches from being a trajectory sink to being a trajectory source or to being singular. *The* limit curve is the curve given by the minimum of the acceleration limit curve and the velocity limit curve. A switching point of the acceleration or velocity limit curve is a switching point of the limit curve if the limit curve the switching point is on is the lower one. The following two sections deal with finding the switching points on both, the acceleration and velocity limit curves.

A. Caused by Acceleration Constraints

This section describes how to find switching points along the path velocity limit curve $\dot{s}^{\max}_{\text{acc}}(s)$ caused by constraints on the joint accelerations.

We distinguish three cases of these switching points, depending on whether the curve is continuous and/or differentiable at the switching point.

1) Discontinuous: $\dot{s}^{\max}_{\text{acc}}(s)$ is discontinuous if and only if the path curvature $f''(s)$ is discontinuous [7]. For a path generated as described in Section IV the discontinuities are exactly the points where the path switches between a circular segment and straigt line segment. If the path's curvature $f''(s)$ is continuous everywhere, this case of switching points does not happen.

At a discontinuity s there are two path velocity limits $\dot{s}^{\max}_{\text{acc}}(s^-)$ and $\dot{s}^{\max}_{\text{acc}}(s^+)$. The switching point is always at the smaller one of the two. If the discontinuity is a positive step and there is a trajectory sink in the negative direction of the discontinuity, then the discontinuity is a switching point. Equally, if the discontinuity is a negative step and there is a trajectory source in the positive direction of the discontinuity, then the discontinuity is a switching point. Or more formally, a discontinuity of $\dot{s}^{\max}_{\text{acc}}(s)$ is a switching point if and only if

$$\begin{aligned}
&\Big(\dot{s}^{\max}_{\text{acc}}(s^-) < \dot{s}^{\max}_{\text{acc}}(s^+) \\
&\qquad \wedge\ \ddot{s}^{max}(s^-, \dot{s}^{\max}_{\text{acc}}(s^-)) \geq \frac{d}{ds}\dot{s}^{\max}_{\text{acc}}(s^-)\Big) \\
&\qquad \vee \Big(\dot{s}^{\max}_{\text{acc}}(s^-) > \dot{s}^{\max}_{\text{acc}}(s^+) \\
&\qquad \wedge\ \ddot{s}^{max}(s^+, \dot{s}^{\max}_{\text{acc}}(s^+)) \leq \frac{d}{ds}\dot{s}^{\max}_{\text{acc}}(s^+)\Big)
\end{aligned} \tag{38}$$

2) Continuous and Nondifferentiable: The original paper [5] introducing the approach claims that every point along

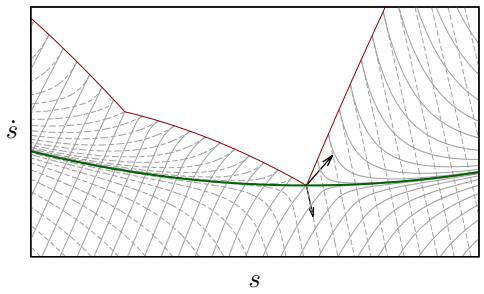


Fig. 3. Minimum and maximum acceleration trajectories near a switching point where the limit curve is nondifferentiable

the limit curve only has a single allowable acceleration with $\ddot{s}^{\min}(s, \dot{s}^{\max}_{\text{acc}}(s)) = \ddot{s}^{\max}(s, \dot{s}^{\max}_{\text{acc}}(s))$. However, this is inaccurate. As [6] notes, there are points along the limit curve where there is an interval of allowable accelerations, i. e. $\ddot{s}^{\min}(s, \dot{s}^{\max}_{\text{acc}}(s)) < \ddot{s}^{\max}(s, \dot{s}^{\max}_{\text{acc}}(s))$. These points are called critical points in [8] and zero-inertia points in [7]. At these points the limit curve is continuous but nondifferentiable [8]. As noted in [6], a neccessary condition for such switching points is

$$\exists i : f_i'(s) = 0 \tag{39}$$

For a path generated from waypoints as described in Section IV, points satisfying Eq. 39 can be easily calculated. There are at most n per circular segment.

[6] does not note that choosing the correct acceleration at such a switching point might be an issue. [6] proposes to integrate backward from such a switching point with minimum acceleration and integrate forward with maximum acceleration as usual. [8] notices that the acceleration at such a switching point needs special consideration. [8] notes that the maximum or minimum acceleration cannot always be followed, because they would lead into the infeasible region of the phase plane. [8] calls such a point a singular point and proposes to follow the limit curve tangentially. However, both [6] and [8] are in error.

Figure 3 shows a switching point at a point where the limit curve is nondifferentiable. At the switching point two arrows indicate the direction of motion in the phase plane when applying minimum or maximum acceleration, respectively. In the feasible region of the phase plane gray curves visualize two vector fields. They show minimum- and maximum-acceleration trajectories. Minimum-acceleration trajectories are dashed, maximum-acceleration trajectories are solid. In the case shown here, if integrating backwards with minimum acceleration, the infeasible region would be entered. Thus, according to [8] we would have to follow the red limit curve tangentially backwards from the switching point. For integrating forward [6] and [8] propose to follow the upward arrow. However, both of these motions are not possible, as they would not move along the vector field sourrounding the switching point. Zero is the only acceleration at the switching point that conforms with the vector field around it, resulting in a trajecory moving horizontally in the phase plane.

We claim that the optimal acceleration is zero at every such switching point. We leave the proof for future work, but all of our experimental data supports this claim.

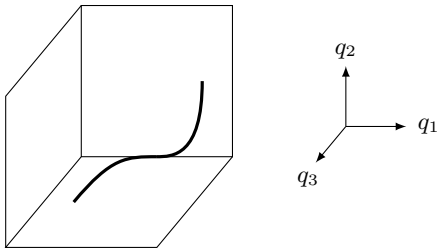


Fig. 4. Joint acceleration space for n=3 with box constraints and a trajectory for a switching point that meets the limit curve tangentially.

In order for the zero acceleration to be feasible, the limit curve must switch from a negative to a positive slope at the switching point. We claim that the limit curve switching from a negative to a positve slope is a necessary and sufficient condition for a switching point at a point of nondifferentiability of the limit curve. Together with the sufficient condition stated in Eq. 39, this allows for an explicit enumeration of all such switching points.

3) Continuous and Differentiable: According to the following lemma this case does not exist.

Lemma 1: If only joint accelerations are constrained, the path is piece-wise coplanar, and the curvature $f''(s)$ is discontinuous at the points where the pieces are stitched together, then there are no switching points at which the limit curve is continuous and differentiable.

Proof: Before the switching point the path acceleration is at the lower limit. This implies that at least one joint is at its acceleration limit. After the switching point the path acceleration is at its upper limit. This implies that at least one joint distinct from the previous one is at its acceleration limit. At the switching point both joints are at their acceleration limit. The limit curve, the phase-plane trajectory and the joint-space trajectory are continuous and differentiable near the switching point. One joint approaches its acceleration limit at the switching point and then stays at the limit. Another distinct joint is at its acceleration limit and leaves the limit at the switching point. Figure 4 shows the joint acceleration space for a 3-joint system with the joint acceleration limits shown as a box. At the switching point two joint are at their acceleration limit. Thus, the trajectory is on an edge of the box at the switching point. Before the switching point the trajectory stays on one surface of the box, approaches the edge tangentially. After the switching point the trajectory leaves the edge tangentially while staying on another surface of the constraint box. This trajectory lives in at least a three-dimensional subspace of the joint acceleration space. There is no two-dimensional subspace that includes the trajectory in the vicinity of the switching point. Because the path is piecewise planar, the accelerations in joint space are also piecewise planar. Because there is no trajectory in joint acceleration space that cannot be included in a two-dimensional subspace, a switching point where the limit curve is differentiable does not exist. The only case where such a switching point can exist

is where two planar parts of the path are stitched together. However, these points are those where the curvature $f''(s)$ is discontinuous and thus the limit curve is discontinuous. ■

B. Caused by Velocity Constraints

This section describes how to find switching points along the path velocity limit curve $\dot{s}^{\max}_{\text{vel}}(s)$ caused by constraints on the joint velocities. We distinguish two cases of these switching points, depending on whether $\ddot{s}^{\min}(s, \dot{s}^{\max}_{\text{vel}}(s))$ is continuous.

1) Continuous: s is a possible switching point if, only if

$$\ddot{s}^{min}(s, s^{\max}_{\text{vel}}(s)) = \frac{d}{ds}\dot{s}^{\max}_{\text{acc}}(s) \tag{40}$$

We search for these switching points numerically by stepping along the limit curve until we detect a sign change. Then we use bisection to more accurately determine the switching point. See also Section VIII-B.

2) Discontinuous: $f_i''(s)$ being discontinuos is a necessary condition for $\ddot{s}^{\min}(s, \dot{s}^{\max}_{\text{vel}}(s))$ being discontinuos. s is a possible switching point if and only if

$$(\ddot{s}^{min}(s^-, s^{\max}_{\text{vel}}(s^-)) \geq \frac{d}{ds}\dot{s}^{\max}_{\text{acc}}(s^-)) \tag{41}$$

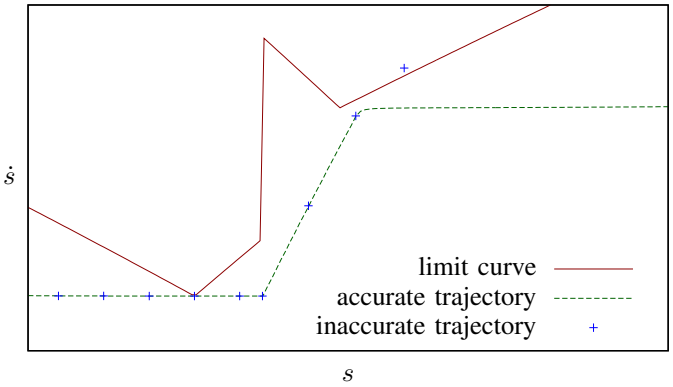
$$\wedge\, (\ddot{s}^{min}(s^+, s^{\max}_{\text{vel}}(s^+)) \leq \frac{d}{ds}\dot{s}^{\max}_{\text{acc}}(s^+)) \tag{42}$$

VIII. Numerical Considerations

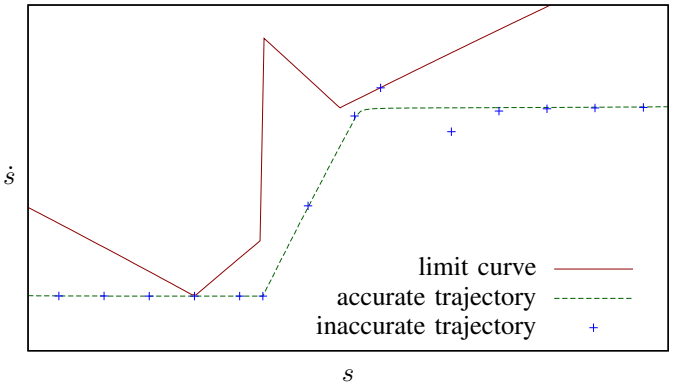
When doing floating-point calculations we must accept approximate results. However, we cannot accept the algorithm failing completely due to numerical inaccuracies. We now describe how numerical inaccuracies can make the algorithm described in section Section VI fail and the measures we have taken to avoid that. None of the previous papers on this approach [5–9] have dealt with numerical issues.

A. Integration

The algorithm described in Section VI assumes that we can exactly integrate the trajectory. Under this assumption it is impossible to hit the limit curve at a trajectory source when integrating forward. However, as we have to do the integration numerically with some non-zero step size, it is not exact. This can lead to the limit curve being hit at a trajectory source. Figure 5 shows an example of the trajectory hitting the limit curve at a trajectory source. The figure shows the limit curve together with an accurate trajectory generated using a small step size and a not-so-accurate trajectory generated with a 10 times larger step size (1 ms). The trajectories enter the figure on the left at minimum acceleration until they touch the limit curve. Then they switch to maximum acceleration. In the following the accurate trajectory gets close to the limit curve but does not hit it. The inaccurate trajectory, however, because of the larger step size, misses the bend shortly before the limit curve and hits the limit curve. According to the algorithm described in Section VI and the algorithms described in previous work [5, 6, 8, 9] we would have to stop the forward integration, search for the next switching point along the limit curve and integrate backward from there until we hit the forward trajectory. However, when integrating backward



(a) failure without source/sink check



(b) success with source/sink check

Fig. 5. Dealing with integration inaccuracies

from the next switching point we might hit the limit curve instead of the forward trajectory. This is because the algorithm relies on the fact that there are no trajectory sources between where the forward trajectory hits the limit curve and the next switching point, which is not the case if the forward trajectory hits the limit curve at a trajectory source. To deal with this issue, we determine the intersection point with the limit curve by bisection and then check whether this point is a trajectory source by comparing the slope of the limit curve with the slope of the maximum-acceleration trajectory. Figure 5 shows the result of this measure. The inaccurate trajectory does not follow the accurate one exactly, but it does not stop when the limit curve is hit. Thus the algorithm succeeds.

Note that although the smaller step size solved the problem in the example shown in Figure 5, it does not do so in general. In general, a smaller step size or a better integration method only make it less likely that the limit curve is hit at a trajectory source. Thus, the method described above is necessary to ensure completeness of the algorithm.

B. Switching Point Search

All previous work at least partially relies on numerical search to find switching points, but none notes the potential issues caused by using it. We also use numerical search, but only along the velocity limit curve. If we only had acceleration constraints, numerical search would not be necessary. We try to avoid numerical search as much as possible, because it

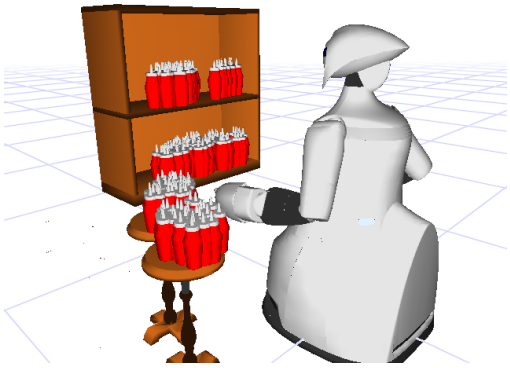


Fig. 6. 200 start and goal locations

Time Step	Computation Time	Execution Time	Failure rate without source/sink check	Failure rate with check
10 ms	0.2 s	4.74 s	13 %	0 %
1 ms	0.9 s	4.71 s	3 %	0 %
0.1 ms	31.8 s	4.70 s	0 %	0 %

TABLE I
RESULTS FOR VARYING TIME STEP

is slower, less accurate and might make the algorithm fail. Our algorithm relies on the fact that we can find the next switching point along the limit curve. However, by searching numerically and stepping along the limit curve, we could theoretically miss a switching point and find one that is not the next one. The algorithm could potentially be adapted to work with any switching point without requiring it being the next one. However, during our experiments we did not encounter a failure caused by missing a switching point.

IX. Experimental Results

We evaluated our approach by generating trajectories for a 7-DOF robot arm that is given the task of picking and placing a bottle from or onto a shelf or table. The results are averages over 100 pick-and-place operations. For each pick-and-place operation we randomly selected a pick and a place location. Figure 6 shows the 200 pick and place locations. We used a bidirectional RRT to plan a configuration space path for the arm. Each pick-and-place operation constists of 3 path segments: from the initial arm configuration to the pick configuration, to the place configuration and back to the intial configuration. In our evaluation we ignore the finger motion necessary to grasp the object. The robot shown in Figure 6 uses only its left arm. The arm configuration shown is the inital arm configuration, which is reached at the beginning and end of every pick-and-place operation. The paths generated by the RRT planner are shortened using random short cutting. The result are paths given by waypoints at most 0.1 apart. Here we evaluate the trajectory generation from these waypoints. This method of generating example paths is very real-world oriented and does not allow us to handcraft waypoints until the trajectory generation accidentally succeeds.

Table I shows computation and execution times for different time steps used for the integration of the trajectory. The computation time was achieved on a single core of an Intel Core i5 at 2.67 Ghz. Smaller time steps lead to a longer computation time and a slightly closer to optimal path. What is not shown here is that a smaller step size also leads to the constraints being satisfied more accurately. Related to that is the fact that a larger time step makes the algorithm more likely to fail if the counter measures for numerical inaccuracies described in Section VIII-A are omitted. Our algorithm is robust enough to successfully generate trajectories for 100 random pick-and-place operations using different time steps.

We also ran our algorithm on a real-robot arm following a path defined by hand-crafted waypoints. A video of this and a few example pick-and-place operations is available at http://www.golems.org/node/1570.

X. Conclusion

We presented an algorithm to follow a path in time-optimal manner while satisfying joint acceleration and velocity constraints. We added blends to a list of waypoint in order to be able to use the output of standart path planners. Our improvements to existing work make the algorithm more robust and less likely to fail. We showed the robustness of our implementation by following paths planned by an RRT planner for 100 random pick-and-place operations.

Acknowledgements

This work is supported by Toyota Motor Engineering & Manufacturing North America (TEMA). We appreciate the great contribution to our robotics research and education.

References

[1] J.J. Craig. *Introduction to Robotics: Mechanics and Control (3rd Edition)*. Prentice Hall, 2004.

[2] B. Siciliano, L. Sciavicco, and L. Villani. *Robotics: modelling, planning and control*. Springer, 2009.

[3] T. Kunz and M. Stilman. Turning Paths Into Trajectories Using Parabolic Blends. Technical Report GT-GOLEM-2011-006, Georgia Institute of Technology, 2011.

[4] J.E. Bobrow, S. Dubowsky, and J.S. Gibson. Time-optimal control of robotic manipulators along specified paths. *The Int. Journal of Robotics Research*, 4(3):3–17, 1985.

[5] 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.

[6] F. Pfeiffer and R. Johanni. A concept for manipulator trajectory planning. *IEEE Journal of Robotics and Automation*, 3(2):115–123, 1987.

[7] J.-J.E. Slotine and H.S. Yang. Improving the efficiency of time-optimal path-following algorithms. *IEEE Transactions on Robotics and Automation*, 5(1):118–124, 1989.

[8] Z. Shiller and H.H. Lu. Computation of path constrained time optimal motions with dynamic singularities. *Journal of dynamic systems, measurement, and control*, 114:34, 1992.

[9] L. Zlajpah. On time optimal path control of manipulators with bounded joint velocities and torques. In *Proc. of IEEE Int. Conf. on Robotics and Automation*, 1996.