2023 IEEE International Conference on Robotics and Automation (ICRA 2023)
May 29 - June 2, 2023. London, UK

Boundary Conditions in Geodesic Motion Planning for Manipulators

Mario Laux[1] and Andreas Zell[1]

***Abstract*— In dynamic environments, robotic manipulators and especially cobots must be able to react to changing circumstances while in motion. This substantiates the need for quick trajectory planning algorithms that are able to cope with arbitrary velocity and acceleration boundary conditions. Apart from dynamic re-planning, being able to seamlessly join trajectories together opens the door for divide-and-conquer-type algorithms to focus on the individual parts of a motion separately.**

While geodesic motion planning has proven that it can produce very smooth and efficient actuator movement, the problem of incorporating non-zero boundary conditions has not been addressed yet. We show how a set of generalized coordinates can be used to transition between boundary conditions and free movement in an optimal way while still retaining the known advantages of geodesic planners. We also outline, how our approach can be combined with the family of time-scaling algorithms for further improvement of the generated trajectories.

I. Introduction

Joint trajectories of robotic manipulators such as the ones shown in Fig. 1 can be subject to many different kinds of constraints. While the initial configuration is given by the manipulator's current state, the target configuration is typically determined by the end-effector's desired position and orientation.

We identify two conceptually different types of constraints: Physical limitations inherent to the design of a manipulator, such as maximum joint accelerations or torques, are unconditional requirements. In most cases, a dynamic model is available, making the inclusion of such conditions in path planning algorithms relatively easy. But there are also much less quantifiable, softer requirements like for example the intuitive predictability of the robot's motion patterns.

Controlling the end-effector's path typically involves choosing waypoints in task space. Once those have been translated back into joint space by means of some inverse kinematics algorithm, the final joint trajectory can be obtained by interpolation. However, the number of waypoints is the crucial parameter for such approaches: Picking many waypoints makes the manipulator's motion very predictable, but leaves little room for further optimization in joint space. This then often leads to rough joint trajectories and faster wear of the manipulator. On the contrary, using fewer waypoints allows for smoother joint trajectories but comes at the price of giving up some control over the motion in task space.

[1]*Chair of Cognitive Systems*, Department of Computer Science, University of Tübingen, Tübingen, Germany. mario.laux@uni-tuebingen.de

Fig. 1. Examples for robotic manipulators. left: Franka Emika Panda Cobot, middle: Kinova JACO^2 Cobot, right: KUKA KR 6 R900 sixx (KUKA AG)

For the lack of other canonical choices, waypoints are often picked to lie on the line segment connecting the initial and the final position of the end-effector. As far as predictability is concerned, this approach might already be unnecessarily restrictive: When reaching for an object, humans themselves don't usually move their hand along an exactly straight path, but find a compromise between ergonomic feasibility and execution time instead.

An overarching, recurring problem in motion planning tasks for robotic manipulators is a trade-off between control and optimization potential in joint space and in task space. While linear interpolation in joint space obviously minimizes the overall joint movement, it most often leads to unintuitive end-effector behavior. This is especially problematic in the case of cobots that are supposed to be able to safely interact with humans.

Geodesic path planning circumvents the need to make uninformed choices such as the position of waypoints prior to the actual optimization. To this aim, it is based on a general notion of cost, which is best explained by an example. Consider the paths shown in Fig. 2: When viewed on a curved surface, they look as "straight" as the surface allows. In fact, they are exactly the paths that a car confined to the surface would follow when the steering wheel was never turned. Such curves are called *geodesics*.

The curvature equips the surface with a position-dependent cost that makes some areas more desirable to travel through than others. It becomes apparent that geodesics tend to prefer regions with lower travel cost, but they by no means always follow the direction of lowest cost. This is a first indicator that following the curvature instead may lead to globally better results.

The path planning problem for manipulators can be treated in a very similar way: First, one has to construct a cost-

979-8-3503-2365-8/23/$31.00 ©2023 IEEE

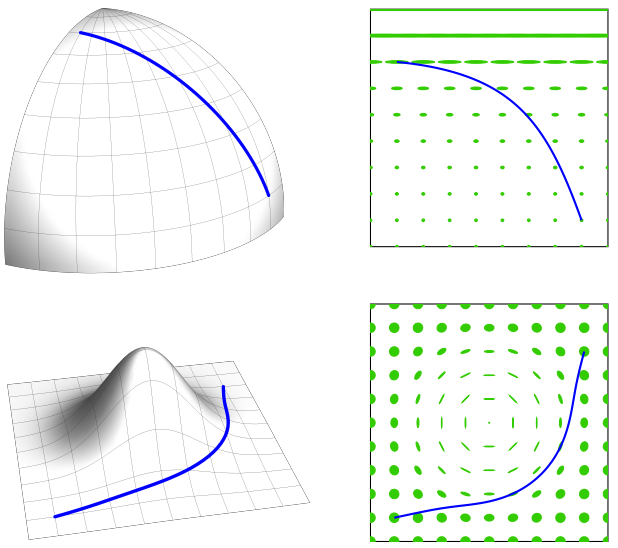

Fig. 2. Examples of geodesics. Cost landscapes (right) can be expressed as curvature (left) and vice versa. The locations of the ellipses in the cost landscape plots correspond to the intersections of the mesh lines in the surface plots. For any given point, the ellipses indicate how far one can travel at a cost of one unit. For the case of the sphere, there is a singularity at the north pole where travel in longitude is free. This is an artifact of the choice of coordinates, not a special property of the north pole.

landscape on the manipulator's joint space that expresses optimization goals such as end-effector or joint movement. Then, when given two points in configuration space, one can find the shortest possible connection between them by following the curvature that is induced by the cost-landscape. In the course of this process, the whole geodesic is determined at once without the need to fix any waypoints in advance.

However, it is already obvious from the two examples in Fig. 2 that the direction in which the starting point is left or in which the destination point is reached, are byproducts of the optimization process and cannot be chosen freely. This fundamental property of geodesics stands in the way of combining the advantages of geodesic motion planning with the ability to respect velocity or acceleration boundary conditions.

II. Related Work

Boundary conditions other than initial and final value are rarely taken into account when discussing motion planning for robot manipulators [1]. The vast majority of planners is based on some kind of interpolation between waypoints [2]–[8], where the role of the exact algorithm usually is to ensure smoothness.

Depending on the number of waypoints, there is more or less room to exploit excess degrees of freedom of higher-degree polynomials, e.g. to ensure that certain boundary conditions are met. While it is certainly possible to construct interpolations in joint space that are compatible with given boundary conditions, these methods lack to ability to optimize amongst all the possible choices.

One of the main reasons that boundary condition agnostic algorithms still find many applications is the fact that every motion can be time-scaled in such a way that it starts from a rest and ends in a rest. In between, time-scaling can be used to accomplish goals secondary to reaching a desired target configuration like e.g. optimal execution time [9], even in the presence of external forces and friction [10] or in the presence of uncertainties [11].

As opposed to a strict hierarchy of a primary and possible secondary tasks, there is only one common notion of cost in the language of geodesic motion planning. While, for example, the decoupling of task space from certain portions of joint space in a robot's null space can be exploited explicitly [12]–[14], geodesic motion planners implicitly capture this potential in form of a smaller travel cost.

As a general rule of thumb, geodesic planning methods deal well with scenarios where there are continually many options to choose from [15]–[19]. While the tendency of geodesics to find overall beneficial compromises between all the optimization goals is usually desirable, it complicates the incorporation of exact boundary conditions. To the best of our knowledge, boundary conditions have never been considered in the context of geodesic path planning before.

III. Preliminaries

A. Conventions and Notation

For manipulators with F degrees of freedom, we will use the symbol $\mathbf{u}(x) \in \mathbb{R}^F$ for its joint trajectories as a function of the motion progress $x \in [0, 1]$. Since the motion progress is dimensionless, $\mathbf{u}, \mathbf{u}'$ and $\mathbf{u}''$ all have *the same* physical dimension. In this context, the symbol $\tilde{\mathbf{u}} := \mathbf{u} \circ s$ shall refer to a time-scaled version of $\mathbf{u}$, where the function s maps an actual point in time to the corresponding motion progress. As time-scaling not only scales velocities but can also influence the *direction* of the acceleration, we cannot omit it from our analysis. Note that with this setup, s' has the physical dimension of an inverse time. In order to conveniently fix a time scale, we introduce the constant τ representing one unit of time, e.g. $\tau = 1\,\mathrm{s}$.

We will use the symbol $\mathbf{y}$ for coordinates in general and the symbol G for a general metric. Such coordinates will depend on the motion progress. The metrics, in turn, will depend on the coordinates. Where general coordinates relate to actual joint trajectories, we may emphasize this using the notation $\mathbf{u}(\mathbf{y})$.

B. The Geodesic Equations

Formally speaking, a metric $G(\mathbf{y})$ associates a travel direction $\mathbf{y}'$ with a local cost of

$$\|\mathbf{y}'\|_G := \sqrt{\langle \mathbf{y}', G(\mathbf{y})\, \mathbf{y}' \rangle}$$

such that the overall cost along a curve can then be obtained by integration. Note that $\|\cdot\|_{\mathrm{id}}$ coincides with the usual Euclidean metric.

Fig. 2 shows some examples for the fact that minimizers $\mathbf{y} = (y_1, ..., y_N) : [0, 1] \to \mathbb{R}^N$ of the overall cost

are geodesics. The visually more accessible characterization would be that a geodesic strictly follows the curvature induced by the metric without taking any detours. The well-known *geodesic equations* express this fact as constraints on the acceleration along the path:

$$y_k'' + \sum_{i=1}^{N}\sum_{j=1}^{N} \Gamma_{ij}^k \, y_i' \, y_j' = 0 \quad \text{for all } k = 1, ..., N \tag{1}$$

This is a system of N coupled second-order ordinary differential equations for the components of $\mathbf{y}$. The so-called Christoffel symbols Γ_{ij}^k depend on $\mathbf{y}$ and can be obtained directly from the metric:

$$\Gamma_{ij}^k = \frac{1}{2}\sum_{m=1}^{N} (G^{-1})_{mk} \left(\frac{\partial G_{im}}{\partial y_j} + \frac{\partial G_{jm}}{\partial y_i} - \frac{\partial G_{ij}}{\partial y_m} \right) \tag{2}$$

Note that it is in fact the coordinate-dependence of the metric that induces curvature. For any constant metric the geodesic equations (1) reduce to the equation of straight lines in "flat" space, $\mathbf{y}'' = 0$. Some more intuition is provided in [15] and [16]. For an in-depth introduction to differential geometry and a derivation of the geodesic equations, we refer the reader to [20] and [21].

In order to develop a feeling for how the geodesic equations might look like, we shortly want to consider the unit sphere as an example. Using the usual coordinates $\mathbf{y} = (\varphi, \theta)$ with φ being the longitude and θ being the latitude, the metric takes the form

$$G(\varphi, \theta) = \begin{bmatrix} \cos^2(\theta) & 0 \\ 0 & 1 \end{bmatrix} \tag{3}$$

and gives rise to the following geodesic equations:

$$\begin{aligned} \theta'' + \tfrac{1}{2}\sin(2\theta)\,\varphi'^{\,2} &= 0 \\ \varphi'' - 2\tan(\theta)\,\theta'\,\varphi' &= 0 \end{aligned} \tag{4}$$

At the north pole for $\theta = 90°$ the metric G becomes singular, which translates to a singularity in (4). In order to obtain the geodesic shown on the sphere in Fig. 2, we had to solve these equations subject to the boundary conditions $\varphi(0) = 10°$, $\theta(0) = 70°$ and $\varphi(1) = 80°$, $\theta(1) = 10°$. Some hints regarding the numerical solution of the geodesic equations can be found in Section VI.

C. Constructing Metrics for Robot Arms

Recall that a metric associates coordinate velocities with a cost. If we want to make it costly for the end-effector to move (as a metric on joint space $\mathbf{u}$), we first have to relate end-effector movement to the coordinate velocity $\mathbf{u}'$. For any manipulator, we can write $\mathbf{v} = J_\mathbf{v} \cdot \mathbf{u}'$, where $\mathbf{v}$ denotes the unscaled Cartesian velocity of the end-effector and $J_\mathbf{v}$ is the corresponding block of the manipulator's Jacobian. Thus, we have

$$\|\mathbf{v}\|_2^2 = \langle \mathbf{u}', J_\mathbf{v}^T J_\mathbf{v}\, \mathbf{u}' \rangle \tag{5}$$

and the metric

$$G_{\text{mov}} := J_\mathbf{v}^T \, J_\mathbf{v} \tag{6}$$

will suppress end-effector movement. The same line of reasoning holds for the angular velocity and the corresponding part $J_{\boldsymbol{\omega}}$ of the Jacobian, such that

$$G_{\text{rot}} := J_{\boldsymbol{\omega}}^T \, J_{\boldsymbol{\omega}} \tag{7}$$

quantifies orientation change. Finally, the metric $\mathbb{1}$ can measure joint movement itself. Once all the desired types of cost have been expressed as a metric, one can simply form a weighted sum arriving at an intuitive overall cost. We use the joint space metric

$$G = \mathbb{1} + 200\, G_{\text{mov}} + 15\, G_{\text{rot}} \tag{8}$$

for all experiments throughout this paper, as it has also been used in [15].

IV. Boundary Conditions

A. Problem Overview

We are interested in a joint trajectory $\tilde{\mathbf{u}} : [0, T] \to \mathbb{R}^F$ that satisfies given boundary conditions of the following form:

$$\begin{aligned} \tilde{\mathbf{u}}(0) &= \mathbf{s}_0 & \tilde{\mathbf{u}}(T) &= \mathbf{s}_1 \\ \tilde{\mathbf{u}}'(0) &= \mathbf{v}_0 & \tilde{\mathbf{u}}'(T) &= \mathbf{v}_1 \\ \tilde{\mathbf{u}}''(0) &= \mathbf{a}_0 & \tilde{\mathbf{u}}''(T) &= \mathbf{a}_1 \end{aligned} \tag{9}$$

While continuity or even smoothness of higher derivatives might be desirable, it is not strictly necessary. On the one hand, manipulators typically allow for very high values of the jerks $\tilde{\mathbf{u}}'''$. On the other hand, the adjustment of higher derivatives *at a single point* can usually be accomplished by small local adjustments without changing the global shape of the trajectory significantly.

As we have discussed in Section I, the problem of finding a geodesic connecting two points automatically determines the initial and final directions of travel. For a manipulator that is already in motion or that must arrive at its destination with a certain Cartesian velocity, geodesic trajectories in their usual form will not work.

A naive workaround might be to simply start in a non-optimal direction compliant with the boundary conditions and then to slowly return to the geodesic trajectory. However, once the robot leaves the geodesic, returning to it is most certainly sub-optimal. Moreover, the question of how to transition back to the original geodesic would be an optimization problem by itself.

Another idea would be to try to change the metric in such a way that it would penalize motion that is not compliant with the desired boundary conditions. But since geodesics do not necessarily follow the direction of lowest travel cost but rather find compromises, such an approach cannot guarantee that the boundary conditions are met exactly.

We will tackle the problem in three steps. First, in subsection IV-B, we will explore the idea that in order to incorporate restrictions, one actually has to *add* new degrees of freedom that handle these restrictions. This will result in a new, more powerful set of coordinates. In IV-C we briefly discuss how to transfer existing metrics like the ones described in III-C into the new coordinate system. Lastly in

IV-D, we will achieve our goal of obtaining an optimized trajectory *from a geodesic* in the new coordinates.

B. Boundary Condition Coordinates

As a system of second-order in F independent variables, the geodesic equations (1) for a trajectory $\mathbf{u} : [0,1] \to \mathbb{R}^F$ allow for only $2F$ boundary conditions. Put differently, the $6F$ boundary conditions (9) constitute a severe over-determination of the problem. This observation points us in the right direction: we need to introduce new degrees of freedom.

Our proposed solution is rooted in the idea to superimpose three different functions. One function $\mathbf{A}$ that behaves as intended in the beginning of the motion, another function $\mathbf{B}$ that has the desired behavior at the end and a "free" function $\mathbf{f}$ that governs the unconstrained part of the motion. Although $3F$ independent variables would perfectly match our $6F$ boundary conditions, superimposing three completely independent functions just to obtain a single one seems to introduce a wasteful amount of redundancy. So instead of allowing for $\mathbf{A}$ and $\mathbf{B}$ to be completely free, we choose suitable functions according to (9)

$$\mathbf{A}(p) := \mathbf{s}_0 + p\,\tau\,\mathbf{v}_0 + \tfrac{1}{2}p^2\,\tau^2\,\mathbf{a}_0 \tag{10}$$

$$\mathbf{B}(q) := \mathbf{s}_1 - (1-q)\,\tau\,\mathbf{v}_1 + \tfrac{1}{2}(1-q)^2\,\tau^2\,\mathbf{a}_1 \tag{11}$$

and only consider their scalar arguments p and q as independent variables. The scaling by the constant τ is required as we are still dealing with the unscaled coordinates $\mathbf{u}$. Then our idea becomes

$$\mathbf{u} = (1-\beta(c))\,\mathbf{A}(p) + \beta'(c)\,\mathbf{f} + \beta(c)\,\mathbf{B}(q)\,, \tag{12}$$

where the final new scalar variable c governs the smooth transition from $\mathbf{A}$ to $\mathbf{f}$ and finally to $\mathbf{B}$ by means of a smooth conditional function [22] like *smootherstep*, which smoothly steps from 0 to 1:

$$\beta(c) := \begin{cases} 0 & \text{if } c \le 0 \\ 6c^5 - 15c^4 + 10c^3 & \text{if } 0 < c < 1 \\ 1 & \text{if } 1 \le c. \end{cases} \tag{13}$$

At $c = 0$, only $\mathbf{A}$ contributes and at $c = 1$ the trajectory is solely determined by $\mathbf{B}$. For $0 < c < 1$, we have $\beta'(c) > 0$ such that the free function $\mathbf{f}$ can shape the outcome and also compensate for the arbitrary choices for $\mathbf{A}$, $\mathbf{B}$ and β if needed. Overall, we have introduced the new set of coordinates

$$\mathbf{y} := (\mathbf{f}, c, p, q) \in \mathbb{R}^{F+3}\,. \tag{14}$$

Note that the $F+3$ components of $\mathbf{y}$ play the same role as φ and θ did in the example from section III-B: The solution to the optimization problem will be a trajectory through the space of all possible coordinate values and thus we will end up with a solution $\mathbf{y} : [0,1] \to \mathbb{R}^{F+3}$, which can then be translated back into the joint trajectory via (12). We will see, that this ansatz is just good enough to enforce all the desired boundary conditions on $\mathbf{u}$.

We want to emphasize that the technique presented here applies more broadly to the concept of geodesic optimization: In order to incorporate a hard constraint, one can try to find a set of coordinates that enforce the desired constraint no matter how they are chosen and then compute the geodesic in those coordinates.

C. Boundary Condition Metrics

Before we can start with the computation of geodesics in our new coordinates, we need to think about which metric we want to use. Assuming we already have a metric G in the original coordinates $\mathbf{u}$, we can simply translate $\mathbf{y}$ back into $\mathbf{u}(\mathbf{y})$ and use G. A short computation shows that for

$$G^* := \tfrac{\partial \mathbf{u}}{\partial \mathbf{y}}^T G\, \tfrac{\partial \mathbf{u}}{\partial \mathbf{y}} \tag{15}$$

we indeed get $\|\mathbf{y}'\|_{G^*} = \|\mathbf{u}'\|_G$ as desired. In addition to the cost derived from $\mathbf{u}$, we also need to add some cost to movement within the new coordinates as to ensure that no unnecessary changes in $\mathbf{f}$, c, p or q occur. We find that the very simple boundary condition metric defined by

$$G_{\text{BC}} := \mathbb{1} + G^* \tag{16}$$

works well in practice. The higher the weighting factor of G^* is chosen, the faster the transitions out of and in to the boundary conditions will occur.

D. Boundary Condition Geodesics

Computing geodesics in our new coordinates $\mathbf{y}$ effectively hands over control about the transition from and to the boundary conditions to the optimization algorithm – as it should be. We obtain our final solution in two steps: First, we compute the geodesic $\mathbf{y} : [0,1] \to \mathbb{R}^{F+3}$ subject to the following conditions at $x = 0$ and $x = 1$:

$$\begin{aligned} \mathbf{f}(0) &= \mathbf{0} & \mathbf{f}(1) &= \mathbf{0} \\ c(0) &= 0 & c(1) &= 1 \\ p(0) &= 0 & q(1) &= 1 \\ p'(0) &= 1 & q'(1) &= 1 \end{aligned} \tag{17}$$

Note that this is a slight variation on the usual setup as we fix p and p' in the beginning and q and q' at the end. Once this geodesic is known, we may use any time-scaling function $s : [0,T] \to [0,1]$ with the properties

$$\begin{aligned} s(0) &= 0 & s(T) &= 1 \\ s'(0) &= 1/\tau & s'(T) &= 1/\tau \\ s''(0) &= -p''(0)/\tau^2 & s''(T) &= -q''(1)/\tau^2 \end{aligned} \tag{18}$$

to obtain our final trajectory

$$\tilde{\mathbf{u}} := \mathbf{u}(\mathbf{y} \circ s)\,. \tag{19}$$

A straightforward calculation shows, that $\tilde{\mathbf{u}}$ satisfies all the boundary conditions from (9). The fact that a specific time-scaling is able to realize boundary conditions is due to the construction of $\mathbf{u}(\mathbf{y})$ in (12). In the general case, this would not be possible.

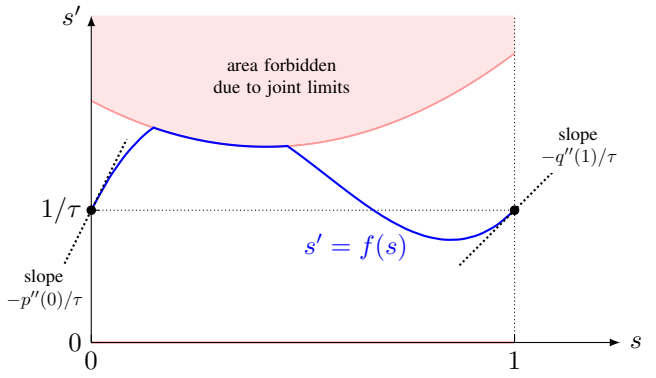

Fig. 3. Conceptual illustration of the geodesic time-scaling problem subject to (18) in phase space (s, s'). Except for the slopes at the beginning and the end, the curve f can be chosen relatively freely. Restrictions on f and f' are usually derived from a manipulator's joint limits.

E. Additional Time-Scaling of Geodesics

Time-scaling is a post-processing optimization technique. It takes advantage of the fact that a re-parameterization of a trajectory does not affect the Cartesian path traced out by any fixed point on the robot. A time-scaling function $s : [0, T] \to [0, 1]$ uniquely maps a point in time to the associated progress along the path. Since one point in time belongs to exactly one progress value, it is possible to express s' as a function of s instead of time, i.e. $s' = f(s)$ for some function $f > 0$. The larger f, the faster the overall motion will be. Fig. 3 outlines this process. For details on time-optimal time-scaling, we refer the reader to [9].

The solution to the equation $s' = f(s)$ for a given positive function f can be expressed as $s = F^{-1}$, where F is uniquely determined by the properties $F(0) = 0$ and $F' = 1/f$. The total duration of the motion is then $T = F(1)$.

F. Illustrative Example

In order to illustrate the role of time-scaling in our algorithm, we want to work through an illustrative example. The task is to move an object along a single dimension from one point to another assuming the boundary conditions

$$\begin{aligned} \tilde{u}(0) &= 0 & \tilde{u}(T) &= 1\,\mathrm{m} \\ \tilde{u}'(0) &= \tfrac{1}{3}\,\mathrm{m/s} & \tilde{u}'(T) &= \tfrac{1}{3}\,\mathrm{m/s} \\ \tilde{u}''(0) &= 0 & \tilde{u}''(T) &= 1\,\mathrm{m/s^2} \end{aligned} \tag{20}$$

for the object's coordinate function $\tilde{u} : [0, T] \to \mathbb{R}$. We choose the time-scaling in such a way that it minimizes the duration T while the limits

$$\begin{aligned} |\tilde{u}'(t)| &\leq 2\,\mathrm{m/s} \\ |\tilde{u}''(t)| &\leq 6\,\mathrm{m/s^2} \end{aligned} \tag{21}$$

have to be respected for all $t \in [0, T]$. This is sometimes referred to as "bang-bang-control" [9] as it requires high acceleration in the beginning and high deceleration towards the end. The results are shown in Fig. 4.

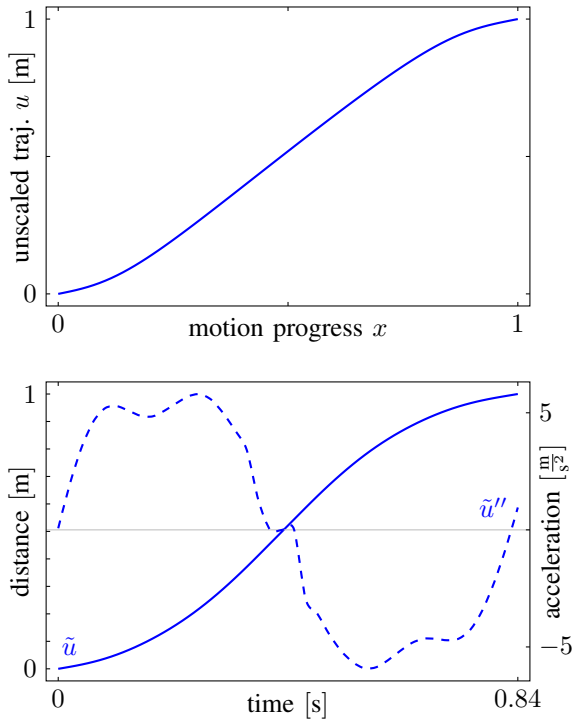

Fig. 4. Example for a one-dimensional motion as generated by the geodesic algorithm with additional time-optimal time-scaling subject to (20) and (21). The geodesic algorithm produced the unscaled trajectory u (top) and the intermediate parameters $p''(0) = 0$, $q''(1) = -0.107$. The subsequent time-scaling using the technique illustrated in Fig. 3 then achieves an overall motion time of $T = 0.84\,\mathrm{s}$ (bottom).

V. Evaluation

Suppose we have planned a joint trajectory and are halfway through the motion. Then, due to some novel information, we decide that we have to abort the motion and return to the initial robot configuration.

It is not always easy to clearly state in what sense a trajectory could be optimal or better than another one. If all we care about is task execution time, maxing out the manipulator's joint limits and simple interpolation in joint space would be unbeatable. If, however, we don't want to give up on the predictability of the motion in task space, we face a motion planning problem subject to non-zero initial boundary conditions.

As an external constraint modeling predictability, we require all portions of a motion to be near direct, i.e. not more than 5% longer than the straight path. For an approach based on polynomial interpolation, this usually meant heavy time-scaling and a near full stop, while with our method the current trajectory could be continued seamlessly. The geodesic planner can avoid a full stop as it allows each individual joint to turn around at a different point in time.

Some typical results are shown in Fig. 5. As the time-scaling depends on the trajectory to which it is applied and possibly to the kinematic model of the manipulator [9], we have omitted it and show the results in terms of the progress variable. Note that the geodesic planner realizes the

velocity boundary conditions automatically even without any time-scaling, while the trajectories obtained by interpolation would still need extra time for acceleration in the beginning and deceleration towards the end.

For systems with additional redundancy (Panda manipulator), the advantage of the geodesic planner not only comes from the ability to handle boundary conditions well, but also from gains over the free part of the motion [15].

The example for the R900 sixx manipulator illustrates how the geodesic planning method accepts some additional joint movement after the midway point compared to the interpolation planner due to the boundary conditions. The tendency to overshoot a given configuration is basically controlled by the choice of the weighting factors in (16).

VI. Numerical Remarks

Since the geodesic equations (1) can only be solved analytically in very special cases, the idea of geodesic path planning hinges on the efficient implementation of methods for their numerical solution.

Our main tool is a so-called relaxation solver similar to the one described in [23]. It is based on an error function that measures how far away from a solution a given function is. An initial guess is then iteratively improved by Newton's method until the error becomes sufficiently small [15].

The computational complexity is quadratic in the number of independent variables. For the largest example in Section V, the boundary condition coordinates for the Panda manipulator, we have $7 + 3$ independent variables. Such a system can still be solved in under 100 ms on an Intel Core i7-4790S CPU. This is fast compared to typical execution times of manipulation tasks.

We would like to point the reader to a potentially unexpected source of error, namely numerical differentiation. Usually, finite differencing works fine, but when iterated too many times, inaccuracies accumulate. In fact, with every finite differencing operation, one looses about a third of the available precision. For the computation of the Christoffel symbols according to (2) we need to take a derivative. Then the relaxation solver will take finite differences in the course of its Newton steps. If the underlying metric was constructed using some kind of differentiation, the relaxation solver starts to become unstable. Therefore, it is crucial to keep the precision high where affordable, e.g. by means of Richardson Extrapolation [23].

VII. Summary and Conclusion

Finding geodesics in a given cost landscape can not only be used to compute joint trajectories directly, but is also a way of viewing more abstract optimization problems. We have used this potential to show that geodesic path planning algorithms are able to optimize the motion of an otherwise free trajectory out of and in to boundary conditions. Our approach is numerically feasible for live re-planning and leverages the possibility of seamlessly joining trajectories together. This leads to faster execution times, especially when combined with an appropriate time-scaling algorithm.

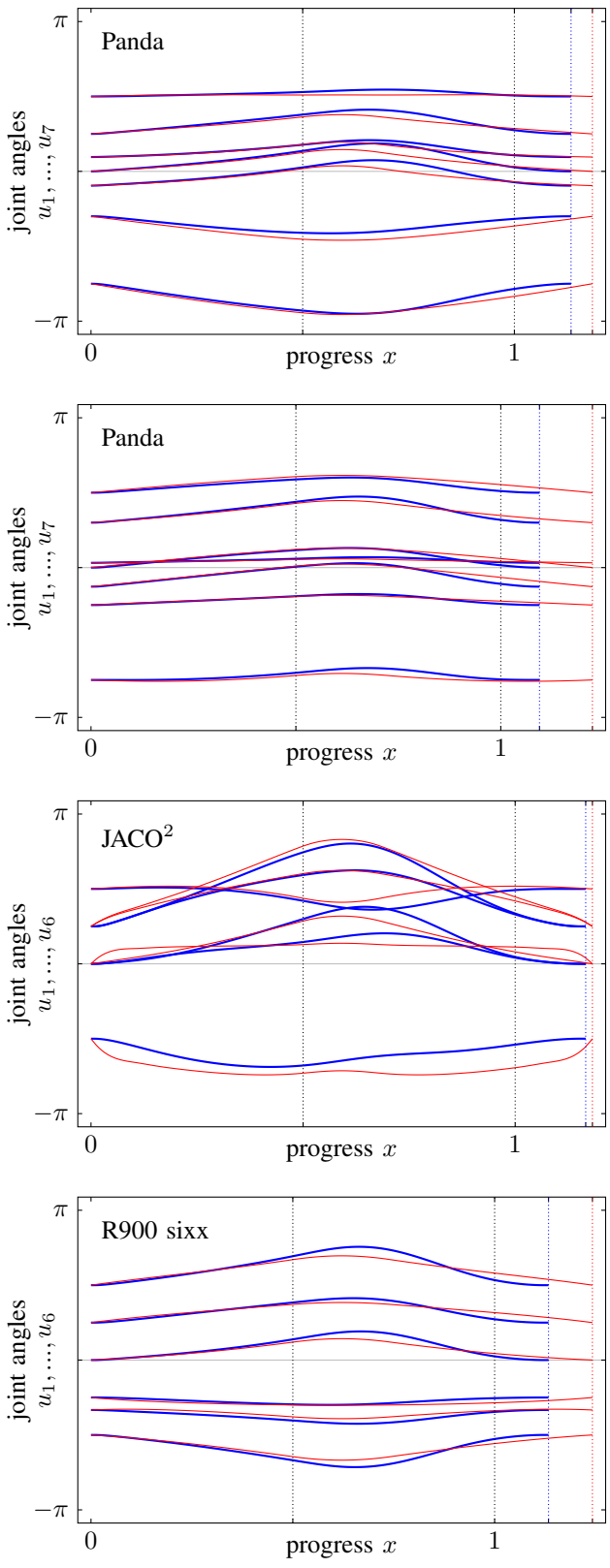

Fig. 5. Comparison of motion planning results for different manipulator geometries (see Fig. 1) in a turn-around scenario: from progress $x = 0.5$ onwards, the manipulator was supposed to return back to the initial configuration. The blue trajectories were obtained by our geodesic planner, the red ones by a standard planner based on interpolation. The geodesic trajectories accomplish the task faster as they avoid a full stop. The progress value $x = 1$ marks the end of the originally planned motion.

REFERENCES

[1] C. Xu, A. Ming, and M. Shimojo, "Optimal Trajectory Generation for Manipulator with Strong Nonlinear Constraints and Multiple Boundary Conditions," *2004 IEEE International Conference on Robotics and Biomimetics*, pp. 192–197, Aug 2004.

[2] C. Lin, P. Chang, and J. Luh, "Formulation and optimization of cubic polynomial joint trajectories for industrial robots," *IEEE Transactions on Automatic Control*, vol. 28, pp. 1066–1074, Dec 1983.

[3] R. H. Taylor, "Planning and execution of straight line manipulator trajectories," *IBM Journal of Research and Development*, vol. 23, pp. 424–436, Jul 1979.

[4] P. Koch and W. Kesheng, "Introduction of b-splines to trajectory planning for robot manipulators." *Modeling, Identification and Control*, vol. 9, pp. 69–80, 1988.

[5] Y.-C. Chen, "Solving robot trajectory planning problems with uniform cubic B-splines," *Optimal Control Applications & Methods*, vol. 12, pp. 247–262, 1991.

[6] E. Papadopoulos, I. Poulakakis, and I. Papadimitriou, "On path planning and obstacle avoidance for nonholonomic platforms with manipulators: A polynomial approach," *International Journal of Robotic Research - IJRR*, vol. 21, pp. 367–383, Apr 2002.

[7] M. Boryga and A. Graboś, "Planning of manipulator motion trajectory with higher-degree polynomials use," *Mechanism and Machine Theory - MECH MACH THEOR*, vol. 44, pp. 1400–1419, Jul 2009.

[8] A. Gasparetto and V. Zanotto, "A new method for smooth trajectory planning of robot manipulators," *Mechanism and Machine Theory*, vol. 42/4, pp. 455–471, 2007.

[9] K. M. Lynch and F. C. Park, "Chapter 9: Trajectory Generation," in *Modern Robotics: Mechanics, Planning, and Control*. USA: Cambridge University Press, 1 ed., 2017

[10] H. Ghanbarpourasl, "Minimum-time path planning for robot manipulators using path parameter optimization with external force and frictions." *Tehnički glasnik*, vol. 13, Mar 2019.

[11] J. Moreno-Valenzuela, *ISA Transactions*, vol. 45/3, pp. 407–418, 2006.

[12] H. Sadeghian, L. Villani, M. Keshmiri and B. Siciliano, "Task-Space Control of Robot Manipulators With Null-Space Compliance," *IEEE Transactions on Robotics*, vol. 30/2, pp. 493–506, Apr 2014.

[13] F. Vigoriti, F. Ruggiero, V. Lippiello and L. Villani, "Control of redundant robot arms with null-space compliance and singularity-free orientation representation," *Robotics and Autonomous Systems.*, vol. 100, pp. 186–193, Feb 2018.

[14] S. Kim, S. Yun and D. Shin, "Numerical Quantification of Controllability in the Null Space for Redundant Manipulators," *Applied Sciences*, vol. 11/13, 2021.

[15] M. Laux and A. Zell, "Robot Arm Motion Planning Based on Geodesics," *2021 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 7585–7591, Jun 2021.

[16] N. Ratliff, M. Toussaint, and S. Schaal, "Understanding the geometry of workspace obstacles in motion optimization," in *2015 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 4202–4209, May 2015.

[17] N. Ratliff, M. Zucker, J. A. Bagnell, and S. Srinivasa, "Chomp: Gradient optimization techniques for efficient motion planning," in *2009 IEEE International Conference on Robotics and Automation*, pp. 489–494, May 2009.

[18] L. Zhang and C. Zhou, "Kuka youbot arm shortest path planning based on geodesics," in *2013 IEEE International Conference on Robotics and Biomimetics (ROBIO)*, pp. 2317–2321, Dec 2013.

[19] C. Youdong, L. Li, and W. Tang, "An improved geodesic algorithm for trajectory planning of multi-joint robots," *International Journal of Advanced Robotic Systems*, vol. 13, Sep 2016.

[20] J. M. Lee, *Riemannian Manifolds: An Introduction to Curvature*. Graduate Texts in Mathematics, Springer New York, 1997.

[21] J. M. Lee, *Manifolds and Differential Geometry*. Graduate studies in mathematics, American Mathematical Society, 2009.

[22] D. S. Ebert, F. K. Musgrave, D. Peachey, K. Perlin, S. Worley, W.R. Mark, and J. C. Hart, "Chapter 2: Building Procedural Textures," in *Texturing and Modeling: A Procedural Approach: Third Edition* Elsevier Inc., 3 ed., 2002.

[23] W. H. Press, S. A. Teukolsky, W. T. Vetterling, and B. P. Flannery, "Chapter 18.3: Relaxation Methods," and "Chapter 5.7: Numerical Derivatives," in *Numerical Recipes 3rd Edition: The Art of Scientific Computing*. USA: Cambridge University Press, 3 ed., 2007.