Integrated Motion and Clasp Planning with Virtual Linking

Johannes A. Stork Florian T. Pokorny Danica Kragic

***Abstract*— In this work, we address the problem of simultaneous clasp and motion planning on unknown objects with holes. Clasping an object enables a rich set of activities such as dragging, toting, pulling and hauling which can be applied to both soft and rigid objects. To this end, we define a virtual linking measure which characterizes the spacial relation between the robot hand and object. The measure utilizes a set of closed curves arising from an approximately shortest basis of the object's first homology group. We define task spaces to perform collision-free motion planing with respect to multiple prioritized objectives using a sampling-based planing method. The approach is tested in simulation using different robot hands and various real-world objects.**

I. Introduction

Robots need to be able to interact with both known *and* previously unseen objects. This problem has been studied extensively which has resulted in state of the art methods for grasp synthesis and planning. In addition to simply fixing an object in the end effector, we would however also like to equip robots with a richer repertoire of grasps affording both in-hand manipulation and additional activities such as dragging, toting, pulling and hauling of soft and rigid objects. These activities may require more force and grip than a point contact-based precision grasp provides and a grasps involving the complete robot hand may be required instead.

To address this, we present work on simultaneous grasp and motion planning, where we concentrate in particular on the *clasping* of unknown objects. A *clasp* is a type of enveloping grasp that affords pushing, dragging and alike. In our work, it is formalized using the Gauss linking integral for closed curves. Clasp planning is performed simultaneously with motion planning using a Task Space Rapidly-exploring Random Tree (TS-RRT) approach. Our approach uses task space control, sampling-based search and a topological representation of objects. Furthermore, we rely solely on a rough object reconstruction from point clouds and a representation that builds on a definition of virtual linking.

The main contribution of our work is to address the problem of clasp and motion planning in an integrated manner. In addition, the approach can be used on previously unseen objects without the need for an exact mesh model. Our approach defines and utilizes multiple task objectives to control the motion of the robot. These objectives have different priorities and we show how to design a controller taking different and altering priorities into consideration.

This work was supported by the Swedish Foundation for Strategic Research and the EU grants FLEXBOT (FP7-ERC-279933) and TOPOSYS (FP7-ICT-318493). The authors are with the Computer Vision and Active Perception Lab, Centre for Autonomous Systems, CSC, KTH Royal Institute of Technology, Stockholm, Sweden, {jastork,fpokorny,dani}@kth.se.

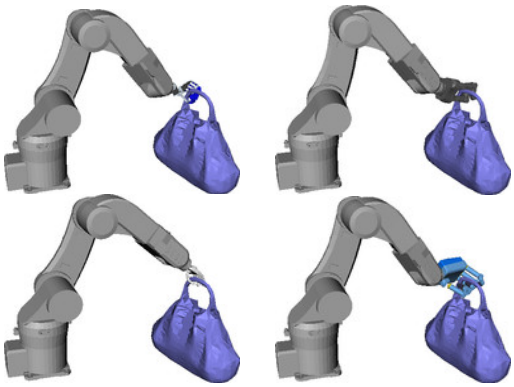

Fig. 1. A few examples of our approach for clasp planning with various robotic hands. Here, the topological feature describes the purse's handle.

II. Related Work and Contributions

Rapidly-exploring Random Trees (RRTs) are commonly used for motion planning [1]–[3], manipulation planning [4]–[6] and the planning of grasping motions [7]–[9]. For complex robots, planning of collision-free trajectories is a PSPACE-hard problem in general [10]. The constraints originate mainly from configuration space ($\mathcal{C}$-space) complexity and from representing the collision-free region, $\mathcal{C}_{free}$. To avoid a time consuming explicit representation of $\mathcal{C}_{free}$, [11]–[13] and others have proposed sampling-based algorithms which implicitly cover $\mathcal{C}_{free}$, namely RRTs [13].

In [4], a novel kind of goal bias is introduced by adopting workspace heuristic functions implicitly defining the $\mathcal{C}$-space goal region. A workspace distance is used for selection of expansion nodes which are then randomly extended. Manipulation involving inverse kinematics and complex objects is addressed in [5]. There, the search is directed towards the goal in a subspace of the manipulator's $\mathcal{C}$-space. Workspace bias is realized using transpose Jacobian control to produce locally optimal expansions. Similarly to our approach, they iteratively apply control until a collision occurs in order to reach a goal. We also consider goal bias in a space different from the original $\mathcal{C}$-space; however, we do not work with a subspace of the $\mathcal{C}$-space.

RRT-based planning of grasping motions commonly relies on a set of pre-defined grasp poses that are computed offline [7]–[9]. Thus, these approaches consider the problem in two separate phases and could analogously be transferred to clasping. However, we are interested in an *integrated* approach for clasp and motion planning similar to the works of [14], [15]. For grasping, such an integrated approach is presented in [16] for single and dual arm problems. During random exploration of the robot arm's $\mathcal{C}$-space, the Grasp-RRT algorithm computes approach movements

towards heuristically selected points on the object followed by grasp scoring. The idea is similar to ours, although the Grasp-RRT plans contact-level grasps. The main difference is that Grasp-RRT considers objects of known geometry and uses grasp quality evaluation while our approach uses a rough object reconstruction from point clouds and virtual linking. Both approaches integrate motion planning and do not rely on pre-calculated hand poses. The Grasp-RRT searches only in the robot arm's $\mathcal{C}$-space while our approach searches in a task space and considers both the arm and the hand simultaneously.

While the approaches of [4], [5] consider subspaces of the $\mathcal{C}$-space to realize goal bias, the concept of task spaces from feedback control design has been proposed to reduce planning space dimensionality [17]. The Task Space RRT (TS-RRT) samples and builds the tree only in the task space and entirely avoids non-trivial constraints in $\mathcal{C}$-space. The use of a TS-RRT is demonstrated by planning for the end effector position of a 1500 DOF robot arm in [17] and by planning legged locomotion over rough terrain in [18]. The hybrid motion planning approach of [19] uses TS-RRT and adapts the $\mathcal{C}$-space trajectory to avoid obstacles. In [20], the task space path is modified and secondary motion objectives in the form of cost functions are employed. For our approach, we add another extension procedure to the TS-RRT and also use secondary objectives. However, our secondary objectives lie in the task space and we additionally use altering task priorities.

For many object interaction tasks, objects do not need to be completely immobilized in the hand. Instead, caging grasps may be enough. Thus, our work focuses on the generation of *clasping grasps* that fall into the category of caging grasps [21]–[24] to *form a chain* consisting of the object and the robot hand. The notion of stretching and squeezing cages was introduced for the analysis of caging mobile rigid bodies by [25] and was later extended to more than two fingers [26]. Caging configurations can be considered as a waypoint towards immobilizing grasps [26] and as a method to deal with uncertainty [27].

Applications in manipulation planning related to caging grasps are proposed by [28]. They consider motion planning for tasks such as opening and closing doors, drawers, etc. Exploring an initial set of caging grasps using a RRT, they ensure caging at each expansion by a random motion escaping test. In our approach, we plan caging configurations along with robot motions that could be used by a system such as the one mentioned above. To show that the resulting clasps are secure, we also perform random movements trying to separate the robot from the object in a rigid body simulation.

In our recent work [29], we use winding numbers to describe how much a robot hand is wrapped around a given point. However, this representation is merely 2-dimensional and requires a projection onto a plane. This projection results in a loss of the 3-dimensional relationship. In this work, we will instead use the notion of the Gauss linking integral described in Sec. III-A which can be used to characterize the linking of curves in three dimensions.

III. Preliminaries

We now formalize the concept of clasping using two closed non-intersecting curves and explain how it is integrated with motion planning.

A. Gauss Linking Integrals

Given two closed non-intersecting curves γ_1, γ_2 in $\mathbb{R}^3$, their linking number $Lk(\gamma_1, \gamma_2) \in \mathbb{Z}$ provides an invariant which describes a spacial relation between the two curves. If it is non-zero, the two curves cannot be separated without breaking the loops. If $\gamma_1, \gamma_2 \colon [0,1] \to \mathbb{R}^3$ are smooth closed curves, Lk can be computed using the Gauss linking integral:

$$Lk(\gamma_1, \gamma_2) = \frac{1}{4\pi} \int_0^1 \int_0^1 \left\langle \frac{\gamma_1(s) - \gamma_2(t)}{\|\gamma_1(s) - \gamma_2(t)\|^3}, \gamma_1'(s) \times \gamma_2'(t) \right\rangle \mathrm{d}s\, \mathrm{d}t. \quad (1)$$

The above integral also leads to a formula for the case of piecewise linear curves [30] and has also been used when γ_1, γ_2 are *not closed* [31], [32]. However, in the latter case the resulting real valued quantity is then not a topological (isotopy) invariant anymore.

B. Shortest Loops and Homology

Informally, the first homology group $H_1(U)$ [33] of a closed surface $U \subset \mathbb{R}^3$ describes equivalence classes of closed curves (1-cycles) up to curves that form the boundary of parts (2-cycles) of U. $H_1(U)$ yields an Abelian version of the well-known first fundamental group $\pi_1(U)$ which describes equivalence classes of closed curves up to continuous deformations. Importantly, $H_1(U)$ is a topological quantity which stays invariant under certain continuous deformations (homotopies) of U itself. When computed over a finite field, $H_1(U)$ is a vector space. In our work, we will not work directly with a surface U, but with a simplicial complex approximation $\mathcal{K}_d$ of U for which $H_1(\mathcal{K}_d)$ is also well-defined. We are interested in a basis for $H_1(\mathcal{K}_d)$ corresponding to curves which are approximately of *shortest length* and use the software *ShortLoop* [34], [35] to approximately determine such curves given $\mathcal{K}_d$.

C. Task Space Planner Outline

To avoid high-dimensionality in motion planning, we facilitate a low-dimensional approach commonly used in feedback control design that has been brought to RRT planning as Task Space RRT (TS-RRT) [17]. A sampling-based search in task spaces is often appropriate since, for systems with many degrees of freedom, no closed-form solution exists to map task spaces to $\mathcal{C}$-space. Most importantly, TS-RRT operates on the task space and combines a projection from $\mathcal{C}$-space to the task space with Moore–Penrose pseudoinverse Jacobian local control. Usual RRTs instead search the $\mathcal{C}$-space directly, possibly using different types of goal bias [4], [5].

We search for a collision-free trajectory and secure clasping grasps simultaneously using a TS-RRT framework with multiple tasks objectives of altering priorities. Two tasks form a joint task space searched by the TS-RRT, while a

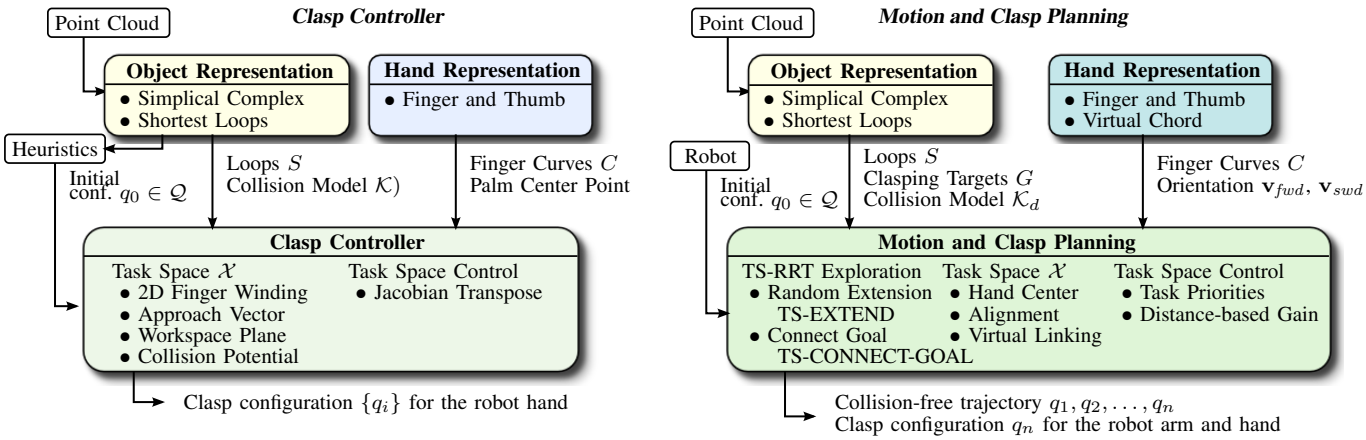

Fig. 2. Comparison of the clasp control approach of [29] *(left)* and the motion and clasp planning proposed in this work *(right)*. Both approaches rely on the same object representation. While the clasp controller considers winding and uses an approach vector heuristic to generate a clasping pose of a disembodied robot hand, the system presented here is based on virtual linking and results in a complete motion plan for a robot arm and hand.

third task realizes a permanent robot pose bias. No explicit $\mathcal{C}$-space target is defined—instead, we allow a pre-defined goal tolerance in the task space to implicitly define goal regions, which are mainly defined in terms of the concept of virtual linking.

A conceptual outline of our planning approach is depicted in Fig. 2 alongside the clasp controller used in our previous work [29]. Common features are the use of shortest loops and finger curve representations. Instead of the 2-dimensional winding number representation used in our earlier work, virtual linking is used to control hand joints towards a clasping configuration. Our current work does not require collision potentials since collision checking is intrinsic to RRT approaches. The clasp controller returns a set of secure clasp poses while the motion and clasp planner results in a collision-free trajectory of a robot arm and hand together with a final clasping pose.

IV. Methodology

We now describe details of our Task Space RRT approach and the representations of object and robot hand.

A. Object Representation

Given a point cloud of an object, we obtain a Delaunay triangulation, $\mathcal{D}$, and proceeded to refine it by eliminating triangles containing edges of length d or larger. This leads to a simplicial complex, $\mathcal{K}_d$, approximating the object. A basis for $H_1(\mathcal{K}_d)$ consisting of approximately shortest loops can then be computed with a polynomial time algorithm [35] using the *ShortLoop* software [34]. Details of the process and its parameters can be found in our previous work [29]. We denote the resulting set of piecewise linear loops by $S = \{\gamma_1, \gamma_2, \ldots, \gamma_m\}$. Each $\gamma \in S$ represents a basic topological and geometric attribute corresponding to a hole in the object. Note that $\mathcal{K}_d$ is a rough approximation of the object which is possibly perceived as a noisy point cloud. Therefore $\mathcal{K}_d$ may contain interior triangles such as in the thin handles of the objects shown in Fig. 3.

A clasping target on a loop $\gamma \in S$ consists of a point $\mathbf{p} \in \gamma$ and its approximated tangent $\mathbf{p}'$. The tangent $\mathbf{p}'$ can easily be determined from two points on γ close to $\mathbf{p}$.

Fig. 3. *Left to right:* real size objects used for experiments and the number of detected loops: *Purse* (1), *Bag* (1), *Travel Bag* (1), *Lawn Mower* (2), *Chair* (7). *Bottom row:* collision model $\mathcal{K}_d$ and selected loops used as object representation in the experiments.

The complete object representation, $\big(S, \mathcal{K}_d, \{G_\gamma \mid \gamma \in S\}\big)$, consists of a set of approximated shortest loops, S, of the simplicial complex, $\mathcal{K}_d$, which is used as a collision model and of sets of clasping targets $G_\gamma = \big\{ \big(\mathbf{p}_i, \mathbf{p}_i'\big) \mid i = 1, 2, \ldots, k_\gamma \big\}$ where the $\mathbf{p}_i$ are sampled on γ. Examples for pairs of $\gamma \in S$ and $\mathcal{K}_d$ can be inspected in Fig. 3. It can be seen that a path γ does not necessarily lie exactly on the original surface since the refined triangulation is based on point-samples and provides only a rough description of the real object. Additionally, γ may run along the interior edges of $\mathcal{K}_d$.

B. End Effector Representation

For a given robot hand with l fingers, we select non-closed piecewise linear curves, $C = \{\alpha_1, \alpha_2, \ldots \alpha_{l-1}\}$, running from the fingertips through the joints to the tip of a respective opposable "thumb". For our algorithm to apply, it is necessary to identify at least one pair of opposable fingers. For most current robot hands such α_i are easily identified. Fig. 4 outlines two curves for the Schunk SDH hand and four finger curves for each the iCub, DLR 1 and Armar III hand (red). A hand's orientation is described by orthogonal unit vectors $\mathbf{v}_{fwd}$, $\mathbf{v}_{swd}$ pointing forward (blue) and sidewards (green), respectively. The vectors are rigidly attached to the hand's tool center point and allow to describe angles between the hand pose and any given frame. Fig. 4 depicts the initial hand pre-shapes used in our planning approach.

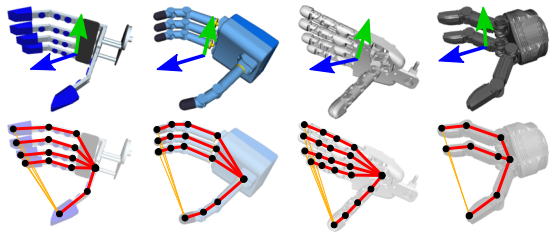

Fig. 4. *From left to right:* the *Armar III*, *DLR 1*, *iCub* and *Schunk SDH* hands in clasping pre-shape. Orientation vectors are depicted in *blue* and *green*. Finger curves (red) run from the fingertips to the tip of the "thumb". A virtual chord (orange) completes each finger curve to a virtual loop.

C. Virtual Linking

For sampling-based search, the description of the goal states—in this case caging grasps—is essential. Utilizing the closed loops $\gamma \in S$ that describe potentially claspable parts of an object and the finger curves $\alpha \in C$ that model the fingers' shape, it is plausible to consider the Gauss linking integral $Lk\left(\gamma, \alpha\right)$ to estimate the linking of a loop γ and a pair of fingers even if α is not closed. The larger the value of $\left|Lk\left(\gamma, \alpha\right)\right|$, the more linked one could consider the fingers and the loop. However, the Gauss linking integral, when applied to a non-closed curve α neglects the fingertip distances and two curves with absolute linking close to 1 can still be untangled in certain cases, even if one of the two curves is a closed loop. Additionally the value of $\left|Lk\left(\gamma, \alpha\right)\right|$ does not necessarily increase monotonically when the open curve α is increasingly closed which constitutes an unpleasant property from the control point of view.

For this reason, we propose a novel linking measure compliant with the representations stated above and based on the Gauss linking integral. Connecting the first and the last node of a non-closed finger curve $\alpha \in C$ with a virtual cord results in a virtual finger loop $\hat{\alpha}$. Fig. 4 exemplifies the idea of virtual cords (orange) for different robot hands. If a virtual finger loop $\hat{\alpha}$ and a loop $\gamma \in S$ have $\left|Lk\left(\gamma, \hat{\alpha}\right)\right| \geq 1$, the loop γ "runs through" the hand. Closing the hand at this state could result in a caging configuration but the closing process of the hand might be obstructed (see Fig. 8). However, reducing the length of the virtual cord to a minimum by closing the opposing fingers will cage the closed loop γ. The following definitions specify this notion mathematically.

Let $|\hat{\alpha}|$ denote the length of a virtual finger loop $\hat{\alpha}$ and δ_α the Euclidean distance of the first and last node of the finger curve α, measuring the length of the virtual cord connecting the finger tip and the tip of the "thumb". We define the *virtual linking* of the closed curve γ and non-closed curve α by

$$VLk\left(\gamma, \alpha\right) = \left(1 - \frac{\delta_\alpha}{|\hat{\alpha}|}\right) Lk\left(\gamma, \hat{\alpha}\right) \quad . \tag{2}$$

The virtual linking of α and γ assumes the value of 0 as long as the loop does not "run through" the fingers. Otherwise it describes the "closing" of the finger curve taking the value $Lk(\gamma, \alpha)$ when the tips of the finger and the "thumb" meet. If the virtual linking value is continuously increased until the fingertips meet, or until the fingers are obstructed, the hand is closed around the object. This makes the virtual linking a suitable projection for task space control.

D. Exploration

Alg. 1 describes the basic planning procedure of our TS-RRT framework. Starting at a $\mathcal{C}$-space pose $\mathbf{q}_{init} \in \mathcal{Q}$, a tree $\mathcal{T}$ in the task space $\mathcal{X}$ is build. To maintain the correspondence between a $\mathcal{C}$-space configuration $\mathbf{q}$ and its task space mapping $\mathbf{x}$, they are always jointly appended to the tree. In each iteration, either an undirected random exploration towards $\mathbf{x}_{rand} \in \mathcal{X}$ or goal directed iterative control is performed. By using task space control and sampling-based $\mathcal{C}$-space collision checking, the tree is ensured to contain only reachable and collision-free configurations. The search terminates in the procedure TS-CONNECT-GOAL when a task space goal has been reached up to a pre-defined tolerance. Thereafter, fingers are closed until contact occurs and the solution sequence is post-processed in configuration space using a randomized path pruning technique also used by [16] to produce a smooth $\mathcal{C}$-space trajectory.

Algorithm 1 BUILD-TS-RRT$\left(\mathbf{q}_{init}, \gamma, G_\gamma\right)$

Require: $\mathbf{q}_{init} \in \mathcal{Q}$, γ loop, G_γ positions on loop with tangents
1: $\mathbf{x}_{init} \leftarrow$ TS-PROJECTION$(\mathbf{q}_{init})$
2: $\mathcal{T}$.init$(\mathbf{q}_{init}, \mathbf{x}_{init})$
3: **for** $k = 1$ to K **do**
4: **if** with some probability P **then**
5: $\mathbf{x}_{rand} \leftarrow$ RANDOM-TS-SAMPLE() $\in \mathcal{X}$
6: TS-EXTEND$(\mathcal{T}, \mathbf{x}_{rand}, \gamma, G_\gamma)$
7: **else**
8: **for all** $\left(\mathbf{p}, \mathbf{p}'\right) \in G_\gamma$ **do**
9: $\mathbf{x}_{goal} \leftarrow \left(\mathbf{p}_1, \mathbf{p}_2, \mathbf{p}_3, 1.0\right)^{\mathrm{T}}$
10: TS-CONNECT-GOAL$(\mathcal{T}, \mathbf{x}_{goal}, \gamma, G_\gamma)$
11: **end for**
12: **end if**
13: **end for**

Two different procedures are applied to extend the task space tree $\mathcal{T}$. TS-EXTEND [17] in Alg. 2 selects the L_2-nearest neighbor $\mathbf{x}_{near}$ in task space as a starting point. Instead of connecting $\mathcal{C}$-space positions, local task space control at $\mathbf{q}_{near}$ is used to generate a new $\mathcal{C}$-space position $\mathbf{q}_{new}$. If the path from $\mathbf{q}_{near}$ to $\mathbf{q}_{new}$ is collision-free, the tree is extended. The goal bias is realized by the procedure TS-CONNECT-GOAL in Alg. 3, which is proposed here as the TS-RRT equivalent of a similar function in JT-RRT [5]. Stepwise control is iterated until the task space goal $\mathbf{x}$ is reached up to a pre-defined tolerance. TS-CONNECT-GOAL ends when a collision appears or when the task space distance to the goal is small enough. In the latter case, the last node, $\mathbf{q}_{new}$, and its ancestors define the solution sequence. A node in $\mathcal{T}$ is only selected once for extension towards a certain goal which has been omitted in Alg. 3 for clarity.

E. Control

The process of clasping an object loop using both one's arm and hand conceptually consists of the actions *(a)* moving the hand towards some segment of the loop, *(b)* turning the opened hand towards the loop segment and *(c)* closing the hand while maintaining the loop segment centered between the actuated fingers. While the first two actions need to be

Algorithm 2 TS-EXTEND$(\mathcal{T}, \mathbf{x}, \gamma, G)$

Require: $\mathbf{x} \in \mathcal{X}$, γ loop, G positions on loop with tangents
1: $(\mathbf{q}_{near}, \mathbf{x}_{near}) \leftarrow$ TS-NEAREST-NEIGHBOR$(\mathcal{T}, \mathbf{x})$
2: $\mathbf{q}_{new} \leftarrow$ TS-CONTROL$(\mathbf{q}_{near}, \mathbf{x}_{near}, \mathbf{x}, \gamma, G)$
3: $\mathbf{x}_{new} \leftarrow$ TS-PROJECTION$(\mathbf{q}_{new})$
4: **if** COLLISION-FREE$(\mathbf{q}_{near}, \mathbf{q}_{new})$ **then**
5: $\quad\mathcal{T}$.append$(\mathbf{q}_{near}, \mathbf{q}_{new}, \mathbf{x}_{new})$
6: **end if**

Algorithm 3 TS-CONNECT-GOAL$(\mathcal{T}, \mathbf{x}, \gamma, G)$

Require: $\mathbf{x} \in \mathcal{X}$, γ loop, G positions on loop with tangent
1: $(\mathbf{q}_{near}, \mathbf{x}_{near}) \leftarrow$ TS-NEAREST-NEIGHBOR$(\mathcal{T}, \mathbf{x})$
2: **loop**
3: $\quad\mathbf{q}_{new} \leftarrow$ TS-CONTROL$(\mathbf{q}_{near}, \mathbf{x}_{near}, \mathbf{x}, \gamma, G)$
4: $\quad\mathbf{x}_{new} \leftarrow$ TS-PROJECTION$(\mathbf{q}_{new})$
5: $\quad$**if** COLLISION-FREE$(\mathbf{q}_{near}, \mathbf{q}_{new})$ **then**
6: $\quad\quad\mathcal{T}$.append$(\mathbf{q}_{near}, \mathbf{q}_{new}, \mathbf{x}_{new})$
7: $\quad\quad$**if** TS-TOLERANCE$(\mathbf{x}, \mathbf{x}_{new})$ **then**
8: $\quad\quad\quad$TERMINATE-SEARCH$(\mathbf{q}_{new})$
9: $\quad\quad$**end if**
10: $\quad\quad(\mathbf{q}_{near}, \mathbf{x}_{near}) \leftarrow (\mathbf{q}_{new}, \mathbf{x}_{new})$
11: $\quad$**else**
12: $\quad\quad$**return**
13: $\quad$**end if**
14: **end loop**

coordinated to position the loop segment between the fingers, the last action has to be performed only when it is already within reach of the fingers. Still, the hand's position needs to be adjusted while the fingers close to clasp. This is most important for asymmetric hands where the center point of the fingers shifts notably when closing. Using a TS-RRT, we can describe each of these three actions by its own task space and mediate them using priorities and gain functions.

To capture the change of hand shape during different stages of the closing, action *(a)* is described by the center point of the axis-aligned bounding box (AABB) of all finger curves. The task space projection $\Pi_{pos} \colon \mathcal{Q} \to \mathcal{X}_{pos} = \mathbb{R}^3$ is defined by $\Pi_{pos}(\mathbf{q}) = \mathbf{p}_{AABB}$. Action *(b)* is described by a task space, $\mathcal{X}_{align}$, that considers alignment of the unit vectors $\mathbf{v}_{fwd}$ and $\mathbf{v}_{swd}$ with the nearest clasping target $\left(\mathbf{p}_{near}, \mathbf{p}'_{near}\right) \in G_\gamma$:

$$\Pi_{align} \colon \mathbf{q} \mapsto \begin{pmatrix} \left|\mathbf{v}_{swd} \cdot \mathbf{p}'_{near}\right| \\ \mathbf{v}_{fwd} \cdot \mathbf{t}_{near} \end{pmatrix}, \tag{3}$$

where $\mathbf{t}_{near} = \frac{\mathbf{p}_{near} - \mathbf{p}_{AABB}}{\|\mathbf{p}_{near} - \mathbf{p}_{AABB}\|}$ describes the normalized direction vector pointing from the hand towards the nearest clasping target. The task space projection satisfies $\Pi_{align}(\mathbf{q}) = (1,1)^{\mathrm{T}}$ if and only if the hand is pointing towards the target section of the loop and is aligned with the loop's local tangent. The action of closing the fingers around the loop, *(c)*, is described by the average virtual linking of all finger curves in task space $\mathcal{X}_{vlink}$:

$$\Pi_{vlink} \colon \mathbf{q} \mapsto \tfrac{1}{|C|} \textstyle\sum_{\alpha \in C} VLk(\gamma, \alpha). \tag{4}$$

TS-EXTEND can be considered as a randomized search for a good starting configuration from which iterated stepwise local control can be applied to reach the nearest clasping target. Consequently, every extension step should try to align the end effector with the nearest clasping target. This implies that the TS-RRT should only search $\mathcal{X} = \mathcal{X}_{pos} \times \mathcal{X}_{vlink}$ randomly, but the goals in $\mathcal{X}_{align}$ should not be random. A controller that governs the high-dimensional joint $\mathcal{C}$-space of arm and hand joints, $\mathcal{Q}$, is described in Alg. 4. The task space goals $\mathbf{c}_{pos}$ and $\mathbf{c}_{vlink}$ are extracted from dimensions 1 to 3 and 4 of the given target $\mathbf{x} \in \mathcal{X}$ while $\mathbf{c}_{align}$ is always set to $(1,1)^{\mathrm{T}}$. The use of a sigmoid-shaped gain function results in a smooth change from controlling position with alignment as a secondary task to prioritizing hand shape control. To limit conflicts and to focus control from different task spaces to either the arm or the hand joints, gain matrices $\mathbf{M}_{hand}$ and $\mathbf{M}_{arm}$ are used. The values are defined such that either all control for the hand or the arm is diminished and the variable influence of the joints depending on their kinematic chain position is accounted for. This controller essentially performs a reaching and then closing motion and is actually capable of clasping an unobstructed clasping goal point by itself.

Algorithm 4 TS-CONTROL$\left(\mathbf{q}_0, \mathbf{x}_0, \mathbf{x}_1, \gamma, G_\gamma\right)$

Require: $\mathbf{q}_0 \in \mathcal{Q}$, $\mathbf{x}_0, \mathbf{x}_1 \in \mathcal{X}$, γ loop, G_γ positions on loop with tangents
1: $\mathbf{c}_{pos} \leftarrow \left((\mathbf{x}_1)_1, (\mathbf{x}_1)_2, (\mathbf{x}_1)_3\right)^{\mathrm{T}}$
2: $\mathbf{u}_{pos} \leftarrow \left(\mathbf{c}_{pos} - \Pi_{pos}(\mathbf{q}_0)\right)$
3: $\mathbf{J}_{pos} \leftarrow$ COMPUTE-JACOBIAN$\left(\Pi_{pos}(.), \mathbf{q}_0\right)$
4: $\mathbf{c}_{align} \leftarrow (1,1)^{\mathrm{T}}$
5: $\mathbf{u}_{align} \leftarrow \left(\mathbf{c}_{align} - \Pi_{align}(\mathbf{q}_0, G_\gamma)\right)$
6: $\mathbf{J}_{align} \leftarrow$ COMPUTE-JACOBIAN$\left(\Pi_{align}(., G_\gamma), \mathbf{q}_0\right)$
7: $\mathbf{c}_{vlink} \leftarrow (\mathbf{x}_1)_4$
8: $\mathbf{u}_{vlink} \leftarrow \left(\mathbf{c}_{vlink} - \Pi_{vlink}(\mathbf{q}_0, \gamma)\right)$
9: $\mathbf{J}_{vlink} \leftarrow$ COMPUTE-JACOBIAN$\left(\Pi_{vlink}(., \gamma), \mathbf{q}_0\right)$
10: $\dot{\mathbf{q}}_{arm} \leftarrow \mathbf{J}^+_{pos}\mathbf{u}_{pos} + \left(\mathbf{I} - \mathbf{J}^+_{pos}\mathbf{J}_{pos}\right)\mathbf{J}^+_{align}\mathbf{u}_{align}$
11: $\dot{\mathbf{q}}_{hand} \leftarrow \mathbf{J}^+_{vlink}\mathbf{u}_{vlink}$
12: $\alpha \leftarrow$GAIN$\left(\|\mathbf{u}_{pos}\|\right)$
13: $\dot{\mathbf{q}} \leftarrow \alpha\mathbf{M}_{arm}\dot{\mathbf{q}}_{arm} +$
14: $\qquad(1-\alpha)\left(\mathbf{M}_{hand}\dot{\mathbf{q}}_{hand} + \left(\mathbf{I} - \mathbf{J}^+_{vlink}\mathbf{J}_{vlink}\right)\dot{\mathbf{q}}_{arm}\right)$
15: **return** $\mathbf{q}_0 +$ LIMIT-STEP-SIZE$(\Delta_t\dot{\mathbf{q}})$

V. Experimental Evaluation

We consider a KUKA KR5 R850 arm equipped with either an Armar III, DLR 1, iCub or Schunk SDH hand. The number of joints for the hands is 10 (Armar III), 12 (DLR 1), 15 (iCub), 6 (Schunk SDH) and 6 for the arm. Unless stated otherwise, the goal tolerance in the task space only considers virtual linking with a threshold of 0.85–0.65 for the DLR 1 and Schunk SDH hands, 0.8–0.65 for the Armar III and 0.75–0.65 for the iCub hand—depending on the shape of the object. Hand representations and finger curves are depicted in Fig. 4. The five real-sized experiment objects are displayed in Fig. 3 along with their representation created from 2000 points sampled on the object's visible surface. From the detected shortest loops, only the loop marked in *red* is used for clasping. The numbers of equally spaced clasping targets on the used loops are 9 (*Purse*), 23 (*Bag*), 17 (*Chair*), 24 (*Lawn Mower*) and 10 (*Travel Bag*).

The TS-RRT is programmed to randomly expand for 20, 50, 100 or 200 cycles before entering the loop in Alg. 1 to speedup the search—depending on the size, complexity and pose of the object. The value P is set to 0.1. The *Simox* [36] software is used for robot simulation and object mesh models are taken from [37], [38].

A. Varying Object Poses

To be viable, a motion and clasp planning approach needs to handle different robot hands, objects and object poses. Therefore, in this experiment, we position each of the objects in three different poses for each robot hand. The number of pre-expansions, $n \in \{20, 50, 100, 200\}$, is selected as low as possible so that a clasping goal is reached within the first few attempts to connect. The stability of the resulting clasping poses is tested in rigid body simulation by trying to separate the robot and object with 10 separate sequences of 500 random translations and rotations. Each transformation is pursued until collision. A subset of the 60 final poses is shown in Fig. 5, a comprehensive chart of all figures can be found at `http://www.csc.kth.se/~jastork/iros2013/`. Each of the resulting configurations passed the stability test and therefore provides a secure clasping pose.

In all cases, the goal was found within 52 cycles after the initial phase, while in 37 of the 60 cases the search terminated less then 10 cycles after the initial phase. The comparably low amount of cycles for a RRT approach shows that the proposed controller is suitable for clasping. The three poses of the *Bag* and the first two poses of the *Travel Bag* turned out to be the simplest clasping problems. In both cases, many kinematically easily reachable clasping goals are placed in the robot's workspace, so that $n = 20$ and $n = 50$ pre-expansions are sufficient. The *Purse* and the *Chair* object are more difficult to clasp. For the *Purse*, the loop is small compared to the DLR 1 and Schunk SDH hands. Additionally, the number of 9 clasping goals is comparably small and about half of them are obstructed by the object shape. All hands but the iCub hand require at least $n = 100$ pre-expansions to find a secure clasp. While the DLR 1 hand could quickly find clasps on the *Chair* object with $n = 20$ and $n = 50$, all the other hands needed $n = 50$ to $n = 200$ pre-expansions. The large fingers of the DLR 1 hand can easily link the loop without being precisely positioned. Despite its shape, the *Lawn Mower* is difficult to clasp. This can be explained by the fact that only a few clasping goals at the middle of the handle are easily reachable with a tangent-perpendicular pose.

The distribution of the minimal and maximal Gauss linking integrals for the finger curves of the individual final poses is depicted in Fig. 6. In all cases, the maximal Gauss linking is larger than 0.5 while most values are higher than 0.8. The minimal Gauss linking ranges from ca. 0.5 to 1.5. The largest spread of values occurs for the DLR 1 hand. The results show that a large range of Gauss linking integrals for finger and object curves corresponds to secure clasps.

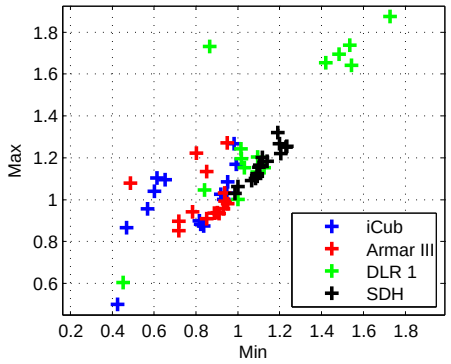

Fig. 6. Evaluation of the absolute values of the Gauss linking integral between object loops and robot hand for each successful clasping pose. For each final configuration, the minimal value for the hand's control curves is plotted against the maximal value.

B. Grasping an Obstructed Loop on a Chair

The advantage of planning is that local obstructions can be bypassed to reach a goal. To show that our approach is capable of clasping loops that cannot directly be reached from the initial position, the robot arm is placed over the chair's seat and a loop under the seat is chosen to be clasped (see Fig. 7). After 606 cycles, the search terminates. The search tree nodes are displayed in *red* showing the grasp center point. *Blue* markers are used for the solution nodes. The search tree spans from the initial position to the open space right of the robot, eventually connecting to a clasping goal. Poses at 20%, 40%, 60% and 80% of the optimized solution shown in *green* are displayed in the right of Fig. 7.

C. The Influence of Virtual Linking Tolerance

In our approach, a pre-defined task space goal tolerance is used to implicitly define the goal regions. Instead of increasing virtual linking, the TS-RRT could be used to only find configurations with virtual linking larger than 0 and execute automatic closing of the robot hand to grasp a loop segment. Here, we show the impact of the goal tolerance for the minimally accepted virtual linking. Different thresholds imply that different goal regions are considered. Fig. 8 shows possible results if a virtual linking of 0.5, 0.65, 0.70 or 0.75 is required to terminate the search. As can been seen clearly, larger values lead to increasingly tighter clasps. While for 0.5 the clasp might be obstructed, a value of 0.75 makes sure that a high virtual linking can be reached without any collision. Therefore it is clear that this tolerance has to be adjusted and that a low tolerance increasingly guarantees better final linking.

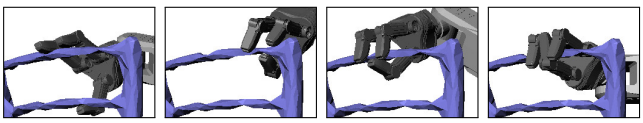

Fig. 8. *From left to right:* possible final configurations for increasing values of accepted minimal virtual linking (0.5, 0.65, 0.70, 0.75) for search termination. Higher values of virtual linking correspond to more secure clasps.

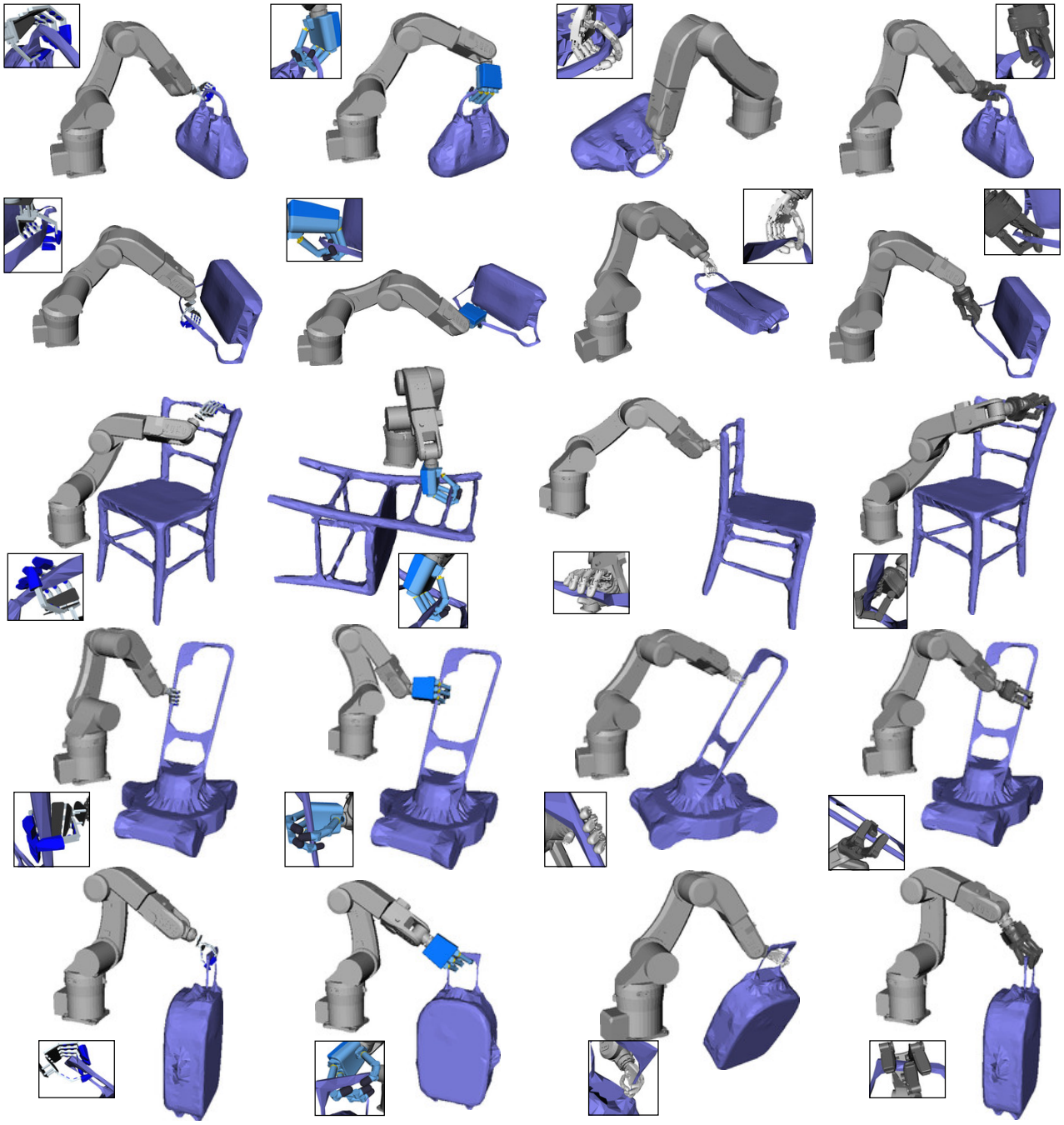

Fig. 5. A subset of the final poses generated for four hands and five objects is displayed. The small figures show details of the clasping pose. Each of the depicted poses is a secure clasping pose in the sense that it passes our pulling and turning test.

Fig. 7. Our approach can clasp loops that are not directly reachable from the robot's initial configuration. Here, the RRT spans from the initial configuration to the free space in front of the chair, visualized by the grasp center point in *red*. The solution's 30 nodes are shown in *blue* and the optimized solution is shown in *green*. The search terminated after 606 cycles and with 248 nodes in the tree.

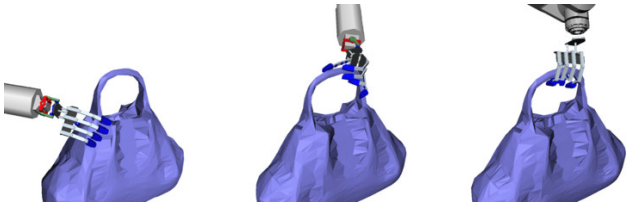

Fig. 9. *Left to right*: a representative grasp generated by the Grasp-RRT approach, a rare Grasp-RRT grasp remotely similar to a caging grasp and a representative clasping configuration as it is achieved by our approach.

D. Comparison to Grasp-RRT

In this experiment, we investigate whether a dedicated clasp planner is a necessity to generate clasping configurations. For comparison, the Grasp-RRT approach of [16] is used several times on *Purse*.

Most of the time, a grasp as displayed on the left of Fig. 9 is generated. The hand does not envelop the handle-part of *Purse* and has approached the object from the side or from below. In less then 5% of the experiments a grasp somewhere on the handle-part is generated and those grasps rarely envelop the handle-part. Compared to that, our approach reliably produces clasps as shown on the right of Fig. 9 every time. Our approach, which uses global topological object information and which generates caging rather than precision grasps, hence provides a complementary strategy for grasp synthesis.

VI. CONCLUSIONS

We have presented an integrated approach to clasp and motion planning based on the novel concept of virtual linking which applies to objects with holes. A clasp is a grasp that falls into the category of caging grasps and which extends the repertoire of a robot to actions that do not necessarily immobilize the manipulated object. We have defined multiple task spaces and employed a priority based control technique together with sampling-based motion planning. We have demonstrated the viability of our approach in simulation using several robot hands and objects. In future, we plan to evaluate our approach on a real robot and would like to deduce currently predefined parameters using a further analysis of the objects. Additionally, we would like to incorporate task constraints to select optimal handle-parts and clasping positions.

REFERENCES

[1] S. M. LaValle and J. J. Kuffner, "Randomized kinodynamic planning," *The International Journal of Robotics Research*, 2001.

[2] J. Kuffner, K. Nishiwaki, S. Kagami, M. Inaba, and H. Inoue, "Motion planning for humanoid robots," *Robotics Research*, 2005.

[3] J. Kim and J. P. Ostrowski, "Motion planning a aerial robot using rapidly-exploring random trees with dynamic constraints," in *IEEE ICRA*, 2003.

[4] D. Bertram, J. Kuffner, R. Dillmann, and T. Asfour, "An integrated approach to inverse kinematics and path planning for redundant manipulators," in *IEEE ICRA*, 2006.

[5] M. Vande Weghe, D. Ferguson, and S. S. Srinivasa, "Randomized path planning for redundant manipulators without inverse kinematics," in *IEEE-RAS Humanoids*, 2007.

[6] J. J. Kuffner Jr and S. M. LaValle, "RRT-connect: An efficient approach to single-query path planning," in *IEEE ICRA*, 2000.

[7] E. Drumwright and V. Ng-Thow-Hing, "Toward interactive reaching in static environments for humanoid robots," in *IEEE/RSJ IROS*, 2006.

[8] D. Berenson, S. S. Srinivasa, D. Ferguson, A. Collet, and J. J. Kuffner, "Manipulation planning with workspace goal regions," in *IEEE ICRA*, 2009.

[9] N. Vahrenkamp, D. Berenson, T. Asfour, J. Kuffner, and R. Dillmann, "Humanoid motion planning for dual-arm manipulation and re-grasping tasks," in *IEEE/RSJ IROS*, 2009.

[10] J. H. Reif, "Complexity of the mover's problem and generalizations extended abstract," in *IEEE Conference on Foundations of Computer Science*, 1979.

[11] L. Kavraki and J.-C. Latombe, "Randomized preprocessing of configuration space for path planning: Articulated robots," in *IEEE/RSJ IROS*. IEEE, 1994.

[12] L. E. Kavraki, P. Svestka, J.-C. Latombe, and M. H. Overmars, "Probabilistic roadmaps for path planning in high-dimensional configuration spaces," *Robotics and Automation, IEEE Transactions on*, 1996.

[13] S. M. LaValle, "Rapidly-Exploring Random Trees A New Tool for Path Planning," 1998.

[14] T. Siméon, J.-P. Laumond, J. Cortés, and A. Sahbani, "Manipulation planning with probabilistic roadmaps," *The International Journal of Robotics Research*, vol. 23, no. 7-8, pp. 729–746, 2004.

[15] K. Hauser and J.-C. Latombe, "Integrating task and PRM motion planning: Dealing with many infeasible motion planning queries," in *Workshop at ICAPS*, 2009.

[16] N. Vahrenkamp, T. Asfour, and R. Dillmann, "Simultaneous Grasp and Motion Planning: Humanoid Robot ARMAR-III," *Robotics & Automation Magazine*, 2012.

[17] A. Shkolnik and R. Tedrake, "Path planning in 1000+ dimensions using a task-space Voronoi bias," in *IEEE ICRA*, 2009.

[18] A. Shkolnik, M. Levashov, I. R. Manchester, and R. Tedrake, "Bounding on rough terrain with the LittleDog robot," *The International Journal of Robotics Research*, 2011.

[19] M. Behnisch, R. Haschke, and M. Gienger, "Task space motion planning using reactive control," in *IEEE/RSJ IROS*, 2010.

[20] M. Behnisch, R. Haschke, H. Ritter, and M. Gienger, "Deformable trees-exploiting local obstacle avoidance," in *IEEE-RAS Humanoids*, 2011.

[21] E. Rimon and A. Blake, "Caging 2D bodies by 1-parameter two-fingered gripping systems," in *IEEE ICRA*, 1996.

[22] C. Davidson and A. Blake, "Caging planar objects with a three-finger one-parameter gripper," in *IEEE ICRA*, 1998.

[23] P. Pipattanasomporn and A. Sudsang, "Two-finger caging of concave polygon," in *IEEE ICRA*, 2006.

[24] S. Makita and Y. Maeda, "3D multifingered caging: Basic formulation and planning," in *IEEE/RSJ IROS*, 2008.

[25] A. Rodriguez and M. Mason, "Two finger caging: squeezing and stretching," *Algorithmic Foundation of Robotics VIII*, 2009.

[26] A. Rodriguez, M. Mason, and S. Ferry, "From caging to grasping," *The International Journal of Robotics Research*, 2012.

[27] W. Wan, R. Fukui, M. Shimosaka, T. Sato, and Y. Kuniyoshi, "Grasping by caging: A promising tool to deal with uncertainty," in *IEEE ICRA*, 2012.

[28] R. Diankov, S. Srinivasa, D. Ferguson, and J. Kuffner, "Manipulation planning with caging grasps." in *IEEE-RAS Humanoids*, 2008.

[29] F. T. Pokorny, J. A. Stork, and D. Kragic, "Grasping objects with holes: A topological approach ," in *IEEE ICRA*, 2013.

[30] K. Klenin and J. Langowski, "Computation of writhe in modeling of supercoiled DNA," *Biopolymers*, 2000.

[31] E. L. Ho and T. Komura, "Character Motion Synthesis by Topology Coordinates," *Comput. Graph. Forum*, 2009.

[32] D. Zarubin, V. Ivan, M. Toussaint, T. Komura, and S. Vijayakumar, "Hierarchical Motion Planning in Topological Representations," in *RSS*, 2012.

[33] A. Hatcher, *Algebraic topology*. Cambridge University Press, 2002.

[34] O. Busaryev, T. Dey, J. Sun, and Y. Wang, "ShortLoop Software for Computing Loops in a Shortest Homology Basis," Software, 2010.

[35] T. Dey, J. Sun, and Y. Wang, "Approximating loops in a shortest homology basis from point data," in *ACM SoCG*, 2010.

[36] N. Vahrenkamp, M. Kröhnert, S. Ulbrich, T. Asfour, G. Metta, R. Dillmann, and G. Sandini, "Simox: A Robotics Toolbox for Simulation, Motion and Grasp Planning," *Intelligent Autonomous Systems*, 2013.

[37] Trimble 3D Warehouse, http://sketchup.google.com/3dwarehouse/.

[38] Archive3d.net, http://www.archive3d.net/.