Spatial Adaption of Robot Trajectories based on Laplacian Trajectory Editing

Thomas Nierhoff · Sandra Hirche · Yoshihiko Nakamura

the date of receipt and acceptance should be inserted later

Abstract Assuming that a robot trajectory is given from a high-level planning or learning mechanism, it needs to be adapted to react to dynamic environment changes. In this article we propose a novel approach to deform trajectories while keeping their local shape similar, which is based on the discrete Laplace-Beltrami operator. The approach can be readily extended and covers multiple deformation techniques including fixed waypoints that must be passed, positional constraints for collision avoidance or a cooperative manipulation scheme for the coordination of multiple robots. Due to its low computational complexity it allows for real-time trajectory deformation both on local and global scale and online adaptation to changed environmental constraints. Simulations illustrate the straightforward combination of the proposed approach with other established trajectory-related methods like artificial potential fields or prioritized inverse kinematics. Experiments with the HRP-4 humanoid successfully demonstrate the applicability in complex daily-life tasks.

Keywords robotics · trajectory adaption · trajectory retargeting · trajectory similarity · local trajectory properties · multiresolution approach · obstacle avoidance

1 Introduction

Multiple approaches exist to find a suitable robot trajectory in a given environment. Depending on the requirements the trajectory is either planned from scratch using for example sampling-based approaches like RRT/RRT* (Lavalle (1998); Karaman and Frazzoli (2011)) or adapted based on a previously learned trajectory. In the latter case Programming by Demonstration (PbD) (Billard et al (2008)) is a standard method for teaching a robot complex movements. Instead of programming every single movement by hand, the robot imitates a demonstrated movement either through physical guidance also known as kinesthetic teaching (Lee and Ott (2011)) or through suitable learning and adaption methods (Schaal (2006); Hoffmann et al (2009)). PbD approaches are also motivated by the goal of a pleasant human-robot-interaction and there are observations in this field based upon motor interference, indicating that similar motions ease the perception of humanoid robots as interaction partners (Kupferberg et al (2011); Kilner et al (2007)).

A major challenge of PbD and planning approaches are its adaption capabilities to changed environments, requiring modifications of the original robot movement. In general, there are two classes of approaches: Direct adaption and indirect adaption through inverse optimization. Direct adaption modifies the existing movement according to the constraints in task space (Pastor et al (2009)). Closely related to this approach are explicit trajectory optimization schemes as CHOMP or TrajOpt (Zucker et al (2013); Schulman et al (2014)). Indirect adaption requires a cost function calculated from a set of demonstrations to be valid over the task space (Levine and Koltun (2012); Mombaur et al (2013)). Whereas indirect adaption methods may have better generalization capabilities, the cost function is difficult to obtain for complex movements and multiple repetitions might be necessary. In this article we focus on direct trajectory adaption. Therefore we assume that a trajectory based on previous demonstrations is given, either from demonstration or a motion planner. The objective is to keep the resulting, de-

T. Nierhoff, S. Hirche
Chair of Information-oriented Control, Technische Universitaet Muenchen, Munich, Germany
E-mail: {tn, hirche}@tum.de

Y. Nakamura
Department of Mechano Informatics, University of Tokyo, Japan
E-mail: nakamura@ynl.t.u-tokyo.ac.jp

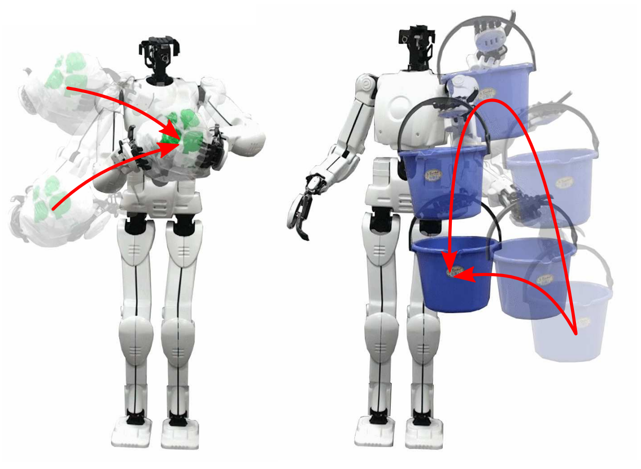


Fig. 1 Application of Laplacian trajectory Editing to deform a given reference motion while maintaining its local shape: Adaption of a bimanual task from different start positions maintaining a fixed spatial distance between both hands (left) and modification of a one-handed pick-and-place task to avoid a possible obstacle (right)

formed trajectory as similar as possible to the original one in terms of position and/or positional differences.

Various non-differential trajectory adaption methods exist in literature, including polynomials, Bézier curves (Hilario et al (2011)), splines, affine transformations (Pham and Nakamura (2013)), Elastic Bands
(Quinlan and Khatib (1993)) or Elastic Strips (Brock and Khatib (2002)). Despite being advantageous for specific applications, they do have individual disadvantages the presented approach overcomes. Polynomials, splines and Bézier curves all suffer from a fixed granularity determined by the number of support points, thus restricting trajectory adaption operations either to the global or local scale. High-order polynomials have the additional problem of overshooting, that are large spatial variations in between two subsequent support points. Concerning affine transformations it has been stated in (Pham (2011)) that three concatenated affine transformations are required in between two fixed sampling points for generic first-order boundary conditions,
hence they produce unintuitive deformed trajectories with straight lines in between. Both Elastic Bands and Elastic Strips share some common properties with the approach presented in this article, yet they aim for shortest paths whereas our approach focuses on shape similarity between original and modified trajectory.

The contribution of the article is the introduction of Laplacian Trajectory Editing (LTE) to deform trajectories and overcome the disadvantages of overshooting and a fixed granularity, thus allowing trajectory modifications both on a local and global scale. The proposed approach provides an intuitive way for deformation. By interpreting a discretized trajectory as an m-dimensional path with associated temporal information, the problem can be transformed to keep the geometric trajectory properties as similar as possible. For this purpose, the discrete Laplace-Beltrami operator encodes intrinsic path properties. The Laplace-Beltrami operator is well-known in the computer graphics community where it is applied to deform (Botsch and Kobbelt (2004a); Sorkine and Cohen-Or (2004)), classify (Luxburg (2007); Reuter et al (2009)) and compress (Karni and Gotsman (2000); Levy (2006)) triangular surfaces meshes. So far, however, the potential of this approach has not yet been exploited for robotics problems. By interpreting a path as an undirected graph, the deformation can be calculated using least squares. Yet the straightforward approach suffers from two drawbacks, a large computational complexity due to a matrix inversion involved (Eck et al (1995); Kobbelt et al (2000)) and the inadequate treatment of large deformations (Lipman et al (2004); Zhou et al (2005)). The method in this article overcomes both challenges by using a multiresolution approach. This way all computationally demanding path modifications are performed only on a reduced set of sampling points. In addition, positional constraints are modified in such a way as to avoid obstacles while maintaining the local shape of the trajectory. The method is also extended to handle the coordinated movements of multiple agents, making cooperative manipulation possible. Simulations compare the approaches presented in this article both in the spatial domain, with respect to computational complexity and with existing state of the art approaches. A demonstration scenario using a HRP-4 robotic platform shows the successful completion of a typical household task involving bimanual pick-and-place operations while avoiding static obstacles 1. .

The remainder of the article is organized as follows: Sec. 2.1 introduces Laplacian Trajectory Editing as a method for direct trajectory adaptation. Extensions are presented in Sec. 3 to solve a set of specific yet commonly occurring problems in robotics. In Sec. 4 both simulations and a robotic experiment validate the proposed approach.

Notation: Throughout the article scalars are written in non-bold letters (e.g a), vectors in bold lower case letters (e.g. $\mathbf{a}$) and matrices in bold capital letters (e.g. $\mathbf{A}$). Accessing a specific element of a matrix/vector is denoted by curly subscript brackets (e.g. $\mathbf{A}_{\{3:\}}$ for the third row, entire column of $\mathbf{A}$).

2 Basics Of Laplacian Trajectory Editing

This section introduces LTE as the underlying framework used throughout the article to adapt and deform discretized trajectories. It also provides an intuitive understanding by relating the abstract Laplace-Beltrami operator to the well-known concept of finite differences along a path.

2.1 General Framework

A trajectory consists of a path $\mathbf{P} = [\mathbf{p}(t_1), \mathbf{p}(t_2), \ldots, \mathbf{p}(t_n)]^T \in \mathbb{R}^{n\times m}$ with m ordered sampling points and corresponding temporal information t_i represented as time $t_i \in \mathbb{R}$, $\mathbf{p}(t_i) \in \mathbb{R}^m$, written $\mathbf{P} = [\mathbf{p}_1, \mathbf{p}_2, \ldots, \mathbf{p}_n]^T$ for simplicity. The path can be interpreted as an undirected graph $\mathscr{G} = (\mathscr{V}, \mathscr{E})$ where each vertex v_i is associated with one sampling point $\mathbf{p}_i$. The neighbor set $\mathscr{N}_i$ of the vertex v_i is the set of all adjacent vertices v_j and the edge set is defined as $\mathscr{E} = \{e_{ij}\}, i, j \in \{1, .., n\}$ with

$$e_{ij} = \begin{cases} w_{ij} & \text{if } j \in \mathscr{N}_i, \\ 0 & \text{otherwise.} \end{cases} \qquad (1)$$

and the edge weight w_{ij}. Multiple weighting schemes for w_{ij} exist in literature, the most prominent ones being uniform umbrella weights $w_{ij} = 1$ working best for regular-shaped meshes and scale-dependent umbrella weights $w_{ij} = \frac{1}{\|\mathbf{p}_i - \mathbf{p}_j\|_2}$ to compensate for irregular-shaped meshes (Desbrun et al (1999)).

Rather than working in absolute Cartesian coordinates, the discrete Laplace-Beltrami operator specifies the local path properties, called Laplacian coordinates $\boldsymbol{\delta}_i$ (Lipman et al (2005)). For vertex v_i, this results in

$$\boldsymbol{\delta}_i = \sum_{j\in\mathscr{N}_i} \frac{w_{ij}}{\sum\limits_{j\in\mathscr{N}_i} w_{ij}} (\mathbf{p}_i - \mathbf{p}_j),$$

Written in matrix form, it turns out that the discrete Laplace-Beltrami operator resembles the graph Laplacian matrix $\mathbf{L} \in \mathbb{R}^{n\times n}$ encoding the topology of the graph as

$$\mathbf{L}_{\{ij\}} = \begin{cases} 1 & \text{if } i = j, \\ -\frac{w_{ij}}{\sum\limits_{j\in\mathscr{N}_i} w_{ij}} & \text{if } j \in \mathscr{N}_i, \\ 0 & \text{otherwise.} \end{cases}$$

Using uniform umbrella weights, one obtains the typical strucure for paths as

$$\mathbf{L} = \frac{1}{2} \begin{bmatrix} 2 & -2 & & & & & \\ -1 & 2 & -1 & & & \mathbf{0} & \\ & -1 & 2 & -1 & & & \\ & & \ddots & \ddots & \ddots & & \\ & & & -1 & 2 & -1 & \\ & \mathbf{0} & & & -1 & 2 & -1 \\ & & & & & -2 & 2 \end{bmatrix}.$$

When concatenating all Laplacian coordinates $\boldsymbol{\delta}_i$ into a single matrix $\boldsymbol{\Delta} = [\boldsymbol{\delta}_1, \boldsymbol{\delta}_2, \ldots, \boldsymbol{\delta}_n]^T$, one can thus write

$$\mathbf{L}\mathbf{P} = \boldsymbol{\Delta}. \qquad (2)$$

As the equation system in (2) is underdetermined, i.e. the Laplacian matrix is singular, the Cartesian coordinates $\mathbf{P}$ cannot be uniquely calculated using the inverse of $\mathbf{L}$ when given only the Laplacian coordinates. However, by specifying additional constraints in the form

$$\bar{\mathbf{P}}\mathbf{P} = \bar{\mathbf{C}}, \qquad (3)$$

$$rank\left(\begin{bmatrix} \mathbf{L} \\ \bar{\mathbf{P}} \end{bmatrix}\right) = n,$$

the resulting concatenated equation system

$$\begin{bmatrix} \mathbf{L} \\ \bar{\mathbf{P}} \end{bmatrix} \mathbf{P}_s = \begin{bmatrix} \boldsymbol{\Delta} \\ \bar{\mathbf{C}} \end{bmatrix}, \qquad (4)$$

can be solved for the trajectory $\mathbf{P}_s = [\mathbf{p}_{s,1}, \ldots, \mathbf{p}_{s,n}]^T \in \mathbb{R}^{n\times m}$

$$\mathbf{P}_s = \begin{bmatrix} \mathbf{L} \\ \bar{\mathbf{P}} \end{bmatrix}^+ \begin{bmatrix} \boldsymbol{\Delta} \\ \bar{\mathbf{C}} \end{bmatrix}, \qquad (5)$$

using least squares. Note that due to the least squares approach $\mathbf{P}_s$ and $\mathbf{P}$ generally differ from each other, see Fig. 2. In addition, the constraints in $\bar{\mathbf{C}}, \bar{\mathbf{P}}$ are only approximately met.

Only few viable options for $\bar{\mathbf{P}}$ with a physical meaning are known so far. The first and probably most important one are positional constraints of the form

$$\mathbf{p}_i = \mathbf{c}_i,$$

pinning a sampling point $\mathbf{p}_i$ to a desired position $\mathbf{c}_i$. By introducing the weighting factors $\omega = \{\omega_i, \omega_{i,1}, \omega_{i,2}, \ldots\}$, $i = 1, \ldots, n$ determining the importance of the corresponding constraint with respect to the Laplacian coordinates $\boldsymbol{\delta}_i$, it can be rewritten as

$$\omega_i \mathbf{p}_i = \omega_i \mathbf{c}_i, \qquad (6)$$

see the Matlab example in (Nierhoff (2013a)).

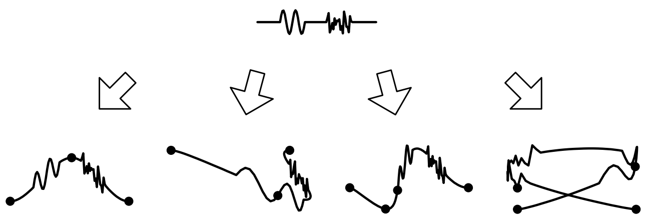


Fig. 2 Various possible path deformations by applying positional constraints to individual sampling points (round dots)

Another option are first order finite difference constraints of the form

$$\mathbf{p}_{i+1} - \mathbf{p}_i = \mathbf{c}_{i,1},$$

resulting in a fixed spatial difference between two sampling points $\mathbf{p}_j$ and $\mathbf{p}_i$ along the path. With the weighting factor $\omega_{i,1}$ it is rewritten as

$$\omega_{i,1}(\mathbf{p}_{i+1} - \mathbf{p}_i) = \omega_{i,1}\mathbf{c}_{i,1}. \qquad (7)$$

This scheme can be extended to higher order finite differences. Hence it is for second order central finite differences

$$\mathbf{p}_{i+1} - 2\mathbf{p}_j + \mathbf{p}_{i-1} = \mathbf{c}_{i,2},$$

and with the weighting factor $\omega_{i,2}$

$$\omega_{i,2}(\mathbf{p}_{i+1} - 2\mathbf{p}_j + \mathbf{p}_{i-1}) = \omega_{i,2}\mathbf{c}_{i,2}. \tag{8}$$

Writing the unweighted first n_z finite differences in matrix form

$$\begin{bmatrix} 1 & & & & & \\ -1 & 1 & & \mathbf{0} & & \\ 1 & -2 & 1 & & & \\ -1 & 3 & -3 & 1 & & \\ 1 & -4 & 6 & -4 & 1 & \\ \vdots & \vdots & \vdots & \vdots & \vdots & \ddots \end{bmatrix},$$

one sees that they are linearly independent. Hence any row of $\overline{\mathbf{P}}$ with n_z nonzero entries can be represented as a weighted sum of the first n_z order derivatives, thus providing an intuitive understanding of an arbitrary, non-zero row of $\overline{\mathbf{P}}$.

Remark 1 Various definitions of discrete Laplace operators exist in literature: Being mainly based upon the connectivity of the underlying graph, the graph Laplace operator belongs to the wider class of *combinatorial mesh Laplacians* (Zhang et al (2010)). Another class are *geometric mesh Laplacians*, explicitly taking into account the underlying Riemannian geometry (Strichartz (1983); Meyer et al (2002); Dierkes et al (2010)). Even if they do capture the geometric properties better, they are only defined on triangle meshes and thus not straightforwardly applicable to paths.

2.2 Interpretation

Despite looking abstract at first glance, there exist intuitive geometric and physical interpretations of LTE.

Given a continuous path $\boldsymbol{\Pi}(s) \in \mathbb{R}^3$ parameterized by arc length s, the Frenet-Serret formula describe the local curvature κ and the local torsion τ. By introducing the unit tangent vector $\mathbf{t}$, the normal unit vector $\mathbf{n}$ and the binormal unit vector $\mathbf{b} = \mathbf{t} \times \mathbf{n}$ with $\times$ as the cross product for a given point along the path, one obtains (Do-Carmo (1976))

$$\frac{d\mathbf{t}}{d\mathbf{s}} = \kappa\mathbf{n},$$
$$\frac{d\mathbf{n}}{d\mathbf{s}} = -\kappa\mathbf{t} + \tau\mathbf{b},$$
$$\frac{d\mathbf{b}}{d\mathbf{s}} = -\tau\mathbf{n}.$$

Then κ can be calculated as

$$\kappa = \left\| \frac{d\mathbf{t}}{ds} \right\| = \left\| \frac{d^2\boldsymbol{\Pi}(s)}{ds^2} \right\|. \tag{9}$$

If the path is not continuous but discrete, one can represent the continuous path $\boldsymbol{\Pi}(s)$ by its discretization $\mathbf{P}$ and $m = 3$. Then the central difference approximation of (9) is

$$\left\| \frac{d^2\mathbf{p}(s)}{d\mathbf{s}^2} \right\| = \frac{\mathbf{p}(s+h) - 2\mathbf{p}(s) + \mathbf{p}(s-h)}{h^2}, \tag{10}$$

with step length h. Assuming the sampling points are equidistantly spaced with distance h, (10) is rewritten as

$$\left\| \frac{d^2\boldsymbol{p}_i}{ds^2} \right\| = \frac{\mathbf{p}_{i+1} - 2\mathbf{p}_i + \mathbf{p}_{i-1}}{h^2}, \quad i = 2,3,\ldots,n-1. \tag{11}$$

This formula is closely related to the Laplacian coordinates (Taubin (1995)), that is

$$\boldsymbol{\delta}_i = \frac{\mathbf{p}_{i+1} - 2\mathbf{p}_i + \mathbf{p}_{i-1}}{-2}, \quad i = 2,3,\ldots,n-1, \tag{12}$$

differing from (11) only in terms of the scaling factor ($\frac{1}{h^2}$ vs. $\frac{1}{-2}$).

Until now only the spatial domain is considered. In reality however, e.g. when recording a movement with a motion capture system, subsequent trajectory points are typically sampled at a constant temporal rate, hence spaced rather equitemporally than equidistantly. With $\Delta t = t_i - t_{i-1}$ as the temporal difference between any two subsequent sampling points the acceleration $\ddot{\mathbf{p}}$ along the trajectory is

$$\ddot{\mathbf{p}}_i = \frac{\mathbf{p}_{i+1} - 2\mathbf{p}_i + \mathbf{p}_{i-1}}{\Delta t^2}, \quad i = 2,3,\ldots,n-1,$$

differing from (12) again only in terms of the scaling factor ($\frac{1}{\Delta t^2}$ vs. $\frac{1}{-2}$). In case two subsequent sampling points are fixed - see (7) - it is interpreted as a velocity constraint. In case of three subsequent sampling points, (8) corresponds to an acceleration constraint.

Looking at the weighting factors ω in (6)-(8) they determine the importance of the additional constraints in $\overline{\mathbf{P}}$ with respect to $\mathbf{L}$. Speaking loosely, they define the admissible amount of deformation. For $\omega \approx 0$ the path is deformed just insignificantly and the constraints specified in $\overline{\mathbf{P}}$ and $\overline{\mathbf{C}}$ are hardly met. On the other hand, weighting factors $\omega \gg 0$ prioritize the constraints, leading to larger deformation.

Numerous variations exist: The Laplacian matrix $\mathbf{L}$ can also be constructed based on the first/third order derivatives (a different name should be used in this case). For a given straight line $\mathbf{P}$ consisting of equidistantly and equitemporally spaced sampling points, the deformed trajectory $\overline{\mathbf{P}}$ resembles a minimum velocity/jerk trajectory. This is consistent with findings about minimum jerk trajectories for human movement generation (Flash and Hogan (1985)).

3 Extension to Laplacian Trajectory Editing

Having introduced the basic concepts of LTE, this article continues with several improvements over the original approach in Sec. 2.1, making it applicable to a wider class of trajectory retargeting problems arising in robotics. As mentioned in the introduction, the approach presented so far suffers from two main drawbacks, a high computational complexity due to the matrix inversion and the incapability to handle nonlinear deformation effects. To overcome both challenges, a multiresolution approach is presented. The severity of not handling nonlinear deformation effects becomes clear when looking at the example in Fig. 3. A handwritten word (“Hello”) is deformed by fixing three sampling points through positional constraints. Whereas the original approach of Sec. 2.1 obviously has low similarity, the multiresolution approach resembles the word well. In addition,

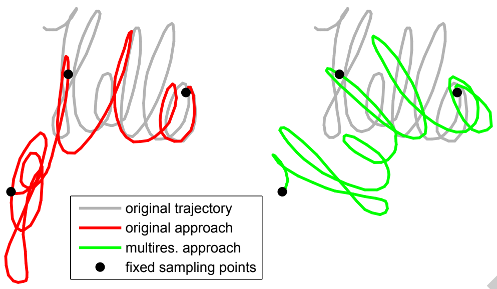


Fig. 3 Comparison between the original and the multiresolution approach

the article shows novel extensions for reactive collision avoidance, cooperative manipulation of multiple manipulators or endeffectors and the inclusion of kinematic constraints for execution on a real robot.

3.1 Arun’s Method For Handling Nonlinear Deformation Effects

As stated in the introduction, nonlinear deformation effects are not handled properly by LTE. Probably the most frequent nonlinear effect occurring during path adaption are rotations. In order to cope with them on a local path scale, the method of Arun (Arun et al (1987); Umeyama (1991); Nierhoff and Hirche (2012)) is combined with LTE. Its core concept can be recapitulated as follows: Assume that one is given two sets of points, namely $\mathbf{P}_r = [\mathbf{p}_{r,1},\ldots,\mathbf{p}_{r,k}]^T \in \mathbb{R}^{k\times m}$ and $\mathbf{P}_d = [\mathbf{p}_{d,1},\ldots,\mathbf{p}_{d,k}]^T \in \mathbb{R}^{k\times m}$ that should be matched using the homogeneous transformation

$$\mathbf{p}_{d,i} = c\mathbf{R}\mathbf{p}_{r,i} + \mathbf{t} \qquad \forall i = 1,\ldots,k,$$

with constant c as a scalar scaling factor, $\mathbf{R} \in \mathbb{R}^{m\times m}$ as a rotation matrix and $\mathbf{t} \in \mathbb{R}^n$ as a translational vector. Because the matching is usually not perfect, one has to find an affine transformation that matches both $\mathbf{P}_r$ and $\mathbf{P}_d$ ”as good as possible”. The problem can be reformulated as a minimization problem using the error term $\mathbf{n}_o$ as

$$\mathbf{n}_o = \sum_{i=1}^{k} \left\|\mathbf{p}_{d,i} - (c\mathbf{R}\mathbf{p}_{r,i} + \mathbf{t})\right\|^2.$$

The elements of the homogeneous transformation can be calculated using Singular Value Decomposition: Let $\mathbf{Q} \in \mathbb{R}^{m\times m}$ be the covariance matrix as

$$\mathbf{Q} = \frac{1}{k}\sum_{i=1}^{k}(\mathbf{p}_{r,i} - \bar{\mathbf{p}}_r)(\mathbf{p}_{d,i} - \bar{\mathbf{p}}_d)^T,$$

with $\bar{\mathbf{p}}_r$ as the centroid of $\mathbf{P}_r$ and $\bar{\mathbf{p}}_d$ as the centroid of $\mathbf{P}_d$

$$\bar{\mathbf{p}}_r = \frac{1}{k}\sum_{i=1}^{k}\mathbf{p}_{r,i}, \qquad \bar{\mathbf{p}}_d = \frac{1}{k}\sum_{i=1}^{k}\mathbf{p}_{d,i}.$$

Similarly, the variance σ_s^2 is calculated as

$$\sigma_s^2 = \frac{1}{k}\sum_{i=1}^{k}\|\mathbf{p}'_{ri}\|^2.$$

The SVD of $\mathbf{Q}$ is calculated such that

$$\mathbf{Q} = \mathbf{U}\mathbf{S}\mathbf{V}^T. \tag{13}$$

Then c, $\mathbf{R}$ and $\mathbf{t}$ can be computed as

$$\mathbf{R} = \mathbf{V}\mathbf{S}'\mathbf{U}^T,$$
$$c = \frac{1}{\sigma_s^2}tr(\mathbf{S}\mathbf{S}'),$$
$$\mathbf{t} = \bar{\mathbf{p}}_d - c\mathbf{R}\bar{\mathbf{p}}_s.$$

with $\mathbf{S}'$ preventing mirrored mappings

$$\mathbf{S}' = \begin{cases} \mathbf{I} \text{ if } \det(U)\det(V) = 1, \\ diag(1,\ldots,1,-1) \text{ if } \det(U)\det(V) = -1. \end{cases}$$

As shown in (Sorkine and Alexa (2007)), the method can be adapted with $c = 1$ to rotate the Laplacian coordinates individually for every sampling point. When applied to paths, this results in new Laplacian coordinates $\hat{\boldsymbol{\delta}}_i$ as

$$\hat{\boldsymbol{\delta}}_i = \mathbf{R}_i\boldsymbol{\delta}_i,$$

with the rotation matrix $\mathbf{R}_i$ based upon the sampling points’ position of original and deformed path. For the Laplacian coordinate $\boldsymbol{\delta}_i$ the two sets of sampling points are $\mathbf{P}_r = [\mathbf{p}_i - \mathbf{p}_{i-1}, \mathbf{p}_i - \mathbf{p}_{i+1}]^T$ and $\mathbf{P}_d = [\mathbf{p}_{s,i} - \mathbf{p}_{s,i-1}, \mathbf{p}_{s,i} - \mathbf{p}_{s,i+1}]^T$. Whereas the SVD solution can be used in arbitrary dimensions and also for higher order derivatives, we realize that both $\mathbf{P}_r$ and $\mathbf{P}_d$ consist only of two vectors when using Laplacian coordinates. Then an optimal rotation $\mathbf{R}_i$ can be calculated in 2D and 3D using basic geometry. Although the latter has a lower computational complexity, it is only marginally faster to compute in Matlab as there are highly optimized routines to calculate the SVD.

3.2 Multiresolution Approach For Possible Online Application

Whereas the method in the previous section can handle non-linear deformation effects, it is also slow as every Laplacian coordinate has to be rotated individually. In combination with the matrix inversion of LTE the method is inapplicable for time-critical applications. To overcome the computational bottleneck, we propose a multiresolution approach together with a detailed evaluation both in the spatial and the temporal domain and an extension for fast path deformation in 3D. By downsampling the path first, Arun's method will be applied only to a reduced number of sampling points during the adaption step. The Laplacian coordinates of all remaining sampling points are then interpolated in a final reconstruction step, thus speeding up calculation 4. A preliminary version of this approach is presented in (Nierhoff et al (2013)), consisting of three steps:

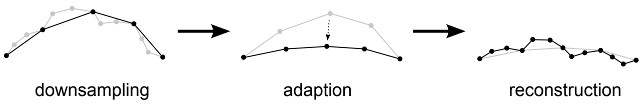


Fig. 4 Overview of the multiresolution approach

3.2.1 Downsampling

The goal of the downsampling step is to find a reduced set of so-called support sampling points $\mathbf{P}' = [\mathbf{p}'_1, \mathbf{p}'_2, \dots, \mathbf{p}'_{n'}]^T \in \mathbb{R}^{n' \times m}$, subject to

$$\mathbf{P}' \in \mathbf{P}, \tag{14}$$

$$\min \sum_{i=2}^{n'-1} (\|\mathbf{p}'_i - \mathbf{p}'_{i-1}\|_2 - \|\mathbf{p}'_{i+1} - \mathbf{p}'_i\|_2)^2. \tag{15}$$

The first condition enables the remaining sampling points to be directly interpolated during the reconstruction step based on the deformed support sampling points. The second condition is necessary as combinatorial mesh Laplacians do not take into account the geometry of the graph and thus rely on a regular mesh structure ($\|\mathbf{p}'_i - \mathbf{p}'_{i-1}\|_2 = \|\mathbf{p}'_{i+1} - \mathbf{p}'_i\|_2$) for a good approximation (Taubin (1995); Botsch and Kobbelt (2004b); Wardetzky et al (2007)). It is

$$\mathfrak{F}(\Delta f(x)) = \mathfrak{F}(\nabla^2 f(x)) \propto u^2 F(u),$$

with $\mathfrak{F}$ as the Fourier transform from the spatial domain $f(x)$ to the frequency domain $F(u)$. Hence the Laplace operator is heavily influenced by high-frequency noise. To increase robustness, the path is smoothed in the spatial/frequency domain using a spatial moving average filter (SMA) respectively a Fast Fourier Transform (FFT). In the limit, this results in a regular mesh structure $\|\mathbf{p}'_i - \mathbf{p}'_{i-1}\|_2 = \|\mathbf{p}'_{i+1} - \mathbf{p}'_i\|_2 = const$. If we impose additional constraints on specific sampling points, they must be included in $\mathbf{P}'$ as well.

3.2.2 Adaption

During the adaption step, a two-staged approach modifies the shape of the downsampled trajectory $\mathbf{P}'$ (Sorkine and Alexa (2007)). For the remainder of this section, we will mark the support sampling points of iteration *it* with $\mathbf{P}'_{it}$ and the Laplacian coordinates of the support sampling points with $\boldsymbol{\Delta}'_{it}$. In the first step, the resulting LTE equation system is solved for $\mathbf{P}'_{it+1}$ as

$$\mathbf{P}'_{it+1} = \begin{bmatrix} \mathbf{L} \\ \omega\bar{\mathbf{P}} \end{bmatrix}^+ \begin{bmatrix} \boldsymbol{\Delta}'_{it} \\ \omega\bar{\mathbf{C}} \end{bmatrix}.$$

In the second step, the elements of $\boldsymbol{\Delta}'_{it}$ are updated individually for every support sampling using Arun's method based on $\mathbf{P}'_{it}$ and $\mathbf{P}'_{it+1}$, resulting in $\boldsymbol{\Delta}'_{it+1}$.

After l iterations, this results in the downsampled path $\mathbf{P}'_l = [\mathbf{p}'_{l,1}, \dots, \mathbf{p}'_{l,n'}]^T$. The final step of the adaption is to calculate a rotation matrix $\mathbf{R}'_k, \;\; k = \{1, 2, \dots, n'\}$ for every support sampling point, measuring the rotation between $\mathbf{P}'_1$ and $\mathbf{P}'_l$ using Arun's method.

3.2.3 Reconstruction using LTE

After deformation of the downsampled path $\mathbf{P}'$, the position of all remaining sampling points must be reconstructed, resulting in $\mathbf{P}_s$. Let the k-th path segment of $\mathbf{P}_s$, named $\mathbf{P}_{s,k}$, be defined as the set of all sampling points between $\mathbf{p}'_{l,k}$ and $\mathbf{p}'_{l,k+1}$, with ”between” referring to the graph structure and not the spatial domain. Then the position of all sampling points of the k-th trajectory segment is calculated as

$$\mathbf{P}_{s,k} = \begin{bmatrix} \mathbf{L}_{s,k} \\ \bar{\mathbf{P}}_{s,k} \end{bmatrix}^+ \begin{bmatrix} \boldsymbol{\Delta}_{s,k} \\ \bar{\mathbf{C}}_{s,k} \end{bmatrix},$$

with $\mathbf{L}_{s,k}$ as the Laplacian matrix for the k-th trajectory segment, $\boldsymbol{\Delta}_{s,k}$ containing the rotated Laplacian coordinates and suitable boundary constraints encoded in $\bar{\mathbf{P}}_{s,k}, \bar{\mathbf{C}}_{s,k}$. The matrix $\mathbf{L}_{s,k}$ is simply a submatrix of $\mathbf{L}$ with similar structure. The boundary constraints in $\bar{\mathbf{P}}_{s,k}, \bar{\mathbf{C}}_{s,k}$ are calculated by fixing the first and last sampling point of every trajectory segment, that is $\mathbf{p}'_{l,k}$ and $\mathbf{p}'_{l,k+1}$. The elements of $\boldsymbol{\Delta}_{s,k}$ are calculated by linearly interpolating the differential coordinates of the path segment based on $\mathbf{R}'_k$ and $\mathbf{R}'_{k+1}$. Depending on the used representation either axis/angle based interpolation or SLERP/NLERP (Shoemake (1985)) might be better suited for up to three dimensions.

3.2.4 Reconstruction using Affine Transformations

Another option for reconstruction are affine transformations $\tilde{\mathbf{p}} = \mathscr{M}(\mathbf{p})$ as presented in (Pham and Nakamura (2013)). For a path consisting of multiple sampling points $\mathbf{p}_i$ they are of the general form

$$\tilde{\mathbf{p}}_i = \mathbf{M}\mathbf{p}_i + \mathbf{w},$$

with $\mathscr{M} = \{\mathbf{M}, \mathbf{w}\}, \mathbf{M} \in \mathbb{R}^{m\times m}, \mathbf{w} \in \mathbb{R}^m$. Boundary constraints for discrete paths ensuring C^q continuity at the beginning/end of the path can be incorporated by fixing the first respectively last $q+1$ sampling points. Hence for C^0 continuity it is

$$\tilde{\mathbf{p}}_1 = \mathbf{M}\mathbf{p}_1 + \mathbf{w},$$
$$\tilde{\mathbf{p}}_n = \mathbf{M}\mathbf{p}_n + \mathbf{w},$$

and for C^1 continuity it is in addition

$$\tilde{\mathbf{p}}_2 = \mathbf{M}\mathbf{p}_2 + \mathbf{w},$$
$$\tilde{\mathbf{p}}_{n-1} = \mathbf{M}\mathbf{p}_{n-1} + \mathbf{w},$$

resulting in the linear equation system

$$\mathbf{V}\mathbf{b} = \mathbf{c}, \tag{16}$$

with $\mathbf{V} \in \mathbb{R}^{4m\times(m^2+m)}$, $\mathbf{b} \in \mathbb{R}^{m^2+m}$ containing the elements of $\mathbf{M}, \mathbf{w}$ and $\mathbf{c} = \left[\tilde{\mathbf{p}}_1^T, \tilde{\mathbf{p}}_2^T, \tilde{\mathbf{p}}_{n-1}^T, \tilde{\mathbf{p}}_n^T\right]^T$. It is clear that C^q continuity can be only achieved if there are less or exactly as many boundary conditions as free variables in $\mathscr{M}$. As such for C^0 continuity at least one dimension and for C^1 continuity at least three dimensions are required. The latter case is for the remainder of this article.

Affine transformations are advantageous from a computational perspective as only a small-sized linear equation system (12 unknowns in $\mathbf{M}_k, \mathbf{w}_k$ for the k-th path segment based upon the four boundary sampling points of every path segment) has to be solved. Yet the method suffers from instabilities if the matrix $\mathbf{V}_k$ for the k-th path segment becomes singular. To prevent this, the four boundary sampling points must span a three-dimensional space. By defining the condition numbers $\kappa(\mathbf{V}_k)$ and $\kappa(\mathbf{V}_{k-1})$ based on (16) for every path segment, we reformulate (14)-(15) as

$$\mathbf{P}' \in \mathbf{P}, \tag{17}$$

$$\min\ f_1 \sum_{i=2}^{n'-1} (\|\mathbf{p}'_i - \mathbf{p}'_{i-1}\|_2 - \|\mathbf{p}'_{i+1} - \mathbf{p}'_i\|_2)^2 + \tag{18}$$
$$f_2 \sum_{i=1}^{n'-1} \kappa(\mathbf{V}_k),$$

with constants f_1 and f_2 ensuring a good tradeoff between a regularly shaped mesh and a non-degenerated solution.

3.3 Positional Constraints for Obstacle Avoidance

An extension of LTE allows reactive obstacle avoidance in task space. The method presented in this chapter is an extension of the work in Nierhoff et al (2013), providing higher robustness and allowing larger deformations. When using positional constraints with weighting factors $\omega_i \gg 1$ the small deviation from the desired position due to the least squares approach constitutes an inevitable yet often negligible error. In contrast this section focuses on low-weighted positional constraints, i.e. $\omega_i \approx 1$ or $\omega_i < 1$ with a non-negligible error. Differing from positional constraints with $\omega_i \gg 1$ imposed generally on a few vertices only, low-weighted positional constraints impose positional constraints on *all* vertices. The desired behaviour of avoiding dynamic obstacles while maintaining the original path shape in a least squares sense is then achieved by smoothly varying both the positional constraint matrix $\bar{\mathbf{P}}$ and the positional constraint $\bar{\mathbf{C}}$ along the path. The presented approach consists of three superposed parts

1. A repulsive positional constraint for obstacle avoidance
2. An attractive positional constraint pulling the path back to its original position
3. Laplacian coordinates maintaining the local path shape

Given an obstacle Ω with with uniquely defined minimum distance $\mathbf{d}_i$ to each sampling point $\mathbf{p}_i$ of the path, the obstacle exerts a repulsive positional constraint on $\mathbf{p}_i$ according to

$$\bar{\mathbf{C}}_{1\{i:\}} = \beta\left(\mathbf{p}_i + \alpha\frac{\mathbf{d}_i}{\|\mathbf{d}_i\|_2^\gamma}\right), \tag{19}$$
$$\bar{\mathbf{P}}_1 = \mathrm{diag}(\beta, \dots, \beta),$$

with constants α, β, γ. The attractive positional constraint pulling the path back is described using each sampling point's original position $\mathbf{p}_{o,i}$ before deformation

$$\bar{\mathbf{C}}_{2\{i:\}} = \delta\mathbf{p}_{o,i}, \tag{20}$$
$$\bar{\mathbf{P}}_2 = \mathrm{diag}(\delta, \dots, \delta),$$

with constant δ. By concatenating the conditions in (19) and (20) into $\bar{\mathbf{C}}$ and $\bar{\mathbf{P}}$ as

$$\bar{\mathbf{C}} = \begin{bmatrix}\bar{\mathbf{C}}_1\\ \bar{\mathbf{C}}_2\end{bmatrix}, \quad \bar{\mathbf{P}} = \begin{bmatrix}\bar{\mathbf{P}}_1\\ \bar{\mathbf{P}}_2\end{bmatrix} \tag{21}$$

and solving (5) for $\mathbf{P}_s$, the desired behavior can be achieved. Some sample code is publicly available under (Nierhoff (2013b)).

Unfortunately the approach becomes unstable for large deformations. In this case the force exerted by the attractive positional constraint and the Laplacian framework that pulls the path back to its original position gets too strong, causing small obstacles to slip through the deformed path. A modified and more stable version evaluates the shortest distance $\hat{\mathbf{d}}_i$ not between obstacle Ω and every sampling point

$\mathbf{p}_i$ as in (19), but between obstacle Ω and every line segment $\mathbf{p}_i + \alpha_L(\mathbf{p}_{i+1} - \mathbf{p}_i)$, $\alpha_L \in [0,1]$.

$$\bar{\mathbf{C}}_{1\{i:\}} = \mathbf{p}_i + \alpha \frac{\hat{\mathbf{d}}_i}{\|\hat{\mathbf{d}}_i\|_2^\gamma}, \qquad (22)$$
$$\bar{\mathbf{C}}_{1\{i+1:\}} = \mathbf{p}_{i+1} + \alpha \frac{\hat{\mathbf{d}}_i}{\|\hat{\mathbf{d}}_i\|_2^\gamma},$$

to prevent small obstacles from slipping through the path if two subsequent sampling point are far apart. Moreover, the attractive positional constraint is scaled with a distance-dependent factor as

$$\bar{\mathbf{C}}_{2\{i:\}} = \mathbf{p}_i + \varepsilon \frac{\mathbf{p}_{o,i} - \mathbf{p}_i}{1 + \|\mathbf{p}_{o,i} - \mathbf{p}_i\|_2}, \qquad (23)$$

with constant ε, imposing upper bounds to the absolute value of the attractive positional constraint. To reduce the effect of the Laplacian coordinates, let $\boldsymbol{\Delta}_o$ and $\boldsymbol{\Delta}_d$ be the matrices containing the Laplacian coordinates of the original respectively currently deformed path. Then the resulting matrix $\boldsymbol{\Delta}$ is calculated as

$$\boldsymbol{\Delta} = (1-\zeta)\boldsymbol{\Delta}_o + \zeta\boldsymbol{\Delta}_d, \qquad (24)$$

with constant $\zeta \in [0,1]$. For $\zeta = 0$ the method is similar to the unmodified version and for $\zeta = 1$ only the Laplacian coordinates of the current path are considered, effectively disabling the convergence back to the original path position. Both equations (23) and (24) diminish the influence of the attractive positional constraint respectively the Laplacian coordinates, thus increasing robustness at the cost of an increased computational complexity and slower convergence speed back to the original path position.

A small scenario illustrating the obstacle avoidance capabilities of LTE in combination with low-weighted positional constraints is depicted in Fig. 5. With increasing number of obstacles (red circles), the initially sinusoidal path deforms more and more to avoid all obstacles. We compare two trajectories, corresponding to the basic and the modified version.

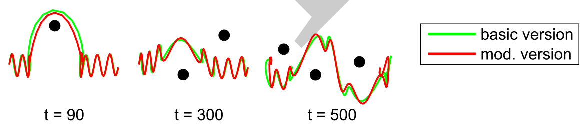


Fig. 5 Obstacle avoidance scenario with spatial plots at different time steps t

3.4 Cooperative Manipulation

So far we only considered single paths. Yet in many scenarios like bimanual manipulation it is necessary to adapt the movement of two or more agents/manipulators in an adequate manner. LTE is adapted in this section to take such kind of constraints into account. Note that we are now considering trajectories, thus corresponding agent positions must match both in the spatial and temporal domain.

Given two agents' trajectories $\mathbf{P}_1 = [\mathbf{p}_{1,1}, \ldots, \mathbf{p}_{1,n}]^T$ and $\mathbf{P}_2 = [\mathbf{p}_{2,1}, \ldots, \mathbf{p}_{2,n}]^T$ with corresponding equation system $\mathbf{L}_1\mathbf{P}_{s1} = \boldsymbol{\Delta}_1$, $\mathbf{L}_2\mathbf{P}_{s2} = \boldsymbol{\Delta}_2$ and same timing $t_i(\mathbf{p}_{1,i}) = t_i(\mathbf{p}_{2,i})$, it can be rewritten as

$$\begin{bmatrix} \mathbf{L}_1 & 0 \\ 0 & \mathbf{L}_2 \end{bmatrix} \begin{bmatrix} \mathbf{P}_{s1} \\ \mathbf{P}_{s2} \end{bmatrix} = \begin{bmatrix} \boldsymbol{\Delta}_1 \\ \boldsymbol{\Delta}_2 \end{bmatrix}.$$

To maintain a defined spatial relation, we expand the equation system similar to (4) as

$$\begin{bmatrix} \mathbf{L}_1 & 0 \\ 0 & \mathbf{L}_2 \\ \bar{\mathbf{P}}_- & \bar{\mathbf{P}}_+ \end{bmatrix} \begin{bmatrix} \mathbf{P}_{s1} \\ \mathbf{P}_{s2} \end{bmatrix} = \begin{bmatrix} \boldsymbol{\Delta}_1 \\ \boldsymbol{\Delta}_2 \\ \bar{\mathbf{C}} \end{bmatrix}, \qquad (25)$$

with the definition of the matrices $\bar{\mathbf{P}}_-, \bar{\mathbf{P}}_+ \in \mathscr{R}^{n\times n}$ and $\bar{\mathbf{C}}$ as

$$\bar{\mathbf{P}}_- = \mathrm{diag}(-\omega_1, \ldots, -\omega_n),$$
$$\bar{\mathbf{P}}_+ = \mathrm{diag}(\omega_1, \ldots, \omega_n),$$
$$\bar{\mathbf{C}}_{\{i:\}} = \omega_i(\mathbf{p}_{1,i} - \mathbf{p}_{2,i}).$$

in analogy to (7). With $\{\omega_1, \ldots, \omega_n\} \gg 1$ the two agents maintain a defined spatial distance $\mathbf{d}_i \approx \mathbf{p}_{1,i} - \mathbf{p}_{2,i}$ at time instance i. Fixed positional constraints according to (6) can be incorporated in a straightforward manner: As the trajectories of both agents are coupled through $\bar{\mathbf{C}}$, it is sufficient to specify positional constraints only for a single agent to deform both trajectories. When extending the approach to three agents the analogy of (25) is

$$\begin{bmatrix} \mathbf{L}_1 & 0 & 0 \\ 0 & \mathbf{L}_2 & 0 \\ 0 & 0 & \mathbf{L}_3 \\ \bar{\mathbf{P}}_- & \bar{\mathbf{P}}_+ & 0 \\ \bar{\mathbf{P}}_- & 0 & \bar{\mathbf{P}}_+ \\ 0 & \bar{\mathbf{P}}_- & \bar{\mathbf{P}}_+ \end{bmatrix} \begin{bmatrix} \mathbf{P}_{s1} \\ \mathbf{P}_{s2} \\ \mathbf{P}_{s3} \end{bmatrix} = \begin{bmatrix} \boldsymbol{\Delta}_1 \\ \boldsymbol{\Delta}_2 \\ \boldsymbol{\Delta}_3 \\ \bar{\mathbf{C}}_1 \\ \bar{\mathbf{C}}_2 \\ \bar{\mathbf{C}}_3 \end{bmatrix}.$$

Yet as computational complexity for a agents is $\mathscr{O}(a^2)$, the approach is limited to few agents only.

Fig. 6 shows a toy scenario in which two respectively three agents have to maintain a defined spatial distance (e.g. when holding an object) while circumnavigating two obstacles (black cylinders). It shows both undeformed trajectories without obstacles and deformed trajectories in the presence of obstacles. An additional graph on the right side displays the spatial distance d_{ij} for the undeformed trajectory and $d_{ij,m}$ for the deformed trajectory between agents j and j over time. It is visible that the distance between every two agents stays constant over time and changes only by an negligible amount due to the least squares solution (<4e-10 m) during deformation. One also sees how the trajectories of all agents adapt when just fixing the position of a single agent. Note that only the most primitive case with both a constant spatial distance and direction is displayed. Depending on the task it is necessary to vary the distance or orientation of the ensemble over time. This is done easily by modifying the elements in $\bar{\mathbf{C}}_1, \bar{\mathbf{C}}_2, \dots$

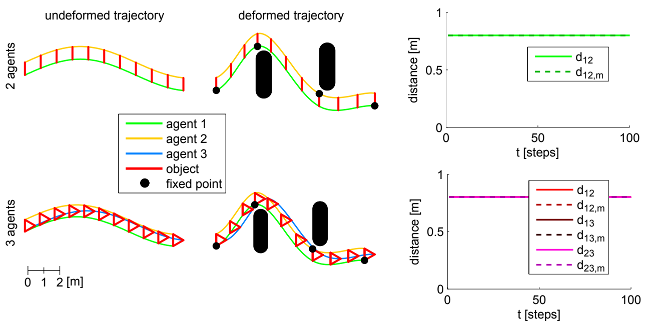


Fig. 6 Cooperative manipulation involving two and three agents in the presence of obstacles. Trajectory paths (left) and distance between every two agents (right)

3.5 Kinematic Constraints

All calculations so far only consider the trajectory adaption of a single point in a n-dimensional space. In most cases this single point refers to the position of the endeffector of a robotic manipulator in 3 dimensions. Here it is often required to fulfill additional constraints like joint limit avoidance, collision avoidance or maintaining a specific endeffector orientation. Such constraints can be incorporated through a prioritized inverse kinematics approach of the form

$$\dot{\boldsymbol{\theta}} = \mathbf{J}_1^+\dot{\mathbf{r}}_1 + (\mathbf{E}-\mathbf{J}_1^+\mathbf{J}_1)\mathbf{J}_2^+\dot{\mathbf{r}}_2, \tag{26}$$

see (Nakamura (1991)). In (26) the variables $\mathbf{J}_1$, $\mathbf{J}_2$ and $\dot{\mathbf{r}}_1$, $\dot{\mathbf{r}}_2$ refer to the task-specific Jacobians and task space velocities of primary and secondary task, $\mathbf{E}$ is the identity matrix and $\dot{\boldsymbol{\theta}}$ denote the generalized coordinates of the robot. Both self-collision avoidance and obstacle avoidance are achieved based upon enclosing cylinders covering all robot links and a repellent artificial potential field. In case the shortest distance d_{ca} between two links or link and obstacle falls below a defined threshold d_{ca}^{min}, the desired collision avoidance velocity becomes

$$\dot{\mathbf{r}}_1 = K_{ca}(d_{ca}^{min} - d_{ca}) \text{ if } d_{ca} < d_{ca}^{min},$$

with gain factor K_{ca}. Joint limits can be avoided by defining upper/lower bounds θ_i^{max}, θ_i^{min} that must not be exceeded for joint θ_i. In case they are exceeded, the desired joint limit velocity becomes

$$\dot{\mathbf{r}}_1 = \begin{cases} K_{jl}(\theta_i^{max} - \theta_i) \text{ if } \theta_i > \theta_i^{max}, \\ K_{jl}(\theta_i^{min} - \theta_i) \text{ if } \theta_i > \theta_i^{min}, \end{cases}$$

with gain factor K_{jl}, see Yamane and Nakamura (2003). If the desired endeffector position is encoded in the secondary, lower prioritized task, it's real position can differ from the desired position during task execution due to other higher prioritized tasks. In this case LTE allows to calculate an updated optimal trajectory online during task execution. This effect is illustrated Fig. 7. The left side shows a planar manipulator with 3 DOFs following a straight trajectory. On the right side an additional constraint is imposed, namely that the last joint (in red) must not collide with an added obstacle (in black). As the real endeffector trajectory deviates from the planned, straight trajectory, LTE replans new trajectories online (in orange).

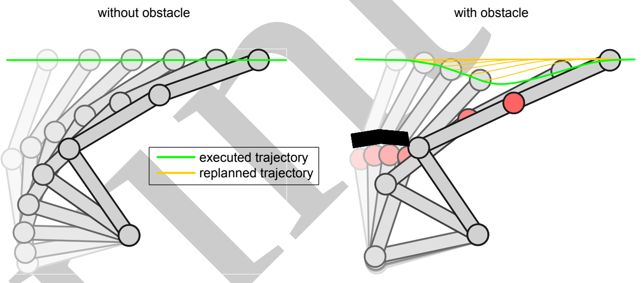


Fig. 7 Prioritized inverse kinematics with continuous trajectory replanning. Trajectory following without obstacle (left) and with obstacle (right)

4 Experimental Evaluation

This section evaluates the presented approaches on the one hand through simulations both with respect to computational complexity and in the spatial domain. As the LTE framework shows similarities with Elastic Strips, both approaches are compared. It concludes with a real-life experiment involving a HRP-4 robot executing a bimanual task while maintaining additional constraints. It must be mentioned that the choice of using sinusoidal paths in many examples is intentional as this type of path is well suited for evaluating the quality of each algorithm by inspection.

4.1 Path Similarity Measure

When evaluating LTE, it offers the advantage of implicitly providing intrinsic measures how similar two paths $\mathbf{P}$ and $\mathbf{P}_s$ are. As the preferred measure may vary from application to application, several are presented in this section to quantify the amount of deformation. A first measure is given by the least squares residual of (5) as

$$E = \|\mathbf{L}\mathbf{P}_s - \boldsymbol{\Delta}\|_F^2 + \sum_{i=1}^{p} \omega_i^2 \|\mathbf{p}_i - \mathbf{c}_i\|_2^2. \tag{27}$$

with $_F$ as the Frobenius norm. Implicitly it is assumed in (27) that only positional constraints apply. For positional constraints the offset $\|\mathbf{p}_i - \mathbf{c}_i\|_2^2$ between desired and resulting sampling point position is invisible to the human eye. Thus a more human-oriented measure (Nierhoff et al (2014)) focuses solely on the local trajectory properties while neglecting the error of the positional constraints as

$$E_1 = \|\mathbf{L}\mathbf{P}_s - \boldsymbol{\Delta}\|_F^2 .$$

Both residuals do not account for nonlinear deformation effects like rotations. Because LTE is expanded to consider them as well, a corresponding measure adapted from (Sorkine and Alexa (2007)) is presented in advance as

$$E_2 = \sum_{i=1}^{n} \sum_{j \in \mathcal{N}_i} w_{ij} \|(\mathbf{p}_j - \mathbf{p}_i) - \mathbf{R}_{is}(\mathbf{p}_{js} - \mathbf{p}_{is})\|_2^2 .$$

with $\mathbf{R}_{is}$ being the rotational matrix described Sec. 3.1 such that the resulting sampling point positions $\mathbf{p}_{js}, \mathbf{p}_{is} \in \mathbf{P}_s$ match the original sampling points $\mathbf{p}_j, \mathbf{p}_i \in \mathbf{P}$ best. A more general measure accounting not only for rotational effects but also for scaling is

$$E_3 = \sum_{i=1}^{n} \sum_{j \in \mathcal{N}_i} w_{ij} \|(\mathbf{p}_j - \mathbf{p}_i) - c\mathbf{R}_{is}(\mathbf{p}_{js} - \mathbf{p}_{is})\|_2^2 .$$

with a scalar scale factor c. One last and quite often used measure in literature is the summed quadratic difference of the Cartesian positions between original and deformed path, defined as

$$E_4 = \sum_{i=1}^{n} \|(\mathbf{p}_{is} - \mathbf{p}_i)\|_2^2 ,$$

for the original sampling point position $\mathbf{p}_i$ and the modified sampling point position $\mathbf{p}_{is}$. When presenting several expansions to LTE later in this article, the different path similarity measures will give the reader not only a qualitative but also quantitative impression about the quality of each expansion, making it easier to compare them. In addition they are well suited to show the amount of deformation over time.

4.1.1 Computational Complexity Comparison

Simulations compare the computational complexity of the different presented approaches in Sec. 2.1 and Sec. 3.2.
Shown in Fig. 8 is the processing time for a single deformation step over the number of path sampling points n. Four different approaches are evaluated, see Tab. 1. The table also shows whether it is a multiresolution approach and whether the approach can handle nonlinear deformation effects. Depending on the requirements, small paths are deformed in real time. As such it takes around 10ms to adapt a path with 300 sampling points using any of the two multiresolution approaches. Due to highly optimized routines for solving sparse equation systems, the original approach is by far

approach	described in	multires.	nonlin.
original	Sec. 2.1		
Laplacian rec.	Sec. 3.2.3	✓	✓
affine rec.	Sec. 3.2.4	✓	✓
ARAP	Sorkine and Alexa (2007)		✓

Table 1 Properties of different approaches

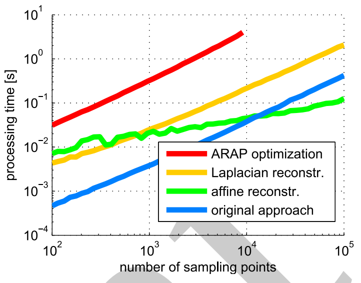


Fig. 8 Processing time comparison between a state-of-the-art approach (ARAP optimization) and the different methods presented in this article

the fastest one for $n < 10^4$, yet unable to cope with nonlinear deformation effects. For large trajectories with $n > 10^4$ the multiresolution approach with affine transformations for the reconstruction step is fastest, yet it is only applicable in three dimensions. All methods clearly outperform an existing state-of-the-art approach (ARAP) in terms of processing time. The original approach has a computational complexity of $\mathcal{O}(nm)$ due to a sparse linear equation system for every dimension. All other approaches have a computational complexity of $\mathcal{O}(nm^3)$ because they rely at some point on Arun's method requiring a SVD on a $m \times m$-matrix (13) and scale linearly with the number of sampling points n.

4.1.2 Spatial Comparison

This subsections shows comparisons between the different approaches of Sec. 2.1 and Sec. 3.2 in the spatial domain. For this purpose, a helix-shaped sinusoidal path is deformed by defining four positional constraints, see Fig. 9. The compared methods are: the original approach (Sec. 2.1), the ARAP optimization (Sorkine and Alexa (2007)) and multiple downsampling/
reconstruction
combinations described in Sec. 3.2. It is visible that the affine reconstruction method without proper downsampling (19) - in cyan - differs strongly from all other approaches. The bottom bar graphs show the normalized similarity measure values for all other methods. As the original approach minimizes E_1 and the ARAP optimization minimizes E_2, their values are smallest in the corresponding plots.

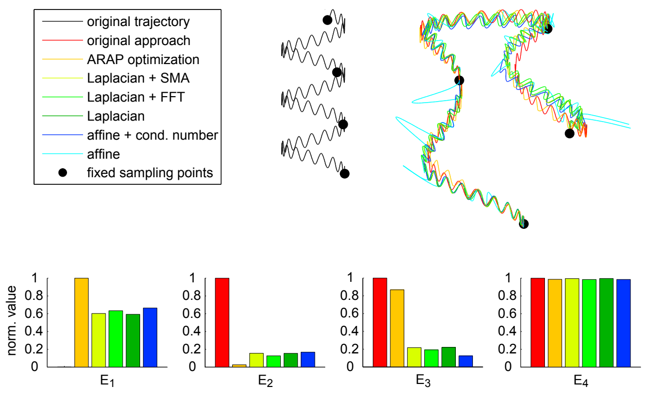


Fig. 9 Spatial comparison. Spatial extension of different methods (top) and corresponding similarity measures E_1-E_4 (bottom)

4.2 Comparison With Elastic Strips

The obstacle avoidance method in Sec. 3.3 shares common properties with the Elastic Strips framework (Brock and Khatib (2002)). Both methods rely on the decomposition into internal forces maintaining the original path shape and external forces deforming the path. Both methods use curvature-based methods to describe the internal forces. Yet both their definition and purpose differs. Whereas LTE uses the discrete Laplace-Beltrami operator

$$\mathbf{F}_i^{int,L} = \boldsymbol{\delta}_i = \frac{\mathbf{p}_{i-1} - 2\mathbf{p}_i + \mathbf{p}_{i+1}}{-2}, \quad i = 2,3,\dots,n-1,$$

to describe the internal force $\mathbf{F}_i^{int,L}$, Elastic Strips rely on a heuristic definition for the internal force $\mathbf{F}_i^{int,E}$ as

$$\mathbf{F}_i^{int,E} = k_c \left(\frac{d_{i-1}}{d_{i-1} + d_i} (\mathbf{p}_{i+1} - \mathbf{p}_{i-1}) - (\mathbf{p}_i - \mathbf{p}_{i-1}) \right), \quad i = 2,3,\dots,n-1, \tag{28}$$

with $d_i = \|\mathbf{p}_{i+1} - \mathbf{p}_i\|_2$. The external force $\mathbf{F}_i^{ext,E}$ is defined as

$$\mathbf{F}_i^{ext,E} = \begin{cases} k_r(d_0 - \|\mathbf{d_i}\|)\frac{\mathbf{d_i}}{\|\mathbf{d_i}\|} & \text{if } \|\mathbf{d_i}\| < d_0, \\ 0 & \text{otherwise.} \end{cases}$$

Whereas Elastic Strips try to maintain the shortest possible path in task space, LTE tries to maintain the original shape of the path. If the undeformed trajectory is a straight line, the result after deformation is roughly the same, see Fig. 10. Yet Elastic Strips cannot be applied to non-straight paths as they will always converge to a straight path in the absence of obstacles. Both methods tackle the problem of large deformations by modifying the internal forces. Whereas Elastic Strips use a modified minimum-distance formulation (28) that shares common properties with curvature based methods, LTE scales the internal forces as described in (23) and (24). Elastic Strips are advantageous from a computational point of view in two ways. First they only add sampling points (robot configurations) to the path when necessary, keeping the overall number low. LTE on the other hand always considers all sampling points. It also relies on a matrix inversion whereas Elastic Strips are calculated through a computationally more efficient gradient descent approach. The advantage of LTE is that it converges faster due to its least-squares approach.

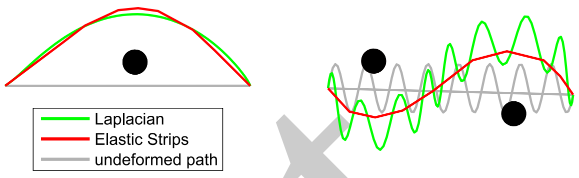


Fig. 10 Comparison between Elastic Strips and LTE in the presence of obstacles (black)

4.3 Robotic task

Real-life experiments consider a typical household task of disposing garbage in a bin by using LTE. The task comprises of lifting a bucket from a lower position onto a table, collecting garbage and disposing the garbage in the bin. To complete the task in a changed environment, safe circumnavigation of obstacles needs to be ensured, requiring the obstacle avoidance scheme in Sec. 3.3. For reliable bimanual manipulation tasks the cooperative scheme of Sec. 3.4 is adopted. The human demonstration movements are recorded at a frame rate of 200Hz using a Vortex motion capture system, tracking the position and orientation of both hands, bin and garbage. A HRP-4 robotic platform is used for task reprodution. A prioritized inverse kinematic approach as described in Sec. 3.5 ensures a physically consistent whole-body motion incorporating joint-angle limitations, self collision and COM-based balance while specifying the desired trajectories of both hands. Displayed in Fig. 11 are pictures of the key frames of the experiment. Each column corresponds to a different run: Human demonstration (left), robotic movement imitation (middle) and robotic movement adaption (right). Due to the different figure of robot and human, the objects of the imitation run are placed closer together. The multiresolution LTE approach of Sec. 3.2 in combination with positional constraints (6) accounting for the changed objects' positions is used to adapt the trajectory. Two modifications let the adaption run differ from the imitation run: During the first part of the adaption run the robot has to avoid an added obstacle (yellow book) when placing the bucket on the table. By creating a repellent artificial potential field (19) around the obstacle, positional constraints with low weights according to (20-21) maintain the shape of the original trajectory while lifting the bucket over the obstacle. This is illustrated in Fig. 11a. During the second part the initial garbage position is elevated by around 40cm, see Fig. 11b. The otherwise independent trajectories of left and right arm are coupled through the cooperative manipulation scheme in (25), maintaining a specific distance when holding the garbage bag and preventing it from falling down as indicated in Fig. 11c/d. As LTE only modifies the position of each endeffector, the orientation of both endeffectors is calculated independently. From the position of all hand markers during the demonstrated motion the human hand orientation is calculated and mapped to the robot. Velocity and acceleration of both endeffectors (EE) are shown in Fig. 12. Whereas there is a small velocity and acceleration of both endeffectors during the imitation run, the adaption run leads to high accelerations and velocities of the left endeffector when avoiding the obstacle. Both runs have a smooth velocity/acceleration profile.

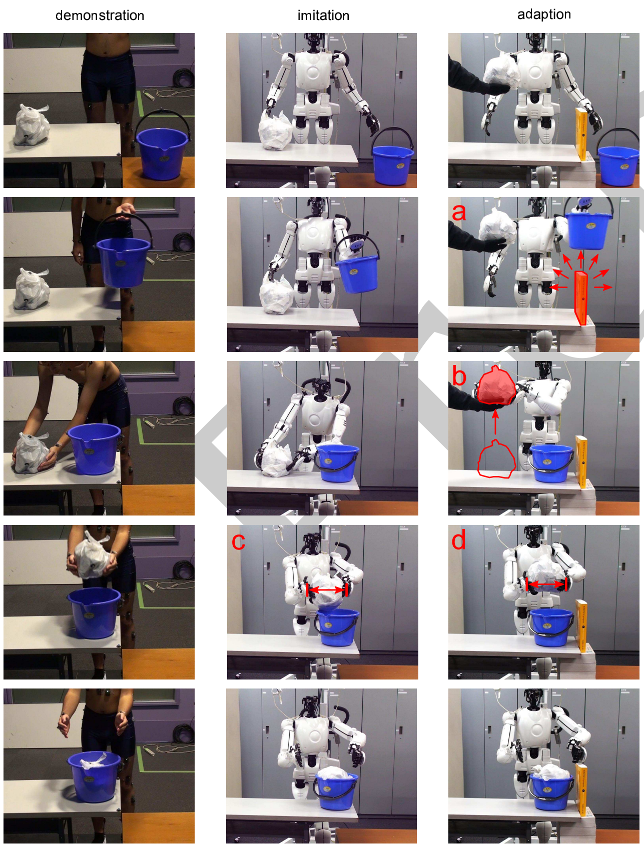


Fig. 11 Motion imitation task: Human demonstration (left), robotic movement reproduction (middle) and adaption (right).

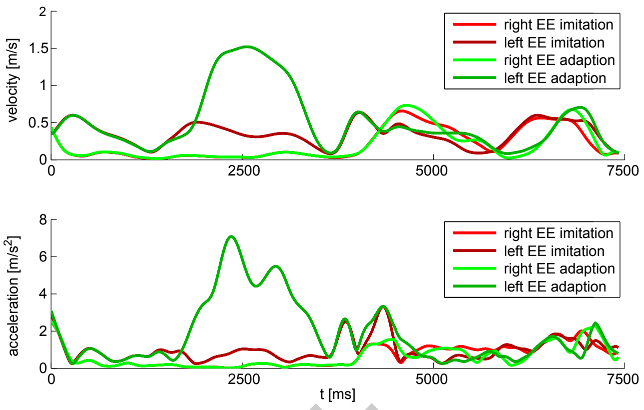


Fig. 12 Motion imitation task: Endeffector velocity (top) and acceleration (bottom) over time for the robotic movement reproduction and adaption

5 Discussion

Experiments showed how LTE can be adapted to suit common robotic problems for discretized trajectories. The multiresolution approach accounts for large deformations, overcomes the Laplacian-typical problems of being a linear operator and proves to be faster than an existing state-of-the-art approach. Positional constraints in combination with low weighting factors make it possible to deform a trajectory in a smooth manner while avoiding obstacles. Through suitable choice of parameters they can be fit to a user-specific tasks. Though a lot of extensions and improvements were presented, the original approach provided satisfying results when dealing with simple-shaped paths and small deformations. This is advantageous as the original approach is intuitive and extremely simple to implement ($<$ 15 LOC in Matlab).

Some issues of the LTE approach need special attention: The multiresolution approach depends on a proper parameterization of the number of support sampling points n' for a good tradeoff between capturing local and global trajectory properties. The same accounts - in weaker form - also for the parameters of the positional constraints for obstacle avoidance as otherwise undesired deformation effects occur. When being executed on the robot, one must be aware of all the problems associated with the prioritized inverse kinematics like a possible deviation from the desired trajectory requiring online replanning and workspace constraints of the hardware which can lead to singular configurations. As the inverse kinematics approach does not consider dynamic constraints like torque limits of the motors, they have to be considered separately.

6 Conclusion

The online adaptation of a-priori planned or learned motion trajectories is an important capability of autonomous robots moving in unstructured and dynamic environments. In this article we introduce Laplacian Trajectory Editing as a general framework for real-time retargeting of trajectories subject to constraints while preserving the local shape of the original trajectory. Due to its generality, the framework can be easily combined with other methods and task-specific extensions, of which some are described in this article. Positional constraints with low weighting factors make it possible to deform a trajectory in a reactive manner to avoid obstacles without explicitly specifying waypoints the trajectory has to pass. The combination with a prioritized inverse kinematics approach makes it possible to consider constraints in joint space while maintaining local trajectory properties in task space. The presented methods are evaluated in the spatial domain, with respect to processing time and through real-life experiments with the HRP-4 robot.

Acknowledgements This work was partially supported by the EU Horizon2020 project RAMCIP, under grant agreement no. 643433 and by the Ministry of Education, Science, Sports and Culture, Grant-in-Aid for Scientific Research (S), 2008-2012, 20220001, “Establishing Human-Machine Communication through Kinesiology and Linguistics Integration” (PI: Y. Nakamura)

References

Arun KS, Huang TS, Blostein SD (1987) Least-squares fitting of two 3-d point sets. IEEE Transactions on Pattern Analysis and Machine Intelligence 9(5):698–700

Billard A, Calinon S, Dillmann R, Schaal S (2008) Robot programming by demonstration. In: Springer Handbook of Robotics, Springer, pp 1371–1394

Botsch M, Kobbelt L (2004a) An intuitive framework for real-time freeform modeling. ACM Transactions on Graphics 23(3):630–634

Botsch M, Kobbelt L (2004b) A remeshing approach to multiresolution modeling. In: ACM Special Interest Group on Graphics and Interactive Techniques, pp 185–192

Brock O, Khatib O (2002) Elastic strips: A framework for motion generation in human environments. International Journal of Robotics Research 21(12):1031–1052

Desbrun M, Meyer M, Schröder P, Barr AH (1999) Implicit fairing of irregular meshes using diffusion and curvature flow. In: ACM Special Interest Group on Graphics and Interactive Techniques, pp 317–324

Dierkes U, Hildebrandt S, Sauvigny F (2010) Minimal surfaces. 1 ; 339, Springer

Do-Carmo MP (1976) Differential Geometry of Curves and Surfaces. Prentice Hall

Eck M, DeRose T, Duchamp T, Hoppe H, Lounsbery M, Stuetzle W (1995) Multiresolution analysis of arbitrary meshes. In: ACM Special Interest Group on Graphics and Interactive Techniques, pp 173–182

Flash T, Hogan N (1985) The coordination of arm movements: An experimentally confirmed mathematical model. Journal of Neuroscience 5(7):1688–1703

Hilario L, Montés N, Mora MC, Falcó A (2011) Real-time bézier trajectory deformation for potential fields planning methods. In: IEEE/RSJ International Conference on Intelligent Robots and Systems, pp 1567–1572

Hoffmann H, Pastor P, Park DH, Schaal S (2009) Biologically-inspired dynamical systems for movement generation: Automatic real-time goal adaptation and obstacle avoidance. In: IEEE International Conference on Robotics and Automation, pp 2587–2592

Karaman S, Frazzoli E (2011) Sampling-based algorithms for optimal motion planning. International Journal of Robotics Research 30(7):846–894

Karni Z, Gotsman C (2000) Spectral compression of mesh geometry. In: ACM Special Interest Group on Graphics and Interactive Techniques, pp 279–286

Kilner J, Hamilton AFdC, Blakemore SJ (2007) Interference effect of observed human movement on action is due to velocity profile of biological motion. Social Neuroscience 2(3-4):158–66

Kobbelt L, Bareuther T, Seidel HP (2000) Multiresolution shape deformations for meshes with dynamic vertex connectivity. Computer Graphics Forum 19(3)

Kupferberg A, Glasauer S, Huber M, Rickert M, Knoll A, Brandt T (2011) Biological movement increases acceptance of humanoid robots as human partners in motor interaction. AI & Society 26(4):339–345

Lavalle SM (1998) Rapidly-exploring random trees: A new tool for path planning. Tech. rep., No. 98-11

Lee D, Ott C (2011) Incremental kinesthetic teaching of motion primitives using the motion refinement tube. Autonomous Robots 31(2-3):115–131

Levine S, Koltun V (2012) Continuous inverse optimal control with locally optimal examples. In: International Conference on Machine Learning

Levy B (2006) Laplace-beltrami eigenfunctions towards an algorithm that ”understands” geometry. In: Shape Modeling International, p 13

Lipman Y, Sorkine O, Levin DCOD, Rössl C, Seidel HP (2004) Differential coordinates for interactive mesh editing. In: Shape Modeling International, pp 181–190

Lipman Y, Sorkine O, Alexa M, Cohen-Or D, Levin D, Rössl C, Seidel HP (2005) Laplacian framework for interactive mesh editing. International Journal of Shape Modeling 11(1):43–62

Luxburg U (2007) A tutorial on spectral clustering. Statistics and Computing 17(4):395–416

Meyer M, Desbrun M, Schröder P, Barr AH (2002) Discrete differential-geometry operators for triangulated 2-manifolds. Visualization and Mathematics pp 35–57

Mombaur K, Olivier AH, Crétual A (2013) Forward and inverse optimal control of bipedal running. In: Modeling, Simulation and Optimization, vol 18, Springer Berlin Heidelberg, pp 165–179

Nakamura Y (1991) Advanced robotics - redundancy and optimization

Nierhoff T (2013a) URL `http://www.itr.ei.tum.de/fileadmin/w00bok/www/CodeExamples/laplacianHardConstraints.m` Accessed 24 June 2015.

Nierhoff T (2013b) URL `http://www.itr.ei.tum.de/fileadmin/w00bok/www/CodeExamples/laplacianSoftConstraints.m` Accessed 24 June 2015.

Nierhoff T, Hirche S (2012) Fast trajectory replanning using laplacian mesh optimization. In: IEEE International Conference on Control, Automation, Robotics and Vision

Nierhoff T, Hirche S, Nakamura Y (2013) Multiresolution laplacian trajectory replanning. In: Proceedings of the Annual Conference of RSJ

Nierhoff T, Hirche S, Nakamura Y (2014) Sampling-based trajectory imitation in constrained environments using laplacian-rrt*. In: IEEE/RSJ International Conference on Intelligent Robots and Systems

Nierhoff T, Hirche S, Nakamura Y (2013) Variable Positional Constraints for Laplacian Trajectory Editing. In: DGR-Tage

Pastor P, Hoffmann H, Asfour T, Schaal S (2009) Learning and generalization of motor skills by learning from demonstration. In: IEEE International Conference on Robotics and Automation, pp 1293–1298

Pham QC (2011) Fast trajectory correction for nonholonomic mobile robots using affine transformations. In: Robotics: Science and Systems

Pham QC, Nakamura Y (2013) A new trajectory deformation algorithm based on affine transformations. In: International Joint Conference on Artificial Intelligence

Quinlan S, Khatib O (1993) Elastic bands: Connecting path planning and control. In: IEEE International Conference on Robotics and Automation, pp 802–807

Reuter M, Biasotti S, Giorgi D, Patanè G, Spagnuolo M (2009) Discrete laplace-beltrami operators for shape analysis and segmentation. Computational Geometry 33(3):381–390

Schaal S (2006) Dynamic movement primitives - a framework for motor control in humans and humanoid robotics. Adaptive Motion of Animals and Machines pp 261–280

Schulman J, Duan Y, Ho J, Lee A, Awwal I, Bradlow H, Pan J, Patil S, Goldberg K, Abbeel P (2014) Motion planning with sequential convex optimization and convex collision checking. International Journal of Robotics Research 33(9):1251–1270

Shoemake K (1985) Animating rotation with quaternion curves. In: ACM International Conference on Computer Graphics and Interactive Techniques, vol 19, pp 245–254

Sorkine O, Alexa M (2007) As-rigid-as-possible surface modeling. In: Eurographics Symposium on Geometry Processing, pp 109–116

Sorkine O, Cohen-Or D (2004) Least-squares meshes. In: Shape Modeling International, pp 191–199

Strichartz RS (1983) Analysis of the laplacian on the complete riemannian manifold. Journal of Functional Analysis 52(1):48–79

Taubin G (1995) A signal processing approach to fair surface design. In: ACM Special Interest Group on Graphics and Interactive Techniques, pp 351–358

Umeyama S (1991) Least-squares estimation of transformation parameters between two point patterns. IEEE Transactions on Pattern Analysis and Machine Intelligence 13(4):376–380

Wardetzky M, Mathur S, Kälberer F, Grinspun E (2007) Discrete laplace operators: no free lunch. In: Eurographics Symposium on Geometry Processing, pp 33–37

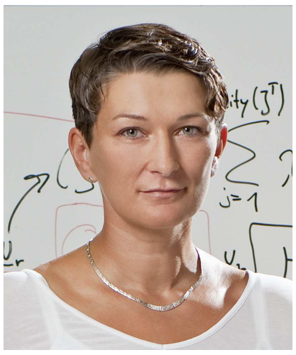
Yamane K, Nakamura Y (2003) Natural motion animation through constraining and deconstraining at will. IEEE Transactions on Visualization and Computer Graphics 9:352–360

Zhang H, van Kaick O, Dyer R (2010) Spectral mesh processing. Computer Graphics Forum 29(6):1865–1894

Zhou K, Huang J, Snyder J, Liu X, Bao H, Guo B, Shum HY (2005) Large mesh deformation using the volumetric graph laplacian. ACM Transactions on Graphics 24(3):496–503

Zucker M, Ratliff N, Dragan A, Pivtoraiko M, Klingensmith M, Dellin C, Bagnell JAD, Srinivasa S (2013) Chomp: Covariant hamiltonian optimization for motion planning. International Journal of Robotics Research 32:1164–1193

Thomas Nierhoff is currently pursuing his doctorate at the Institute of Information-oriented Control, Technische Universitaet Muenchen, Munich Germany. He received the B.S. and Diploma Engineer degree in electrical engineering from the same university in 2009 and 2010. From 2012 to 2014 he has been visiting researcher at Nakamura Labs at the University of Tokyo. His reserch interests are in the field of robotics, motion planning and machine learning.



Sandra Hirche (Senior Member, IEEE) received the Dimploma Engineer degree in mechanical engineering and transport systems from the Technical University Berlin, Berlin, Germany, in 2002 and the Doctor of Engineering degree in electrical engineering and computer science from the Technische Universitaet Muenchen, Munich, Germany, in 2005. From 2005 to 2007, she was a Japanese Society for the Promotion of Science (JSPS) Postdoctoral Researcher at the Tokyo Institute of Technology, Tokyo, Japan. Since 2008, she has been an Associate Professor heading the Associate Institute for Information-oriented Control in the Department of Electrical Engineering and Information Technology, Technische Universitaet Muenchen. Her research interests include control over communication networks, networked control systems, control of large-scale systems, cooperative control, human-in-the-loop control, humanmachine interaction, robotics and haptics. Dr. Hirche has been Chair for Student Activities in the IEEE Control System Society (CSS) since 2009, Chair of the CSS Awards Subcommittee on 'BCDC Best Student-Paper Award' since 2010, and an elected member of the Board of Governors of IEEE CSS since 2010.

Yoshihiko Nakamura received the B.S., M.S., and Ph.D. degrees in Precision Engineering from Kyoto University, Japan in 1977, 1978, and 1985. He was Assistant Professor at the Automation Research Laboratory, Kyoto University, from 1982 to 1987. He joined the Department of Mechanical and Environmental Engineering, University of California, Santa Barbara, in 1987 as an Assistant Professor, and became an Associate Professor in 1990. He was also a co-director of the Center for Robotic Systems and Manufacturing at UCSB. Since 1991, he has been with University of Tokyo, Japan; and is currently Professor at Department of Mechano-Informatics. Humanoid robotics, cognitive robotics, neuro musculoskeletal human modeling, biomedical systems, and their computational algorithms are his current fields of research. He is Fellow of Japan Society of Mechanical Engineers, Fellow of Robotics Society of Japan, Fellow of IEEE, and Fellow of World Academy of Arts and Science. Currently, Dr. Nakamura (2012-2015) serves as President of International Federation for the Promotion of Mechanism and Machine Science (IFToMM). Dr. Nakamura is Foreign Member of Academy of Engineering Science of Serbia and Montenegro, and TUM Distinguished Affiliated Professor of Technische Universitaet Muenchen.