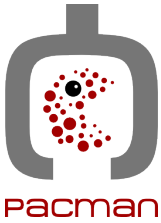


FP7-IST-60918
1 March 2013 (36months)

DR 4.2:
Methodologies for grasp acquisition and planning

H. Marino, C. Rosales, M. Kopicki, G. Santaera, E. Luberto, Y. Wu, T. Sartor, J.L. Wyatt, and M. Gabiccini

Università di Pisa, Pisa Italy
⟨m.gabiccini@ing.unipi.it⟩

Due date of deliverable:	2016-02-29
Actual submission date:	2016-02-29
Lead partner:	Università di Pisa
Revision:	final
Dissemination level:	PU

This report describes the activities related to the control of grasping actions given the detection of an object. It includes work on: modelling underactuated and adaptive robotic hands, acquiring target grasps learned from data gathered either from humans or simulations, synthesising grasps for novel objects using either heuristic or statistical machine learning approaches, planning complex grasping and manipulation tasks either using discrete search methods or via direct trajectory optimization methods for systems with unspecified contact sequences.

Executive Summary

This report describes the activities — most of which are already performed, while some others scheduled for finalization in the weeks that precede the PaCMan review meeting in May 2015 — within the PaCMan consortium to define methodologies for *incipient grasping evaluation*. The material included in this report describes the results that fall under Task 4.2 (M 7-18), and the progress of Task 4.3 (M 7-30) at M 24.

Role of incipient grasp evaluation in PaCMan

This deliverable documents the effort of the consortium in devising new strategies to define *practical* approaches to tackle the problem of *grasping under uncertainty and novelty*, either caused by clutter, visual occlusions, or new object shapes. To this sake, the ability to evaluate *incipient grasps* in terms of their expected quality, robustness, and probability of success, is of paramount importance.

Contribution to the PaCMan scenario

To our eyes, grasp methodologies that try to cope with pose uncertainty, object shape estimation inaccuracies, clutter and partial occlusions, either (i) by accomodating grasp unmodelled dynamics by exploiting prior knowledge about the adaptivity and the compliance of the Pisa/IIT SoftHand (as mainly proposed by the UNIPI Team), or (ii) by inferring grasps for novel objects from as little as one training example (of a chosen grasp type) that are shown to be generalizable within and *across* object categories, even with partial shape information, (as mainly proposed by the UoB Team), represent central contributions in the scenario depicted in the PaCMan project.

1 Tasks, objectives, results

1.1 Planned work

Humans have no troubles in selecting the correct grasp strategies even if the object to be grasped is unknown or partially visible. On the contrary, though robotics has made significant progress in the field of autonomous grasping, the problem of grasping novel objects in cluttered scenarios has not been completely solved yet. In this report, we document the activities of the consortium to tackle this problem.

The work to be documented in this report regards: "Methodologies for incipient grasp evaluation". This should mostly contain the results of Task 4.2 and the progress of Task 4.3 and Task 4.4 at this stage. In particular, the report should contain a description of the procedures to infer first-order object surface properties and the quality of the associated grasp from fingerpad sensor recordings, and algorithms for local optimization of contact point locations and on-line grasping force optimization.

1.2 Actual work performed

We have to clearly state, right away, that not all the pieces of work presented in this report, even if sharing the very same goals of WP4, i.e., "to develop methods to control grasping actions given a detection of an object using the compositional object model", fit exactly the breakdown structure suggested by the Tasks reported in the "Description of Work". In particular, the description of the activities to be performed in Tasks 4.1, 4.2 and 4.3, represents the sequential temporal stages that one could envision in performing a robust grasp when adopting a fully actuated and sensorized robotic hand. In fact, the pre-shaping of the hand to a first object contact (Task 4.1), the assessment of the first-order properties of the object surface, along with the local optimization of the contact point locations (Task 4.2), and the actual grasp acquisition with the optimal distribution of contact forces (Task 4.3), indeed assume that the action is performed by employing a dexterous hand with a plethora of force and position sensors. Moreover, here the robustness of the grasp was envisioned by reasoning, at each stage of the process, whether the previous step was successfully performed and taking proper corrective actions if that was not the case.

The approach that we came to pursue sees instead a central role of compliantly underactuated hands, where uncertainty in the relative pose between the hand and the object and object shape itself is mechanically absorbed by the adaptive behavior of the Pisa/IIT SoftHand. This new research direction brought about somewhat completely unplanned (and unexplored) issues that we considered central to the goals of WP4 and worth to be pursued. First of all, the sensorization of a compliantly adaptive hand

with dislocable joints asked for a whole new approach to provide proprioceptive feedback to the hand (this topic is actually matter of DR 3.1). The force sensorization problem, which is somewhat assumed in the Tasks 4.2 and 4.3, poses non trivial technological problems for the hand under investigation, and is still under development and testing. Moreover, the envisioned phases to transit from grasp formation to successful grasp acquisition are inextricably entangled and somehow blurred for an adaptive and underactuated hand with just one single Degree of Actuation, as the dynamic interaction between the hand and the object — and possibly the environment — is responsible for the global behavior of the system. This new perspective asked for a rescheduling and reorganization of the needed pieces of work.

For the reasons mentioned above, the work that is presented in this report is mostly focused on proposing robust grasping solution for an adaptive hand and, in particular, for our Pisa/IIT SoftHand. The learning and transfer of grasps within and across object categories, instead, makes use of the fully actuated and sensorized 20-DoF DLR-HIT Hand II, since a finer control over the fingertip position is required.

In Sec. 1.2.1, we present our approach to define successful synthetic poses for the Pisa/IIT SoftHand in grasping the PaCMan Object Database [1]. In Sec. 1.2.2, the previous approach is sharpened by employing a custom decomposition into cuboids of the object point cloud. Each cuboid suggests a number of grasps to be performed on the object. These grasps are used to warm start the search for successful grasps to be synthetically found in the simulation environment and to be attached to that object's grasp database. The feasibility of using this approach on-line to find graspable parts on objects in the scene with partial point cloud information and irrespective to their prior recognition is under investigation using the framework described in MS 4.2, Sec. A.1-A.4.

In Sec. 1.2.3, we shift our attention onto the generalization and adaptation ability of the previously reported grasp planning algorithm for fully actuated hands showing, in experiments using the DLR-HIT Hand II, how it is possible to just use a few example grasps, taught in a kinesthetic manner, in order to be able to transfer this knowledge to unknown, even partially visible objects.

In Sec. 1.2.4, we consider the grasps of the previous section as the starting point, but now wish to execute them more reliably than in an essentially open loop fashion. We present the progress on our method for planning tactile information gathering to reduce the uncertainty in the pose of an object. The method works by embedding the information to be gained in the physical space, using it to shorten distances, and thus make information gathering trajectories of lower cost. We show improvement in our results over the previous year.

Sec. 1.2.5 presents a new approach to planning for soft robot manipulators under task constraints. This piece of work is central to motion planning

in dual arm configurations, e.g. when an object is held simultaneously with two compliant underactuated hands. The corresponding control problem, which makes use of a geometric formulation and is based on a non-interacting scheme, is tested on a dual-arm (closed kinematic loop) configuration.

Sec. 1.2.6 presents our approach to solve high-level planning for grasping an object and passing it from one hand to another. This work constitutes the theoretical foundation for the high-level planning needed to perform the Task 5.3.

In Sec. 1.2.7, we describe a planning strategy based on direct trajectory optimization that does not require a-priori specification of the contact sequence and presents the benefits of providing dynamically consistent plans, from the outset, and allowing for opportunistic exploitation of environmental constraints.

1.2.1 Grasping With Soft Hands

In this section, we summarize the work performed to tackle the problem of grasp planning and grasp acquisition for hands that are simple — in the sense of low number of actuated degrees of freedom (one degree of actuation for the Pisa/IIT SoftHands) — but are soft, i.e. continuously deformable in an infinity of possible shapes through interactions with objects. This scenario presented us many different issues that could hardly be reconciled with the more classical approach to grasping — to be performed with fully-actuated and sensorized robotic hands — which is entailed in the consecutive temporal stages of grasp formation described Tasks 4.1, 4.2 and 4.3. Here, we face them all simultaneously, as we employ an adaptive underactuated hand to perform the task. This should explain why the material presented in this section is not exactly aligned to what the description of the Tasks presents. However, we believe that the work performed share with the description of the Tasks the same ultimate goals: robust grasping in uncertain scenarios.

This research topic was started during the first year of the project — it was already documented in DR 4.1 in the preliminary form of a technical report — and has now reached a higher level of maturity, as testified by the accompanying paper [2].

During the past thirty years, the problem of autonomous grasping has been one of the most widely investigated. Several approaches have been proposed to define the optimal finger placement on the object, either based on geometric [3] or force features [4], on our previous work on grasp quality measures [5], specifically tailored to convex objects [6] or generalized to non-convex ones.

Perhaps because of the fragility of the mechanics of most robot hands, the multitude of the planning methods were thought for interactions between the hand and the objects that only occur at the fingertips, limiting contacts with other parts of the hand and avoiding contacts with the rest of the envi-

ronment at all. This "timid" approach to manipulation generated by rigidity and fragility of the hand has been recently challenged by the introduction of adaptable, underactuated and/or soft hands. This allows to use these hands in a more "daring" way with the objects in the environment, using their full surface for enveloping grasps, and exploiting object and environmental constraints to functionally shape the hand, going beyond its nominal kinematic limits through wise exploitation of their structural softness.

The analysis of these possibilities constitute a new challenge for existing grasping algorithms: adaptation to totally of partially unknown scenes remains a difficult task, towards which only some approaches have been investigated so far.

Here, moving beyond the state of the art, we present a first approach to explore this novel kind of manipulation, based on an accurate simulation tool for the SoftHand, developed using the multi-body system software ADAMS. A batch simulation setup was created and used to perform the automatic creation of a database of grasp affordances for the Pisa/IIT SoftHand on the PaCMan database of kitchenware objects.

More details on this new approach and on the achieved results can be found in Sec. A.1.

1.2.2 Grasp Planning for the Pisa/IIT Softhand via Bounding-Box Object Decomposition

In this section, we summarize the work performed to sharpen the approach of grasp planning and grasp acquisition for adaptive and compliant hands, such as the Pisa/IIT SoftHand. The ability we try to embed in our grasp planner is that of selecting a successful grasp without the need of identifying *a priori* an object, but only based on its composition of shape primitives with known associated grasps, i.e. resorting to *part-based grasps*.

To this sake, we follow a mid-level solution, according to a purely top-down strategy, based on the Minimum Volume Bounding Box (MVBB) *fit-and-split* algorithm to object decomposition that was originally proposed in [7]. The method consists of iteratively building MVBBs of points resulting from splitting the point cloud of the object. The split procedure is performed in such a way that the sum of the two areas (proper box projections) resulting from the convex region of each set of points is minimized.

We propose an improved version of the MVBB algorithm for object decomposition in [7], as it presents the following properties: (i) it is very efficient and can accomodate scattered 3D points delivered by arbitrary 3D sensors; (ii) the outcome constellation of boxes is quite insensitive to noise. However, with respect to [8] and [9], where 2D grasp hypotheses are made and evaluated on point projections onto box faces and assume, from the outset, the use of rigid and fully-actuated robotic hands or grippers, in our work we propose a shift of prospective in the creation of grasp hypotheses

as we employ a compliant and adaptive hand, the Pisa/IIT SoftHand. We affirm that this contribution represents an advancement with respect to the state of the art.

With a reduced burden to plan detailed finger placements, the box set representation of an object, also ease the mapping of complex actions to box and/or box distribution properties. For example, as also mentioned in [8], in order to *pick-up* an object and place it somewhere, it may intuitively be a good option to grasp the *largest box*. Instead, in order to show the same object to a camera to gather more views, it may be preferable to grasp the object from the *outermost box*, or from a box that allows a pincher grasp, so to minimize occlusions caused by hand parts.

Considering the adaptability of soft hands, we present a strategy to propose grasp postures to grasp a subset of the PaCMan kitchenware object database previously decomposed into MVBBs. The performances of the method are compared with those presented in [2] and, through simulations performed with the MBS software ADAMS^{TM} [10], we show that with the strategy presented in this paper we increase the successful rate of grasps for the same objects.

More details on this new approach and on the achieved results can be found in Sec. A.2.

1.2.3 Grasping Novel Objects with a Dexterous Hand

In this section we show the extension of our approach to adaptation of a dexterous grasp type from year one, which is the other modus operandi we followed to tackle Task 4.3. We now solve two problems: *grasp type selection* and *adaptation of a grasp type*. The complete method enables learning of several grasp types, each from one kinesthetic demonstration. Thus the grasp type learning is one-shot. When presented with a novel object in any orientation the grasp generation algorithm generates hundreds of candidate grasps from each grasp type and ranks them according to likelihood. The extended method has been implemented on Boris, the bimanual robot platform at Birmingham (using a DLR-HIT Hand II) , having previously been tested on the Innsbruck robot with a different hand (Schunk 3-finger hand).

Previous work in learning generalisable grasps falls broadly into two classes. One class of approaches utilises the shape of common object parts or their appearance to generalise grasps across object categories [11, 12, 13, 14]. This works well for low DoF hands. Another class of approaches captures the global properties of the hand shape either at the point of grasping, or during the approach [15]. This global hand shape can additionally be associated with global object shape, allowing generalisation by warping grasps to match warps of global object shape [16]. This second class works well for high DoF hands, but generalisation is more limited. We achieve the advantages of both classes, generalising grasps across object categories with high DoF hands.

Both the learning and generation stages of our approach use an incomplete point cloud from a depth camera – no prior model of object shape is used. As previously, the learned model is a product of experts, in which experts are of two types. The first is a *contact model* and is a density over the pose of a single hand link relative to the local object surface. The second is the *hand configuration model* and is a density over the whole hand configuration. The grasp inference implementation for novel objects (grasp transfer), which optimises the product of these two model types, has been improved to make it faster, generating thousands of grasp candidates in seconds. Also with improvements in computing power the time to compute a set of grasps has been reduced by a factor of more than 10. The key step in grasp transfer is the computation of a *query density* for each hand link. This is achieved by a Monte Carlo procedure that convolves the point cloud for the new object with the expert for an individual hand link.

In experiments with Boris, the training set consisted of five different grasps, and the test set of forty-five previously unseen objects. The success rate of the first choice grasp is 77.7% using a single view of the test object, and increases to 84.4% if seven views are taken instead.

More details on the highlighted method and the experimental results can be found in Sec. A.3.

1.2.4 Grasping under Uncertainty

This work extends that of [17], which offers a way to avoid the complexity of planning in a high dimensional belief space. It does this in two ways, i) by approximating the informational value of actions from a low-dimensional subspace of the belief state; and ii) by embedding that informational value into the physical space. Whereas Platt employed a 2DoF arm with a laser sensor, our extension allows reasoning about a 21 DoF arm-hand system with tactile observations.

In our previous work [18, 19] we proposed a information gain re-planning strategy to improve the reliability of grasping under uncertainty. We achieved this by using a hierarchical sample-based path planner, here a Probabilistic Roadmap (PRM) planner, which explicitly encodes expected information gain (using a low-dimensional approximation of the belief state) along each of its trajectory segments. This extended the work in [17]. A particular contribution of our approach is refining the belief state for the object pose using an observation model for tactile sensing by a multi-finger hand that can palpate the object to be grasped.

We have extended our previous work in four ways. First we now employ as the target grasp, one of the grasp candidates from the work on grasping of novel objects described elsewhere in this deliverable. Second we now enable re-planning from the configuration of the robot at the point where an unexpected contact occurs. This is instead of moving the hand away from

the contact to a safe configuration. Third, we now plan dexterous grasping trajectories for non-convex object shapes by implementing an efficient collision detection method which reasons directly with sensed point clouds rather than derived representations of object shape such as mesh representations. Fourth, we now use an active compliant controller to ensure that continued contact(s) between the hand and the object minimise the risk of perturbing the system.

The updated method has been tested in simulation. Pose uncertainty is now caused by errors in matching a partial point cloud from one or two viewpoints to a previously sensed cloud from seven views. In each trial the simulated robot attempts to grasp the presented object using three different strategies. The first strategy represents a "naive" approach in which the robot tries to grasp the object in its estimated pose with a single attempt. The second strategy allows the robot to recover from unexpected contacts if the estimated pose is not correct, and a new grasp attempt is planned and executed until a termination condition is met. The third strategy is teh information gain strategy mentioned above. This works with a cost function that allows deviations from the shortest physical path in order to make tactile contacts that will reduce object pose uncertainty. Essentially this can be seen as warping the physical space using information gain to create a non-Euclidean distance metric for motion planning. The goal of the trials is to show that the information gain planner is superior, having a higher success rate, and requiring fewer re-planning iterations than the other methods before a grasp is achieved.

1.2.5 Sample-Based Motion Planning for Soft Robot Manipulators Under Task Constraints

In this section we present a new sample-based approach to motion planning for soft robot manipulators which considers also task constraints.

As our final goal is to use a bimanual system to grasp and position objects, motion planning cannot neglect the constraints arising from closing the kinematic loop between the two arms, the two hands, and the object being grasped. Moreover, while using compliant manipulators gives new and unforeseen opportunities, it also poses new challenges even on the motion planning side.

Normally, random sampling-based methods for motion planning such as [20, 21] have the advantage to efficiently sample the robot configuration space (for which we have an explicit parameterization) and may iteratively improve the connection between an initial and a desired final robot configurations. This advantage deteriorates when we consider constrained motion planning because the lack of an explicit parameterization of the non linear submanifold of the configuration space which satisfies the constraints makes it difficult to find samples which are valid nodes of the plan to be

found. Explicitly, the fraction of the free-from-obstacle configuration space $\mathcal{M}_{\text{free}} \in \mathbb{R}^d$ which strictly satisfies a constraint of the type $C(q) = 0$, is a zero-measure submanifold $\mathcal{M}_v$ for which the practical probability of sampling a point $q \in \mathcal{M}_v$ is zero. Previous approaches are based either on the decomposition of the chain in an active and a passive chains [22], or in the projection of any random sample to the valid configuration space [23, 24], slowing down the planning process.

Specializing the algorithm for compliant systems, we can avoid this increase in computational burden relaxing the constraint to be of the type $C(q) \leq \varepsilon$: in the case in which the tight constraint is violated, a force proportional to the violation arises between two parts in contact, but still the configuration is feasible (although undesired forces on the grasped object could be generated). In this way, we can sample a boundary layer $\mathcal{M}_r := \{q \in \mathcal{M}_{\text{free}}, C(q) \leq \varepsilon\}$ which has the same dimensionality as the whole configuration space, and volume decreasing with ε. We can then use a biased random sampling technique like the one in [25], which allows us to obtain a sample distribution which tends to be uniform in $\mathcal{M}_r$.

Once the planning problem is solved, the main challenge arises from the fact that the relaxed constraint has now to be enforced during execution in order to avoid undesired forces acting on the object. This has to be ensured via a real-time controller, which can be synthesised exploiting the geometric formulation introduced in [26, 27] where an algorithm to make the position and force control of certain systems non-interacting has been devised, allowing for regulation of the internal object forces to an appropriate level without jeopardising the planned motion.

More details on this new approach and on the currently achieved results can be found in Sec. A.5.

1.2.6 High-Level Planning for Bimanual Object Passing

In order to successfully complete Task 5.3, which requires the object to be passed between the hands of the robot, we describe in this section our approach to solve high-level planning for this matter.

Our idea considers the possibility of having the object in a known position, which can be recognized using vision, and a higher-level agent (such as a user) decides a target configuration the object has to reach.

From this information, a semantic graph is constructed which has as nodes a grasp id and a workspace id, and as arcs all possible known transitions between nodes.

Once a minimum cost path has been found, it is translated back into Cartesian information for collision-free motion planning, grasp commands, and all low-level requirements to execute the plan successfully.

While there are many previous works on combining high-level semantic reasoning and low-level cartesian path planning (see e.g. [28, 29, 30]) which

mostly take advantage of object-specific reasoning introduced in [31], our main contribution is to include the possibility of passing an object between two end-effectors.

Specifically, considering end-effectors with different properties, the table (and possibly other non-movable surfaces) has its own set of grasps for an object, and is treated exactly like a hand when generating arcs in the graph, apart from the fact that it cannot be moved and, thus, there will be no arc connecting the same "table grasp" in adjacent workspaces (while there could be one if the grasp is performed via a hand of the robot).

Thus, considering that previous approaches mostly rely on a table in order to move an object from one arm workspace to another, we can simply classify the *primitives* which allow a transition from a grasp/workspace pair into another in three categories:

1. *pick* with an end-effector from a fixed surface
2. *move* the end-effector
3. *place* with an end-effector onto a fixed surface

Our approach uses instead a more general action of "grasp transfer", which can be specialized in:

1. moving actions of an end-effector among its reachable workspaces
2. pick and place actions if one end-effector involved is non-movable (such as a table)
3. a grasp-ungrasp sequence if both end-effectors involved are movable

This list is still under development, and we will try to generalize it even more in the future with other, more interesting *primitives* which could possibly exploit dynamics, friction, gravity, and so on.

A more detailed view on this method and on the achieved results can be seen in Sec. A.6.

1.2.7 A Computational Framework for Environment-Aware Robotic Manipulation Planning

Moving beyond with respect to what we committed to investigate in Tasks 4.2 and 4.3, we introduce, in this section, a computational framework for direct trajectory optimization of general manipulation systems without *a-priori* specified contact sequences, possibly exploiting *environmental constraints* as a tool to accomplish a task.

Originally, we planned to approach the problem of robust grasping by dividing it into three main stages: (i) move from hand pre-shape to first

object contact (Task 4.1), (ii) perform an incipient grasp and assess the first-order properties of the surface of the object, possibly changing candidate contact locations so that the quality of the incipient grasp is maximized (Task 4.2), and (iii) perform the actual grasp acquisition by optimizing the distribution of the applied contact forces (Task 4.3).

In this section, we report on our efforts to devise a framework to be employed not only for grasp planning, but also for manipulation planning, that should be able to merge all three previous phases in a systematic and coherent way. The user should be focused on providing high-level objectives, e.g. "move object A from pose 1 to pose 2 in a given amount of time", and the framework should be able to provide a manipulation plan that should take care of all the rest, e.g. should specify low-level actions and their correct sequence for the manipulation system at hand (single-arm configuration, dual-arm configuration), defining the whole contact sequence (where and when to make and to break contacts), should provide trajectories consistent with the dynamics of the manipulation system, unilateral contacts, friction constraints and actuation limits.

Related work to which we compare ours indeed include those using: (i) *traditional grasp planners*, such as [32] and [33]; (ii) *general purpose planning algorithms*, such as [34] and [35]; (iii) *machine learning approaches*, as [36] and [37]; (iv) *optimization-based trajectory planners*, such as [38], [39], and [40]. In particular, with respect to [40], which is the closest to ours, we can affirm that, besides a much more efficient computational pipeline, by explicitly instantiating environmental contact forces, we fabricate a strategy such that, if the task may be more efficiently and/or more robustly performed with the aid of constraints provided by the environment, the proposed approach can devise manipulation strategies that cleverly and opportunistically exploit environmental constraints.

Moreover, two approaches are presented to model the dynamics of systems with intermittent contacts: the first one, in which continuous contact reaction forces are generated by nonlinear virtual springs, is convenient to tackle scenarios where we try to avoid *sliding* contacts; the second one, which is based on a velocity-based time stepping scheme, is suitable in scenarios where *sliding* interaction primitives may lead to convenient interactions among the hand, the manipulandum and the environment.

In both cases, beside system's state and applied torques, object and environment contact forces are included among the free optimization variables, and they are rendered consistent via suitably devised sets of *complementarity* conditions. To maximize computational efficiency, sparsity patterns in the linear algebra expressions generated during the solution of the optimization problem are exploited, and Algorithmic Differentiation (also known as Automatic Differentiation) is leveraged to calculate derivatives. These aspects appear completely unexplored in the literature of high-level planning for systems with intermittent contacts.

The approach is evaluated in three simulated planar manipulation tasks: (i) moving a circular object from an initial pose to a final pose in the workspace with two independent fingers, (ii) rotating a capsule-shaped object with an underactuated two-fingered gripper, and (iii) rotating a circular object in hand with three independent fingers. Tasks (i) and (ii) show that our algorithm quickly converges to locally optimal solutions that opportunistically exploit environmental constraints. Task (iii) demonstrates that even dexterous fingertip gaits can be obtained as a special solution in the very same framework.

More details on this new approach and on the achieved results can be found in Sec. A.7.

References

[1] PaCMan Object Datasets. [Online]. Available: http://www.pacman-project.eu/datasets/

[2] M. Bonilla, E. Farnioli, C. Piazza, M. Catalano, G. Grioli, M. Garabini, M. Gabiccini, and A. Bicchi, "Grasping with soft hands," in *IEEE-RAS Int. Conf. on Humanoid Robots (Humanoids)*, Madrid, Spain, November 2014.

[3] C. Ferrari and J. Canny, "Planning optimal grasps," *Proceedings 1992 IEEE International Conference on Robotics and Automation*, pp. 2290–2295, Jan. 1992.

[4] D. Prattichizzo, J. Salisbury, and A. Bicchi, "Contact and grasp robustness measures: Analysis and experiments," *Experimental Robotics IV*, Jan. 1997.

[5] C. Rosales, R. Suarez, M. Gabiccini, and A. Bicchi, "On the synthesis of feasible and prehensile robotic grasps," in *Robotics and Automation (ICRA), 2012 IEEE International Conference on*, 2012, pp. 1–7.

[6] M. Teichmann and B. Mishra, "Reactive algorithms for grasping using a modified parallel jaw gripper," in *Robotics and Automation, 1994. Proceedings., 1994 IEEE International Conference on*, 1994, pp. 1–6.

[7] K. Huebner, S. Ruthotto, and D. Kragic, "Minimum volume bounding box decomposition for shape approximation in robot grasping," in *Robotics and Automation, 2008. ICRA 2008. IEEE International Conference on*, May 2008, pp. 1628–1633.

[8] K. Huebner and D. Kragic, "Selection of robot pre-grasps using box-based shape approximation," in *Intelligent Robots and Systems, 2008. IROS 2008. IEEE/RSJ International Conference on*, Sept 2008, pp. 1765–1770.

[9] S. Geidenstam, K. Huebner, D. Banksell, and D. Kragic, "Learning of 2D grasping strategies from box-based 3D object approximations," in *Proceedings of Robotics: Science and Systems*, Seattle, USA, June 2009.

[10] MSC Software. (2015) Adams. [Online]. Available: http://web.mscsoftware.com/

[11] A. Saxena, L. L. Wong, and A. Y. Ng, "Learning grasp strategies with partial shape information." in *AAAI*, vol. 3, no. 2, 2008, pp. 1491–1494.

[12] R. Detry, C. H. Ek, M. Madry, and D. Kragic, "Learning a dictionary of prototypical grasp-predicting parts from grasping experience,"

in *Robotics and Automation (ICRA), 2013 IEEE International Conference on*. IEEE, 2013, pp. 601–608.

[13] A. Herzog, P. Pastor, M. Kalakrishnan, L. Righetti, J. Bohg, T. Asfour, and S. Schaal, "Learning of grasp selection based on shape-templates," *Autonomous Robots*, vol. 36, no. 1-2, pp. 51–65, 2014.

[14] O. Kroemer, E. Ugur, E. Oztop, and J. Peters, "A kernel-based approach to direct action perception," in *Robotics and Automation (ICRA), 2012 IEEE International Conference on*. IEEE, 2012, pp. 2605–2610.

[15] H. Ben Amor, O. Kroemer, U. Hillenbrand, G. Neumann, and J. Peters, "Generalization of human grasping for multi-fingered robot hands," in *Intelligent Robots and Systems (IROS), 2012 IEEE/RSJ International Conference on*. IEEE, 2012, pp. 2043–2050.

[16] U. Hillenbrand and M. A. Roa, "Transferring functional grasps through contact warping and local replanning," in *Intelligent Robots and Systems (IROS), 2012 IEEE/RSJ International Conference on*. IEEE, 2012, pp. 2963–2970.

[17] R. Platt, L. Kaelbling, T. Lozano-Perez, and R. Tadrake, "Simultanious localization and grasping as a belief space control problem," CSAIL, MIT, Tech. Rep., 2001.

[18] C. Zito, R. Stolkin, M. S. Kopicki, M. D. Luca, and J. L. Wyatt, "Exploratory reach-to-grasp trajectories for uncertain object poses." in *Proc. Workshop on Beyond Robot Grasping: Modern Approaches for Dynamic Manipulation. Intelligent Robots and Systems (IROS)*, 2012.

[19] C. Zito, M. Kopicki, R. Stolkin, C. Borst, F. Schmidt, M. A. Roa, and J. Wyatt, "Sequential trajectory re-planning with tactile information gain for dextrous grasping under object-pose uncertainty," in *IEEE Proc. Intelligent Robots and Systems (IROS)*, 2013.

[20] 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*, vol. 12, no. 4, pp. 566–580, 1996.

[21] S. M. LaValle, "Rapidly-exploring random trees: A new tool for path planning," 1998.

[22] J. Cortes, T. Siméon, and J.-P. Laumond, "A random loop generator for planning the motions of closed kinematic chains using prm methods," in *Robotics and Automation, 2002. Proceedings. ICRA'02. IEEE International Conference on*, vol. 2. IEEE, 2002, pp. 2141–2146.

[23] M. Stilman, "Global manipulation planning in robot joint space with task constraints," *Robotics, IEEE Transactions on*, vol. 26, no. 3, pp. 576–584, 2010.

[24] D. Berenson, "Constrained manipulation planning," Ph.D. dissertation, Citeseer, 2011.

[25] J. Bialkowski, M. Otte, and E. Frazzoli, "Free-configuration biased sampling for motion planning," in *Intelligent Robots and Systems (IROS), 2013 IEEE/RSJ International Conference on*. IEEE, 2013, pp. 1272–1279.

[26] D. Prattichizzo and A. Bicchi, "Consistent task specification for manipulation systems with general kinematics," *Journal of dynamic systems, measurement, and control*, vol. 119, no. 4, pp. 760–767, 1997.

[27] ——, "Dynamic analysis of mobility and graspability of general manipulation systems," *Robotics and Automation, IEEE Transactions on*, vol. 14, no. 2, pp. 241–258, 1998.

[28] L. Karlsson, J. Bidot, F. Lagriffoul, A. Saffiotti, U. Hillenbrand, and F. Schmidt, "Combining task and path planning for a humanoid two-arm robotic system." Citeseer, 2012.

[29] D. Leidner, C. Borst, and G. Hirzinger, "Things are made for what they are: Solving manipulation tasks by using functional object classes," in *Humanoid Robots (Humanoids), 2012 12th IEEE-RAS International Conference on*. IEEE, 2012, pp. 429–435.

[30] D. Leidner and C. Borst, "Hybrid reasoning for mobile manipulation based on object knowledge," in *Workshop on AI-based Robotics at IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2013.

[31] L. Levison, "Connecting planning and acting via object-specific reasoning," Ph.D. dissertation, Citeseer, 1996.

[32] C. Rosales, "Grasp planning under task-specific contact constraints," Ph.D. dissertation, Universitat Politècnica de Catalunya, 2013.

[33] A. Miller and P. Allen, "Graspit! a versatile simulator for robotic grasping," *IEEE Robotics and Automation Magazine*, vol. 11, no. 4, pp. 110–122, 2004.

[34] Y. Koga and J.-C. Latombe, "On multi-arm manipulation planning," in *IEEE Int. Conf. on Robotics and Automation (ICRA)*, 1994.

[35] B. Cohen, M. Phillips, and M. Likhachev, "Planning single-arm manipulations with n-arm robots," in *Robotics: Science and Systems (RSS)*, 2014.

[36] S. Levine and P. Abbeel, "Learning neural network policies with guided policy search under unknown dynamics," in *Neural Information Processing Systems (NIPS)*, 2014.

[37] S. Levine, N. Wagener, and P. Abbeel, "Learning contact-rich manipulation skills with guided policy search," 2015, under review.

[38] I. Mordatch, Z. Popović, and E. Todorov, "Contact-invariant optimization for hand manipulation," in *Eurographics/ACM Symposium on Computer Animation*, 2012.

[39] I. Mordatch and E. Todorov, "Combining the benefits of function approximation and trajectory optimization," in *Proceedings of Robotics: Science and Systems*, Berkeley, USA, July 2014.

[40] M. Posa, C. Cantu, and R. Tedrake, "A direct method for trajectory optimization of rigid bodies through contact," *Int. Journal of Robotics Research (IJRR)*, vol. 33, no. 1, pp. 69–81, 2014.

A Annexes

A.1 Article: Grasping with soft hands

Authors M. Bonilla, E. Farnioli, C. Piazza, M. Catalano, G. Grioli, M. Garabini, M. Gabiccini, A. Bicchi

Info In proceedings of IEEE-RAS Int. Conf. on Humanoid Robots (Humanoids), 2014

Abstract Despite some prematurely optimistic claims, the ability of robots to grasp general objects in unstructured environments still remains far behind that of humans. This is not solely caused by differences in the mechanics of hands: indeed, we show that human use of a simple robot hand (the Pisa/IIT SoftHand) can afford capabilities that are comparable to natural grasping. It is through the observation of such human-directed robot hand operations that we realized how fundamental in everyday grasping and manipulation is the role of hand compliance, which is used to adapt to the shape of surrounding objects. Objects and environmental constraints are in turn used to functionally shape the hand, going beyond its nominal kinematic limits by exploiting structural softness. In this paper, we set out to study grasp planning for hands that are simple - in the sense of low number of actuated degrees of freedom (one for the Pisa/IIT SoftHand) - but are soft, i.e. continuously deformable in an infinity of possible shapes through interaction with objects. After general considerations on the change of paradigm in grasp planning that this setting brings about with respect to classical rigid multi-dof grasp planning, we present a procedure to extract grasp affordances for the Pisa/IIT SoftHand through physically accurate numerical simulations. The selected grasps are then successfully tested in an experimental scenario.

Relation with the deliverable: Grasping under uncertainty by exploiting the adaptive behavior encoded in the mechanical design.

Attachment (following pages until next annex)

Grasping with Soft Hands

M. Bonilla[1], E. Farnioli[1,2], C. Piazza[1], M. Catalano[2], G. Grioli[2], M. Garabini[1], M. Gabiccini[1,2,3], A. Bicchi[1,2,4]

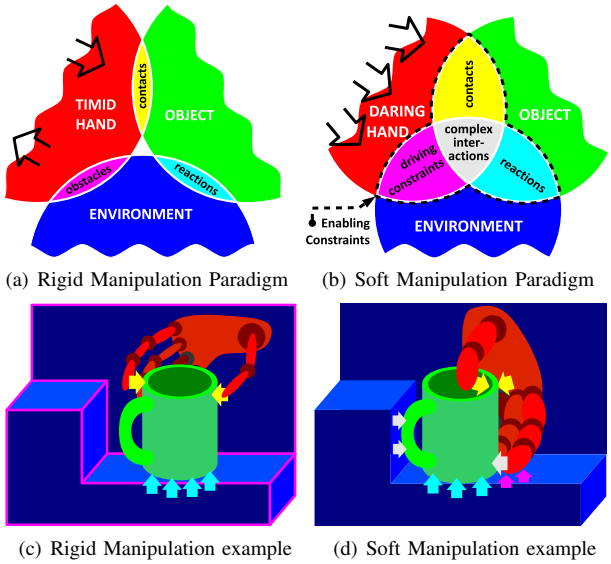
***Abstract*— Despite some prematurely optimistic claims, the ability of robots to grasp general objects in unstructured environments still remains far behind that of humans. This is not solely caused by differences in the mechanics of hands: indeed, we show that human use of a simple robot hand (the Pisa/IIT SoftHand) can afford capabilities that are comparable to natural grasping. It is through the observation of such human-directed robot hand operations that we realized how fundamental in everyday grasping and manipulation is the role of hand compliance, which is used to adapt to the shape of surrounding objects. Objects and environmental constraints are in turn used to functionally shape the hand, going beyond its nominal kinematic limits by exploiting structural softness.**

In this paper, we set out to study grasp planning for hands that are simple - in the sense of low number of actuated degrees of freedom (one for the Pisa/IIT SoftHand) - but are soft, i.e. continuously deformable in an infinity of possible shapes through interaction with objects. After general considerations on the change of paradigm in grasp planning that this setting brings about with respect to classical rigid multi-dof grasp planning, we present a procedure to extract grasp affordances for the Pisa/IIT SoftHand through physically accurate numerical simulations. The selected grasps are then successfully tested in an experimental scenario.

I. Introduction

During the past thirty years, the problem of autonomous robotic grasping has been one of the most widely investigated. For well known scenes and object models, pre-programming of autonomous grasping and manipulation tasks may be an option. Toward this goal, several approaches have been proposed to define the optimal finger placement on the object, either based on some geometric [1] or force [2], [3] grasp quality measures, specifically tailored to convex objects [4], with optimal on-line contact adjustments [5], and also generalized to non-convex objects [6]. Perhaps because of the fragility of the mechanics of most robot hands, a multitude of the planning methods were thought for interactions between the hand and the object that only occurred at the fingertips, limiting contacts with other parts of the hand and avoiding contacts with the rest of the environment at all.

This "timid" approach to manipulation generated by rigidity and fragility of the hand has been recently challenged by the introduction of adaptable, underactuated and/or soft hands. Devices such as the underactuated RobotiQ hand [7], the RBO and RBO 2 hands [8], [9], the iHY hand [10] and the Pisa/IIT SoftHand [11], are designed to be much simpler, and much more robust with respect to the whole interaction process. This allows to use these hands in more "daring" interactions with the objects in the environment, using their full surface for enveloping grasps, and exploiting objects and environmental constraints to functionally shape the hand, going beyond its nominal kinematic limits by exploiting structural softness.



(a) Rigid Manipulation Paradigm (b) Soft Manipulation Paradigm

(c) Rigid Manipulation example (d) Soft Manipulation Example

Fig. 1. Paradigm shift in manipulation, from rigid manipulation (left) to soft manipulation (right). Primary colors identify the scenario main actors: red for the robotic hand, blue for the environment, green for the target object. Secondary colors codify simple interactions between the actors: yellow for hand-object, cyan for object-environment and magenta for environment-hand. Finally, complex interactions, which involve all the three actors at the same time, are white colored. Refer to text for a deeper description.

The differences between a rigid and soft approach to manipulation are sketched in Fig. 1. In the classical paradigm (cfr. panel (a)), the planner searches for suitable points on the object that generate a nominal grasp of good quality, and for trajectories that can bring there the fingertips while avoiding contacts of the hand with the environment. In the example of panel (c), to grasp the green cup while avoiding the wall on the left the planner has to find a path in a narrow passage. However, soft manipulation subverts this scheme (panel (b)). In the example of panel (d), hand-object, object-environment and hand-environment contacts are not avoided but rather sought after and exploited to shape the hand itself around the object.

The set of all possible physical interactions between the hand, the object and the environment, which define the hand-object functional interaction, will be referred to as the set of *Enabling Constraints*. The analysis of such

[1]Research Center "E. Piaggio", Università di Pisa, Largo Lucio Lazzarino 1, 56122 Pisa, Italy
[2]Department of Advanced Robotics, Istituto Italiano di Tecnologia, Via Morego 30, 16163 Genova, Italy
[3]Department of Civil and Industrial Engineering, Università di Pisa, Largo Lucio Lazzarino 1, 56122 Pisa, Italy
[4]Department of Information Engineering, Università di Pisa, Via Caruso, 2, 56122 Pisa, Italy

possibilities constitute a rather new challenge for existing grasping algorithms: adaptation to totally or partially unknown scenes remains a difficult task, toward which only some approaches have been investigated so far. Some of them are model-free and propose geometrical features which indicate good grasps [12], [13], some others evaluate also topological object properties such as holes [14]. Typically, grasps are ranked on a fixed list of suitable hypotheses, do not require supervised learning, but do not adapt over time. Other methods are based on learning the success rate of grasps given some descriptor extracted from sensor data, either evaluated on a real robotic system [15], [16], or on simulated sensor data [17], [18].

Moreover, beside vision-based methods, hand compliance offers the real possibility to use tactile exploration for 3D reconstruction of unknown environments and objects. Tactile sensing can solve some severe limitations of computer vision, such as sensitivity to illumination and limited perspective. As an example, a combined procedure based on dynamic potential fields, that aims at reconstructing 3D object models, which are then used for grasp planning and execution was presented in [19] and recently extended in [20].

In this paper, we consider planning grasps with hands that are simple – in the sense of low number of actuated degrees of freedom, e.g. one for the Pisa/IIT SoftHand – but are soft, i.e. continuously deformable in an infinity of possible shapes through interaction with objects. We present a first approach to the study of this novel kind of manipulation, based on an accurate simulation tool for the SoftHand, developed using the multi-body system software ADAMS [21]. A batch simulation set-up was created and used to perform the automatic creation of a database of grasp affordances for the Pisa/IIT SoftHand and a set of kitchenware objects. The method is related to that followed with simulators such as GraspIt! [22], Open-Rave [23] and OpenGRASP [24], although the softness of the hand introduces a new vista on the problem.

To stress the differences between the rigid and soft manipulation paradigms, Section II presents some of the benefits of using Soft Hands to grasp objects. Section III recalls the mathematical model of the SoftHand after which the simulation tool, presented in Section IV, has been developed. Section V presents the automated simulation batch environment, that led to the results presented in section VI. Finally conclusions are drawn in Section VII.

II. A New Set of Possibilities

By observing the way humans use their real hands, it is possible to realize that, in everyday grasping and manipulation, the role of hand compliance is fundamental. In the first place it is used to adapt to the shape of the hand surroundings: both the target object and the rest of the environment. On the other hand, it is important to notice how the objects and the environment constraints are used, in turn, to functionally shape the hand, going beyond its nominal kinematic limits by exploiting its structural softness.

Although one could ascribe such levels of dexterity to the high levels of sensory-motor capabilities of the human hand itself, it is astounding to compare the performance of the human naked hand with that of a person using a simple robot hand, as the Pisa/IIT SoftHand arm-mounted device shown in Fig. 3.

Fig. 2. A human hand grasping a cup with three different approaches (top panels) and the same grasps reproduced with the Pisa/IIT SoftHand (bottom panels).

Fig. 3. The Pisa/IIT SoftHand mounted on an a human arm.

Thanks to its under-actuated mechanisms the SoftHand is capable to grasp several number of objects by matching to their shape. These combination of simplicity, adaptivity and robustness lets the person experiment in a very natural way with the robotic hand, and soon achieve a level of performance comparable, often similar, to that obtained with their true hands. This achievement is obtained despite the presence of just one degree of actuation on the mechanism and an almost total lack of tactile feedback. Fig. 2, shows three very different ways to grasp a cup, implemented with both the naked hand and the SoftHand.

The SoftHand can substantially match the grasping performance of the human hand thanks to its possibility of exploring and exploiting the *Enabling Constraints* that define, at a very basal level, the problem of grasping and manipulation.

As a further example, consider Fig.s 4, where the combined action of adaptability and robustness allow the user to manipulate and interact with both the environment and the object at the same time, in a complex way (refer also to Fig. 1). Exploiting all the physical constraints that are external with respect to the hand itself: walls, surfaces and edges, force closures of the object between the hand and the environment can be obtained and used to generate simple and effective manipulation tasks, in this case sliding and pivoting

Fig. 4. A person with the arm-mounted SoftHand can seamlessly execute also difficult manipulation tasks which involve combined interactions between hand, object and the environment.

a book.

III. Model of the Pisa/IIT SoftHand

As an example of the new advantages introduced for the soft manipulation paradigm, we investigate how to plan grasps for the Pisa/IIT SoftHand.

Fig. 12 shows that despite the fact the hand is always driven by the same closing command for any object, the final grasps are characterized by different joint configurations, automatically obtained thanks to the adaptability of the hand. The only variables needed to synthesize grasp remain those ones describing the wrist pose. In next sections we will present a randomized investigation method, in order to discover sets of hand/object configurations bringing to the grasp. As follows from previous discussions, the dynamic evolution of the system play a key role, from the pre-grasp phase until a stable grasp is achieved. Hence, starting from the kinematic model of the hand, extensively presented in [11], we developed a dynamic simulation tool in ADAMS. The main equations to be considered in the simulator come from the kineto-static model of the Pisa/IIT hand, which can be described by the following

$$s = Rq, \quad (1)$$

$$\tau = R^T \eta - K_q q. \quad (2)$$

In eq. (1) variable $s \in \mathbb{R}$ describes the displacement of the extremities of the actuation tendon, while $q \in \mathbb{R}^{19}$ is the joint displacement vector, comprising 5 revolute joints for the abduction movement, and 14 soft roll-articular (SR) joints for the flex/extension movement of the fingers. The map between the two variables is the so called *adaptive synergy matrix* $R \in \mathbb{R}^{1\times 19}$. From eq. (1), by kineto-static duality, eq. (2) follows, where $\eta \in \mathbb{R}$ is the motor torque, and $\tau \in \mathbb{R}^{19}$ is torque vector at the joint level. The *joint stiffness matrix* $K_q \in \mathbb{R}^{19\times 19}$ is introduced to properly consider the effect of the joint elastic band on the net torque.

IV. Simulator Implementation

After the CAD models of the parts composing the hand have been imported, a virtual link is placed between the two real ones, Fig. 5(a). Each real part is connected to the virtual one by virtue of a revolute joint, Fig. 5(b). Finally, the coherence between the movements of the simulated SR joint and the real one is assured by introducing a gearwheel-type constraint, Fig. 5(c).

In order to properly describe the behavior of the Pisa/IIT SoftHand, it is important to take also into account: (i) the kinematic constraints imposed by the tendon routing, (ii) the

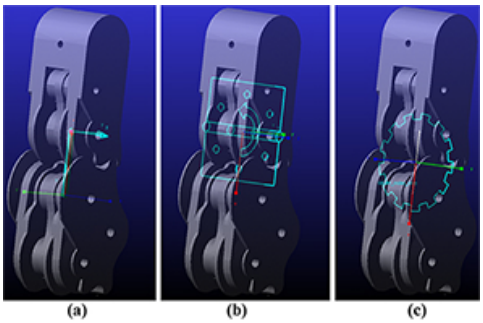


Fig. 5. Implementation of the SR joint in ADAMS: a virtual link is introduced between the two elements (a); a revolute joint connects the virtual link with each real parts (b); a gear-wheel constraint is imposed between the two revolute joints (c).

effects of the elastic bands at the joints since, as follows from (2), both the terms contribute to the net joint torques. The effect of the motor torque was introduced in ADAMS by imposing the desired torque on each joint, considering both the motor torque curve versus time, and the adaptive synergy matrix R. The contribution of the elastic bands is introduced in the model using the ADAMS rotational springs. By properly tuning the damping term in the ADAMS model of springs, the Coulomb friction at the joints is modeled, this has also the the beneficial effects of avoiding unphysical oscillations of the simulation if not present.

Regarding to Fig. 1, the *Enabling Constraints* exploited in the next set of simulations are *contacts* and *reactions*. It means that all contact interactions among object, table and hand geometries are enabled. In order to reduce the set of possible grasp configurations to explore, the inclusion of *driving constraints* is kept as a future work.

V. Batch Simulation Setup

In order to perform a large set of simulations for the Pisa/IIT SoftHand, the ADAMS model was fully parameterized via a template script. Design parameters like object inertia, contact parameters and joint stiffness and friction, were properly chosen to mimic the real hand as closely as possible, they do not need to be modified during the simulation campaign. The sole parameters to be modified are those defining the pose of the hand with respect to the object. To this aim, and to keep the number of simulations reasonably low, without sacrificing the quality of the results, a certain strategy had to be devised.

The first strategy we attempted was an extensive investigation sampling four of the six DoFs necessary to define the position and orientation of the palm frame with respect

to the object frame. More in detail, the position of the origin of the palm frame was parameterized in spherical coordinates (radius, azimuth and elevation). The normal to the palm frame always points toward the center of the sphere which coincides with the origin of the object frame. The remaining DoF is the rotation of the hand frame around the normal. With this strategy a large number of attempts were unsuccessful because, in the starting configuration, the hand was either already interpenetrating the object, or excessively far from it. In the first case, the simulation was skipped, because of the non-feasible condition, while in the latter the object was out of reach or the ejection of the object occurred. The main reason is that a sphere is not a good generalization of many object geometries, such as a pot or a cup.

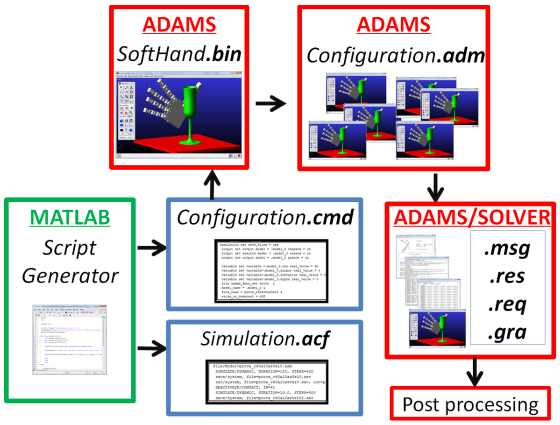


Fig. 6. A flow chart of the use of ADAMS and MATLAB for running batch simulations. In MATLAB, a *.cmd* file is used to define the hand configurations in which attempting the grasp. By loading this file in ADAMS, together with a *.bin* file containing the object/hand model, we can obtain a file (*.adm*) for each configuration. Running the *.acf* file, the *.adm* ones are read, performing the simulations in batch mode. Simulation variables (joint angles, contact forces, etc...) are available in the exit files (*.msg*, *.res*, *.req*, *.gra*) for post-processing operations.

This suggested us to redesign the strategy taking into account the geometry of the object, focusing the attention on the mesh describing the shape of the object.

The mesh of the object to be grasped was imported in MATLAB, and the points of and the normal vectors to the surface were extracted. For 50 randomly selected points of the mesh, the pose of the hand frame was chosen positioning its origin 5 mm (coming from observations of real experiments) outside the surface mesh along the outward normal. The normal to the palm was aligned with the normal to the object at the point, but with opposite direction (the palm always faces the object). Finally, 8 configurations were selected rotating the hand around its normal axis. The 400 palm configurations obtained were put on a test for achieving a stable grasp. Fig. 6 represents a scheme of the flow chart for running batch simulations.

The results achieved after this first investigation step are represented, for the case of the pot, in Fig. 9, where green frames represent successful configurations (stable grasp achieved), red frames the unsuccessful ones. In order to evaluate if a grasp is stably achieved, all the simulations were split into two parts. In the first part the object to be grasped lies on a plane, orthogonal to the gravity vector, in order to hold it on without over-constraining it. No contact was set up between the hand and the table to do not preclude any possible approach direction. This simulation part is three seconds long, that is approximately one second longer than the time of free closure of the hand. Later, starting from the final configuration of the first simulation and keeping active the hand motor torque, a two seconds long simulation is performed removing the table. Afterwards, the velocity of the object is read from the output files, and the grasp is rated as achieved and stable if the velocity is smaller than a tolerance value.

From the results of the first set of simulations, the points that brought to a stable grasp were extracted, and their neighbourhood was further investigated. This research was performed by randomly choosing 10 of the 40 closest points on the mesh, around the successful one. For each new point, again 8 rotations around the palm normal were considered to attempt the grasp.

VI. Simulation Results

The free closure movement of the Pisa/IIT SoftHand obtained with ADAMS is shown in Fig. 7. As explained in Sec. IV, the hand model considers both the tendon routing, eq. 1, and the elastic bands at the joints, eq. 2. The reader can find a deeper analysis about the kineto-static model of the hand in [11].

In our simulation campaign we used everyday objects: a cup, a pot, a colander and a plate, see Fig 8. For the cup example, Fig. 9 shows: a) the points composing the object mesh, b) successful (green) and unsuccessful (red) grasps configurations after the first 400 simulations, c) the result for all tested hand postures and d) all successful configurations.

In order to validate the dynamic behaviour of the simulator and to put the obtained results on a test, some successful hand palm configurations were also implemented with the Pisa/IIT SoftHand prototype. A KUKA lightweight robot was used to exactly replicate the hand/object configuration suggested by the simulation. A comparison snapshot sequence for the pot and the colander is shown in Fig.s 10 and 11.

In TABLES I and II some numerical results of the simulations are summarized. Specifically, in TABLE I, for every object we list: (i) successful postures, the number of palm frame configurations bringing to a stable grasp, (ii) successful points, the number of unique origin positions of the hand frame with possibly multiple orientations, (iii) clouds, number of neighbourhoods investigated, (iv) best cloud, maximum number of successful grasps in a cloud. As we can see, the best result in terms of number of grasp achieved, as well as in terms of individual successful points, is obtained for the pot. However, the best cloud is found for the plate.

In TABLE II, some simple quality indices are listed. In particular, the *cloud quality* (CQ) index reports the highest percentage of stable grasps achieved in the tested clouds. The CQ index is computed as

$$CQ = \frac{g_b}{c_b} 100, \tag{3}$$

where g_b is the number of stable grasp achieved and the c_b is the number of hand palm configurations tested, both considered for the best cloud. The *closure* index (CI) is a measure of how much the hand is closed at the end of the

	successful postures	successful points	clouds	best cloud
Cup	30	13	8	10
Pot	138	41	16	35
Colander	21	6	2	14
Plate	112	29	10	37

TABLE I

Simulation results: (i) number of initial postures leading to a successful grasp, (ii) number of successful individual starting points for the hand placement, (iii) number of clouds studied and (iv) maximum number of successful configurations found in a cloud (best cloud).

	CQ Index %	Closure Index (min-max)%	Net Force (N)
Cup	11.3	40-73	1-57
Pot	39.7	48-68	2-36
Colander	15.9	54-67	11-52
Plate	42.1	57-69	6-28

TABLE II

Simulation results and quality measures. *Cloud Quality* (CQ) index is the percentage of stable grasp achieved, with respect to the number of postures attempted, in the best cloud. *Closure* index (CI) is how much the hand is near to the complete closure configuration, minimum and maximum values found for all the successful grasps are reported. *Net Force* (NF) measures the amount of contact forces exceeding the weight of the object (minimum and maximum).

simulation. In TABLE II the minimum and maximum values found for each object are shown. This index is computed as

$$\text{CI} = \frac{100}{n_{q_c}} \sum_{i=1}^{n_{q_c}} \frac{q_{f_i}}{q_{c_i}}, \tag{4}$$

where q_{f_i} is the final configuration of the i^{th} joint during the grasp, q_{c_i} is the maximum reachable value allowed by the mechanical constraints for the i^{th} joint, and n_{q_c} in the number of *flex-extension joints*.

The *Net Force* (NF) is a measure of the amount of internal forces produced in the grasp by the finger limbs. It is computed as

$$\text{NF} = \left(\sum_{i=1}^{c} f_{c_i}^T f_{c_i} - w_o^T w_o \right)^{\frac{1}{2}}, \tag{5}$$

where c is the number of contact points, f_{c_i} is the force at the i^{th} contact point, and w_o is the weight of the grasped object.

As it is evident from table II, the best *cloud quality* (CQ) index is achieved for the plate. However, also high values of *closure index* are realized when grasping the plate. This result can be explained by taking into account that all the tests were performed with the same torque vs time motor curve. In same cases, especially for the plate, results show that the amount of motor torque is enough to overcome the friction and to bring the hand in a closure configuration. The greatest amount of the NF index, is realized to grasp the colander. The explanation for this result is, again, an excessive level of motor torque, in particular with respect to the low weight of the object. The relevant difference between the minimum and the maximum value of the NF index for all the objects can be explained considering that the palm force is not measured in the simulator. This implies that high values of the NF index correspond to grasps in which the interaction forces are primarily executed by the fingers, low values correspond to grasps that mainly involve the palm.

Generally speaking, the not excellent results of the colander (just 2 clouds and 21 successful configurations) can be explained with the difficulty of randomly selecting points near to the upper edge or the handles. Moreover, the meshes of the objects, obtained from CAD files, are characterized by the presence of (nominally) normal vectors that, indeed, are not orthogonal to the object surface. In some cases, an hand/object interpenetration can occur, caused by a palm normal orientation not orthogonal to the object surface, and the selected point (potentially good) is discarded. For similar reasons, we can consider the cup and the pot as being penalized. Potentially, more points could be have been found on their handle having a better representation of the mesh points and of the normal vectors to the surface.

VII. Conclusion

This paper moves a first step in the direction of studying grasp planning for simple soft hands, imbued with the capabilities of comply with the manipulated object within, and together with, the environment.

Some considerations extracted from the observation of humans to execute simple and more complex tasks, using both their hands and an arm-mounted robot hand, led us to suggest a possible change of paradigm in grasp planning that this setting brings about. After that, we presented a procedure to extract grasp affordances for the Pisa/IIT SoftHand, built upon a physically accurate numerical simulations system that was purportedly implemented. This allowed to select a set of possible grasps that were then successfully tested in an experimental scenario. Despite this early results, much work remains to be done to sharpen the grasp search and to abstract learned grasps between different objects. Although the reported results only scratch the surface of the wholly new problem of soft manipulation planning, however, our results indicate that soft manipulation can be a viable solution for obtaining stable and robust robot grasps.

Acknowledgments

This work is supported by the EC under the CP-IP grant no. 600918 “PaCMan”, within the FP7-ICT-2011-9 program “Cognitive Systems”, ERC Advanced Grant no. 291166 “SoftHands” - A Theory of Soft Synergies for a New Generation of Artificial Hands-, under grant agreements no. 611832 “Walk-Man” and by CONACYT through the scholarship 266745/215873.

References

[1] C. Ferrari and J. Canny, “Planning optimal grasps,” *Proceedings 1992 IEEE International Conference on Robotics and Automation*, pp. 2290–2295, Jan. 1992.

[2] D. Prattichizzo, J. Salisbury, and A. Bicchi, “Contact and grasp robustness measures: Analysis and experiments,” *Experimental Robotics IV*, Jan. 1997.

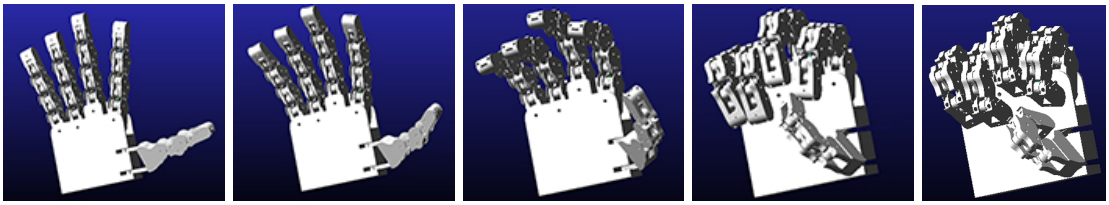
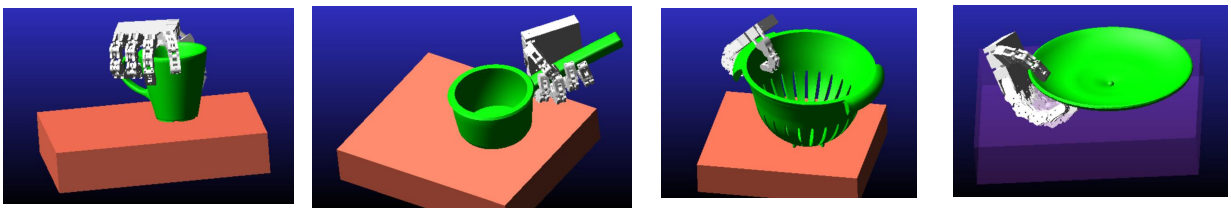
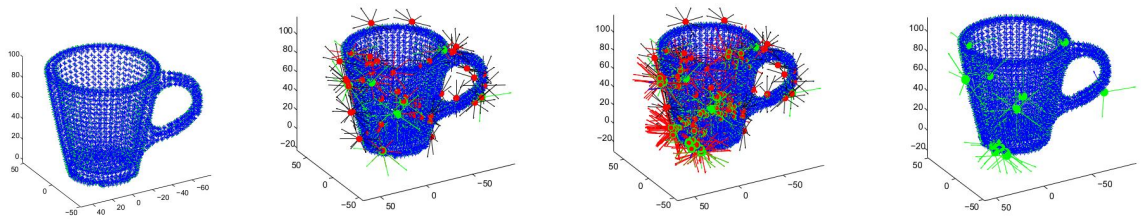


Fig. 7. Sketches of the free closure movement of the Pisa/IIT SoftHand in ADAMS simulation. Since there is neither object nor environment, the movement of the hand is described by the synergistic model of actuation described in eq. (1).



(a) (b) (c) (d)

Fig. 8. Examples of stable grasp achieved with the ADAMS simulations. Using the same control command to close the hand, the final joint configuration is determined by the object shape.



(a) (b) (c) (d)

Fig. 9. MATLAB representation of grasping attempt results for the cup. Red points correspond to palm configuration that did not bring to a stable grasp. Green points correspond to successful hand configurations. Fig. (b) shows the results for the first 50 points, Fig. (c) shows the results for all the points tested. Fig. (d) shows only successful configurations.

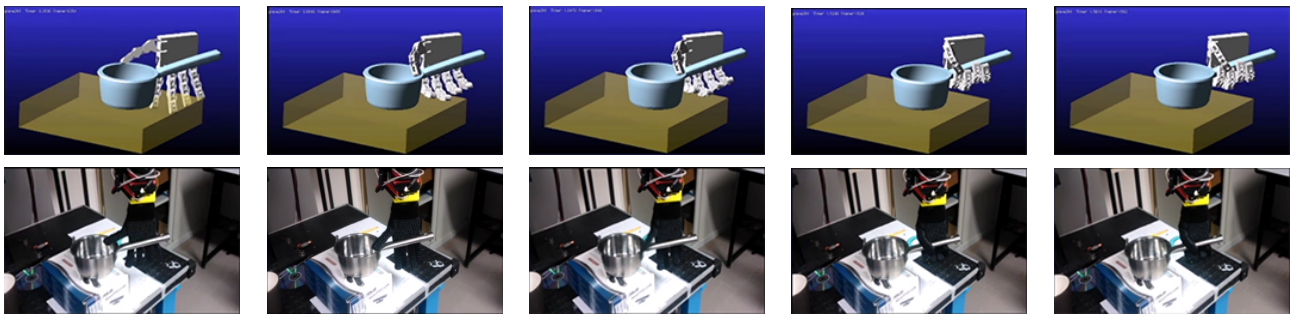


Fig. 10. A snapshot sequence for the pot, comparing the simulation results and the experiment performed with the Pisa/IIT SoftHand prototype.

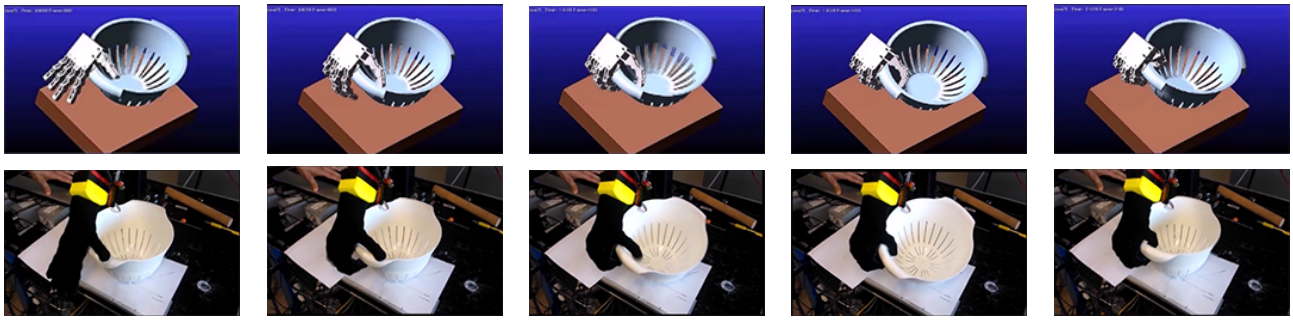


Fig. 11. A snapshot sequence for the colander, comparing the simulation results and the experiment performed with the Pisa/IIT SoftHand prototype.

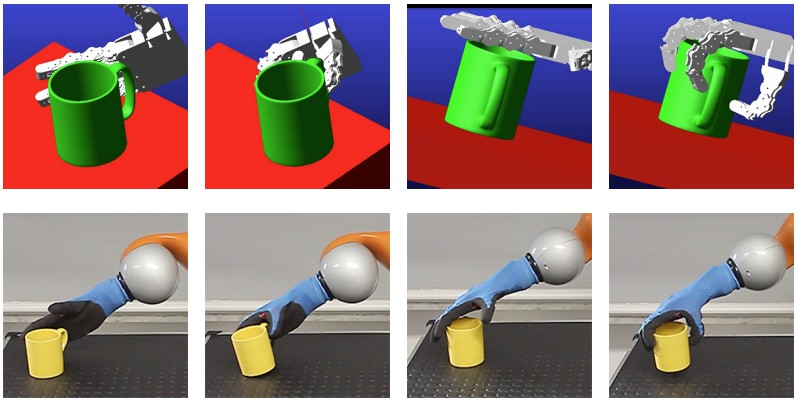


Fig. 12. A possible application of the analysis tool: the top four panels show some grasps that were found in simulation, and the bottom four panels show the same grasps implemented in the SoftHand attached to a KUKA lightweight robot.

[3] C. Rosales, R. Suarez, M. Gabiccini, and A. Bicchi, "On the synthesis of feasible and prehensile robotic grasps," in *Robotics and Automation (ICRA), 2012 IEEE International Conference on*, 2012, pp. 1–7.

[4] M. Teichmann and B. Mishra, "Reactive algorithms for grasping using a modified parallel jaw gripper," in *Robotics and Automation, 1994. Proceedings., 1994 IEEE International Conference on*, 1994, pp. 1–6.

[5] R. Platt, A. H. Fagg, R. A. Grupen, W. Bluethmann, R. Ambrose, M. Diftler, E. Huber, A. Fagg, M. Rosenstein, R. Grupen, C. Breazeal, A. Brooks, A. Lockerd, R. A. Peters, O. C. Jenkins, M. Mataric, and M. Bugajska, "Null-Space Grasp Control: Theory and Experiments," *Robotics, IEEE Transactions on*, no. 2, pp. 1–14, 2010.

[6] D. Wang, B. Watson, and A. Fagg, "A switching control approach to haptic exploration for quality grasps," in *Proceedings of the Workshop on Robot*, Jan. 2007, pp. 1–8.

[7] Robotiq, "Robotiq adaptive gripper: Specification sheet," Jun. 2014. [Online]. Available: `http://robotiq.com/en/`

[8] R. Deimel and O. Brock, "A compliant hand based on a novel pneumatic actuator," in *Robotics and Automation (ICRA), 2013 IEEE International Conference on*. IEEE, 2013, pp. 2047–2053.

[9] ——, "A novel type of compliant, underactuated robotic hand for dexterous grasping," in *Proceedings of Robotics: Science and Systems*, Berkeley, USA, July 2014.

[10] L. U. Odhner, L. P. Jentoft, M. R. Claffee, N. Corson, Y. Tenzer, R. R. Ma, M. Buehler, R. Kohout, R. D. Howe, and A. M. Dollar, "A compliant, underactuated hand for robust manipulation," *The International Journal of Robotics Research*, vol. 33, no. 5, pp. 736–752, 2014.

[11] M. G. Catalano, G. Grioli, E. Farnioli, A. Serio, C. Piazza, and A. Bicchi, "Adaptive synergies for the design and control of the pisa/iit softhand," *International Journal of Robotics Research*, vol. 33, p. 768–782, 2014.

[12] K. Hsiao, S. Chitta, M. Ciocarlie, and E. G. Jones, "Contact-reactive grasping of objects with partial shape information," in *Intelligent Robots and Systems (IROS), 2010 IEEE/RSJ International Conference on*, 2010, pp. 1–8.

[13] E. Klingbeil, D. Rao, B. Carpenter, V. Ganapathi, A. Y. Ng, and O. Khatib, "Grasping with application to an autonomous checkout robot," *2011 IEEE International Conference on Robotics and Automation*, pp. 2837–2844, May 2011.

[14] F. T. Pokorny, J. A. Stork, and D. Kragic, "Grasping objects with holes: A topological approach," *Proceedings - IEEE International Conference on Robotics and Automation*, pp. 1100–1107, 2012.

[15] L. Montesano, M. Lopes, A. Bernardino, J. Santos-Victor, F. S. Melo, and R. Martinez-Cantin, "Learning grasping affordances from local visual descriptors," in *Development and Learning, 2009. ICDL 2009. IEEE 8th International Conference on*, 2009, pp. 1–6.

[16] R. Detry, E. Başeski, M. Popović, and Y. Touati, "Learning continuous grasp affordances by sensorimotor exploration," *From motor learning to*, Jan. 2010.

[17] J. Bohg and D. Kragic, "Learning grasping points with shape context," *Robotics and Autonomous Systems*, vol. 58, no. 4, pp. 362–377, Apr. 2010.

[18] A. Saxena, J. Driemeyer, and A. Y. Ng, "Robotic Grasping of Novel Objects using Vision," *The International Journal of Robotics Research*, vol. 27, no. 2, pp. 157–173, Feb. 2008.

[19] A. Bierbaum and M. Rambow, "Grasp affordances from multi-fingered tactile exploration using dynamic potential fields," *Humanoid Robots*, Jan. 2009.

[20] A. Herzog, P. Pastor, M. Kalakrishnan, L. Righetti, J. Bohg, T. Asfour, and S. Schaal, "Learning of grasp selection based on shape-templates," *Autonomous Robots*, vol. 36, no. 1-2, pp. 51–65, 2013.

[21] M. ADAMS. (2014, Jan.) Msc software last accessed. [Online]. Available: `http://web.mscsoftware.com/`

[22] A. T. Miller, S. Knoop, H. I. Christensen, and P. K. Allen, "Automatic grasp planning using shape primitives," *2003 IEEE International Conference on Robotics and Automation (Cat. No.03CH37422)*, pp. 1824–1829, Jan. 2003.

[23] R. Diankov and J. Kuffner, "Openrave: A planning architecture for autonomous robotics," *Robotics Institute, Pittsburgh, PA, Tech. Rep. CMU-RI-TR-08-34*, pp. 1–18, Jan. 2008.

[24] B. León, S. Ulbrich, R. Diankov, G. Puche, M. Przybylski, A. Morales, T. Asfour, S. Moisio, J. Bohg, J. Kuffner, and R. Dillmann, *OpenGRASP: A Toolkit for Robot Grasping Simulation*. Springer Berlin Heidelberg, Jan. 2010.

A.2 Article: Grasp Planning for the Pisa/IIT Softhand via Bounding-Box Object Decomposition

Authors M. Bonilla, D. Resasco, M. Gabiccini, A. Bicchi

Info Submitted to the 2015 IEEE/RSJ Int. Conf. on Intelligent Robots and Systems

Abstract In this paper, we present a method to plan grasps for soft hands. Considering that soft hands can easily conform to the shape an the object, with preference to certain types of basic geometries and dimensions, we decompose the object into one type of these geometries, particularly into Minimal Volume Bounding Boxes (MVBBs), which are proved to be efficiently graspable by the hand we use. A set of hand poses are then generated using geometric information extracted from such MVBBs. All hand postures are used in a dynamic simulator of the PISA/IIT Soft Hand and put on a test to evaluate if a proposed hand posture leads to a successful grasp. We show, through a set of numerical simulations, that the probability of success of the hand poses generated with the proposed algorithm is very good and represents an evident improvement with respect to our previous results published in A.1.

Relation with the deliverable Part-based grasping under uncertainty for soft hands.

Attachment (following pages until next annex)

Grasp Planning with Soft Hands using Bounding Box Object Decomposition

M. Bonilla[1], D. Resasco[1], M. Gabiccini[1,2,3] and A. Bicchi[1,2]

***Abstract*— In this paper, we present a method to plan grasps for soft hands. Considering that soft hands can easily conform to the shape an the object, with preference to certain types of basic geometries and dimensions, we decompose the object into one type of these geometries, particularly into Minimal Volume Bounding Boxes (MVBBs), which are proved to be efficiently graspable by the hand we use. A set of hand poses are then generated using geometric information extracted from such MVBBs. All hand postures are used in a dynamic simulator of the PISA/IIT Soft Hand and put on a test to evaluate if a proposed hand posture leads to a successful grasp. We show, through a set of numerical simulations, that the probability of success of the hand poses generated with the proposed algorithm is very good and represents an evident improvement with respect to our previous results published in [1].**

I. Introduction

The increase of robot capabilities to actively execute tasks and modify surrounding scenarios, thereby reaching versatile goals, is tightly linked to the ability to generate stable grasps for objects that are even unknown to the robot. A key tool to this ability is the faculty of selecting a successful grasp without the need of identifying *a priori* an object, but only based on its composition of shape primitives with known associated grasps, i.e. resorting to *part-based grasps*.

To this sake, assuming that distorted and/or scattered clouds of 3D points of an object are given — here we assume the segmentation algorithm robust enough to distinguish the manipulandum from the environment — two basic ingredients are needed: (i) a suitable algorithm enabling the representation of these points as a set of shape primitives (e.g., boxes, spheres or cylinders), and (ii) a strategy that takes advantage of the simple primitive shape representation of parts to associate grasps to them.

This approach is well entrenched in the grasp planning community, as testified by [2], [3] and [4]. A not nearly complete classification of the vast literature on this topic can be attempted on the basis of the types of primitive shapes employed for object decomposition. Many bottom-up approaches — starting from point clouds they synthesize object part shapes — use superquadrics (SQs). SQs are parametrizable models that offer a large variety of different shapes. However, to approximate object parts, only superellipsoids out of the groups of SQs are employed in practice, as only these represent closed shapes. In these group we can recall [5], [6] and [7], not mention only a few. However, the more complex the shape is, there higher the number of SQs that have to be instantiated to accurately represent the different parts, and the immense parametrization capabilities lead to highly inefficient algorithms. To overcome these difficulties, beside the *region-growing* (bottom-up) strategy which proved to be not very effective in practice [7], the *split-and-merge* (split: top-down, merge: bottom-up) technique was proposed. Even if this technique allowed to better cope with irregular and unorganized data, the high sensitivity to noise and outliers of SQs approximations, make their wide use still impractical.

Fig. 1. The main idea of this paper is to find different hand poses with high probability of being successful when grasping objects

Following a mid-level solution, according to a purely top-down strategy, a *fit-and-split* algorithm based on Minimum Volume Bounding Box (MVBB) algorithm to object decomposition for grasp planning was originally proposed in [8]. The method consists on iteratively build MVBB of points resulting from splitting the point cloud of the object. The split procedure is performed in such a way that the sum of the two areas resulting from the convex region of each set of points is minimized. An algorithm for grasp planning for fully actuated hands using MVBB decomposition of object was proposed in [9].

In this paper, we adopt a modified version of the MVBB algorithm for object decomposition in [8], as it presents the following properties: (i) it is very efficient and can accomodate scattered 3D points delivered by arbitrary 3D sensors; (ii) the outcome constellation of boxes is quite insensitive to noise. However, with respect to [9] and [10], where 2D grasp hypotheses are made and evaluated on point projections onto box faces and assume, from the outset, the use of rigid and fully-actuated robotic hands or grippers, in

[1]Research Center "E. Piaggio", Università di Pisa, Largo Lucio Lazzarino 1, 56122 Pisa, Italy

[2]Department of Advanced Robotics, Istituto Italiano di Tecnologia, Via Morego 30, 16163 Genova, Italy

[3]Dipartimento di Ingegneria Civile ed Industriale, Largo Lucio Lazzarino 1, 56122 Pisa, Italy

Preprint submitted to 2015 IEEE/RSJ International Conference on Intelligent Robots and Systems. Received March 4, 2015.

the present work we propose a shift of prospective in the creation of grasp hypotheses as we employ a compliant and adaptive hand, the Pisa/IIT SoftHand.

New developments in robotic hands [11], [12] have shown that the correct combination of underactuation and embedded compliance is key to have robotic hands with dexterity comparable to the human hand. In practice, thanks to their mechanism, these hands adapt to the shape of the object while executing a grasp. This behavior brings about a shift of paradigm in grasp planning [1], since old algorithms used to find points or regions in the object to establish a contact can not be directly applied. Considering the mentioned behavior, a rough approximation of the geometry of the object via decomposition into bounding boxes *is enough* to generate candidate hand poses to grasp an object. The possibility to rely on the adaptive behavior of the hand embedded in its the mechanical design, alleviates the task of planning the exact finger placement on the object, and can easily absorb shape primitives approximations due to noisy data, as well as limitations in the representation itself. The framework allows for an easier handling of approximation inaccuracies through haptic feedback and grasp controllers for online corrections.

With a reduced burden to plan detailed finger placements, the box set representation of an object, also ease the mapping of complex actions to box and/or box distribution properties. For example, as also mentioned in [10], in order to *pick-up* an object and place it somewhere, it may intuitively be a good option to grasp the *largest box*. Instead, in order to show the same object to a camera to gather more views, it may be preferable to grasp the object from the *outermost box*, and or, from a box that allows a pincher grasp, so to minimize occlusions caused by hand parts.

In this paper, we recall the method to decompose objects into MVBB presented in [8]. Then, considering the adaptability of soft hands, we present a strategy to propose grasp postures to grasp a set of kitchenware objects [13] previously decomposed into MVBBs. The performances of the method are compared with those presented in [1] and, through simulations performed with the MBS software ADAMS™ [14], we show that with the strategy presented in this paper we highly increase the successful rate of grasps for the same objects.

A. Organization

Section II recalls the algorithm used to decompose the object into MVBB. In Section III, the method developed in this paper to generate hand postures to grasp an object is presented. In Section IV, we present the analysis of the simulations performed with a set of kitchenware objects and, finally, Section V presents the conclusions.

II. Bounding Box Decomposition

Algorithm 1 was presented in [9]. The idea is to decompose the object in MVBBs minimizing the volume of the boxes which fit partial point clouds. The algorithm takes a point cloud of an object ($points^{3D}$) and approximates it with a MVBB. The points are then projected onto three planes which are the non-opposite faces of the box. Then, using Algorithm 2, the points are split in two sets. The split is performed for each of the projected points ($faces$), and for each of the two projection axes. After that, the points are approximated with a box and its area is computed. At the end, the algorithm returns the point and the direction minimizing the mentioned area. The split is then preformed for the set of all 3D points: it results in two boxes with a set of 3D points. The reduction rate of the volume of the two new boxes compared with the original is then compared with a user-given parameter t to judge the split useful or not. In case it is found useful, the split is performed and each of the boxes are consider as new point clouds to repeat the procedure on; the algorithm is stopped otherwise.

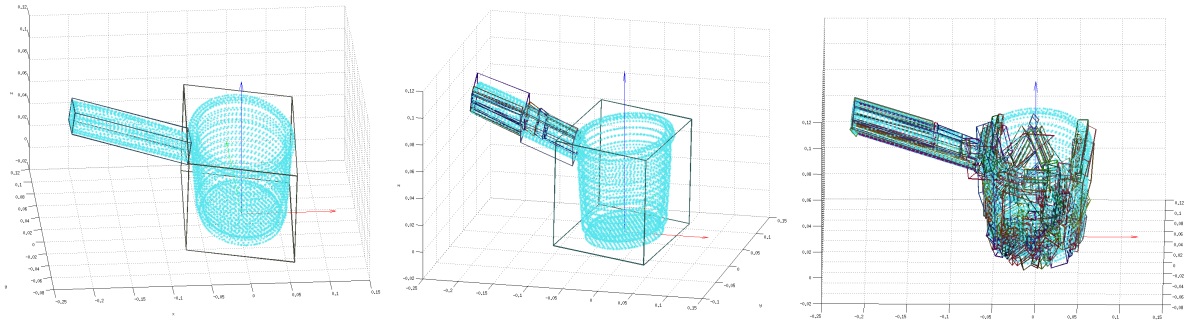
Algorithm 1 Approximate the object in MVBB

```
1: procedure BoxApproximate(faces, points^3D)
2:     box ← FindBoundingBox(points^3D)
3:     faces ← nonOppositeFaces(box)
4:     (p, q) ← split(FindBestSplit(faces, points^3D))
5:     if (percentualVolume(p + q, box) < t then   ▷ t is a stop criteria
6:         BoxApproximate(p)
7:         BoxApproximate(q)
8:     end if
9: end procedure
```

Algorithm 2 Split the point cloud

```
1: procedure FindBestSplit(faces, points^3D)
2:     for i ← 1 to 3 do
3:         p^2D ← projects(points^3D, faces[i])
4:         for x ← 1 to width(faces[i]) do
5:             (p1, p2) ← verticalSplit(p^2D, x)
6:             a1 ← boundArea^2D(p1)
7:             a2 ← boundArea^2D(p2)
8:             if (a1 + a2 < minArea) then
9:                 minArea ← (a1 + a2)
10:                bestSplit ← (i, x)
11:            end if
12:        end for
13:        for x ← 1 to height(faces[i]) do
14:            (p1, p2) ← verticalSplit(p^2D, y)
15:            a1 ← boundArea^2D(p1)
16:            a2 ← boundArea^2D(p2)
17:            if (a1 + a2 < minArea) then
18:                minArea ← (a1 + a2)
19:                bestSplit ← (i, y)
20:            end if
21:        end for
22:    end for
23: end procedure
```

Fig. 2 shows a comparison of different values of t. Depending on the task that we want to perform, this parameter can assume different values. For example, if we want to



(a) $t = 0.0001$ (b) $t = 0.00001$ (c) $t = 0.00000025$

Fig. 2. Comparison of the MVBB generated by the algorithm 1 using different values of t.

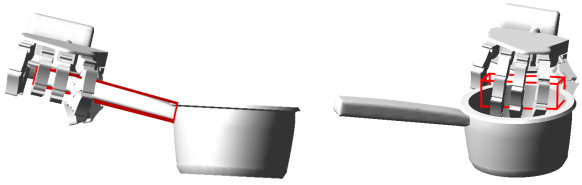
grasp objects from handles, like in cups or pots, they can be isolated using a small value of t, see Fig. 2(a). Similar values can be used to grasp a cup from above for example. On the other hand, if we want to explore more deeply the geometry of the object to, for example, being able to grasp edges in pinch grasp configurations, the parameter t must be increased, see Fig. 2(b) and Fig. 2(c).

In practice, the selection of the parameter t is related to the translation of high level task specifications to low level grasp actions. Let us consider an example in which a robot has to pick up a pot to pour the content into another glass. In this case, a convenient choice is to grasp the pot from the handle, as shown in Fig 3(a). Therefore, in this case a very fine object decomposition is not necessary. On the other hand, if we consider the task passing an object from one hand to another in a bi-manual manipulation setting, the selected box to be grasped could be one in the border of the cooker body, see Fig. 3(b), such that the second arm has more options to decide where to grasp the object without collisions with the first hand. Thus, in this case, a finer object decomposition is beneficial.

III. Proposing Grasp Poses

The aim of this section is to explain how we align the hand with respect to the object in order to grasp it. We consider the orientation of each of the MVBBs and the orientation of the object itself. The orientation of the boxes come from the principal axis defined by the Principal Component Analysis (PCA) performed inside the $FindBoundingBox$ function. The inclusion of the PCA is one of the differences with respect to the original algorithm in [9], and makes the algorithm invariant to the reference frame of the point cloud. In the other hand, the frame attached to the point cloud of the object is defined always with the z axis in the normal direction of its base. Then, the x axis is oriented to some features of interest (a handle, for example) and parallel to the plane of the object base. The origin is placed always in the intersection of the middle axis of the object and the base plane. Once the object is decomposed into MVBBs, the next step is to select a box to grasp. There are many criteria to do this, the most promising and useful depending on the task that the robot has to perform once the object is grasped. In this paper, we start generating hand poses from the outermost box. This choice is driven by our first priority of just grasping the object in a successful manner — most probably in a power grasp configuration, as the hand is just closed to a certain extend — for, e.g., clearing a table. Once a MVBB is selected, the procedure followed to find the transformation T_O^H describing the pose of the hand with respect to the object is the following:

1) Align the x axis of the hand parallel the longest side of the MVBB.
2) Align the z axis of the hand with the axis of the box which has the smallest angle with respect to the z axis



(a) (b)

Fig. 3. The selection of the box to grasp depend on the high level task specification.

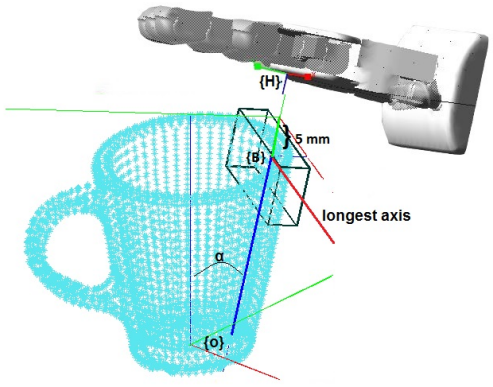


Fig. 4. Graphical explanation of the procedure performed to align the hand with each bounding box.

Preprint submitted to 2015 IEEE/RSJ International Conference
on Intelligent Robots and Systems. Received March 4, 2015.

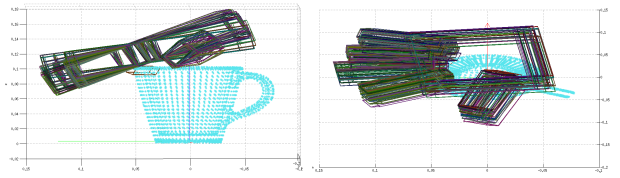


Fig. 5. One example of the free collision hand poses generated for one of the box of the cup.

Object	Old method (%)	MVBB method (%)
Colander	5.25	61.66
Cup	7.5	52.5
Plate	28	80
Pot	34.5	97.5
Average	**18.81**	**77.61**

TABLE I

COMPARISON OF THE SUCCESSFUL GRASP PERCENTAGE OF THE METHOD PRESENTED IN [1] AND THE PORPOSED IN THIS PAPER

of the hand.

3) Compute the orientation of the y axis to form a right-handed frame.

From this procedure, we can generate the rotation matrix R_O^H defining the orientation of the hand frame H with respect to the object frame O. The frame H is placed 5 mm out of the MVBB, in the negative direction of the z axis defined previously. The procedure is explained graphically in Fig. 4.

From the previous steps we can see that, from steps 1) and 2), the selected axis can be the same. In this case, we still align the x axis with the longest side of the MVBB. Instead, the z axis is aligned with the axis of the box which has the smallest angle with respect to the vector connecting the centroid of the MVBB with the object centroid.

A. Pose Variations

The previous procedure generates just a single hand configuration. However, once a MVBB is generated, there is a large number of possibilities to grasp it.

In order to generate more variations for a box, we first set the range of motion in which we can move the hand, translating a distance x_t and rotating by an angle α_t, both along the longest axis of the box, while still not colliding with the object. Fig. 6 shows the random variations created for the cup. Variables x_t and α_t generates a 2D space, with high probability of being fee collision, from where we pick a random point, with uniform distribution, and the check for collision. If this configuration is collision free, then it is a candidate pose to grasp the object. In this work, we generate 40 random configurations for each box and considered the first 5 boxes on the object, thus for each object there are 200 candidate poses.

This procedure constitutes one of the differences with respect to the method proposed in [1], where the authors did not consider collisions in the procedure previous to the simulations. As they explained in the paper, they had a high percentage of failures just because from the very beginning of the simulations, there were many collisions with the object.

In this work, we used a simple collision detector. The hand was approximated using 6 boxes, one for the palm and one for each of the fingers. In order to detect a collision we check if each one of the points constituting the point cloud is inside the boxes of the hand. One example of collision-free hand poses generated for the cup can be observed in Fig. 5.

IV. SIMULATIONS

In this section, we present the simulations preformed for the set of kitchenware objects presented in Fig. 7, which are extracted from the PaCMan Grasp Dataset available at [13]. This figure shows: a) the mesh of the object used for simulations and for MVBB generation, b) the first 2 MVBBs generated by Algorithm 1, c) a finer MVBB generation, d) the first MVBB selected for the method, and finally d) show the first hand pose suggested by the method proposed in this paper.

Table I reports the comparison of the percentage of successful grasp generated with the method presented in this paper and the one presented in [1]. The results are improved mainly because of two reasons, 1) simulations never fail because of collision of the hand and the object at the beginning of the simulations, and 2) the MVBB decomposition helps to better place the hand with respect to the object, also giving more informed options about where and how the variations must be generated.

In order to go through all the process we take the example of the kettle. This is an object that, in practice, because of the dimensions of the object itself and those of the hand, can be grasped just either from the handle or the spout. The first step is to decompose the object in MVBBs. As it can be observed in Fig. 9(b) the algorithm presented in Section II segments the handle and the spout. The next step is to select a box to grasp: Fig. 9(c) shows the selected box in red — this is the outermost one. Then, the collision free hand poses are generated for the selected poses, see Fig. 9(d). Finally, the simulations are performed. Fig 9(e) shows the

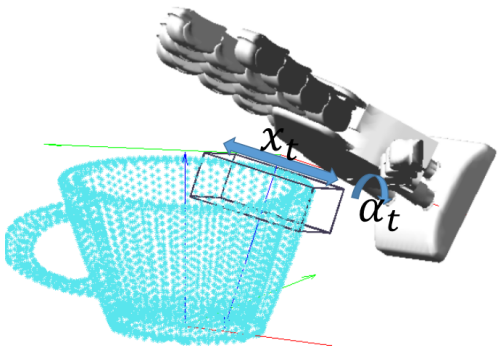
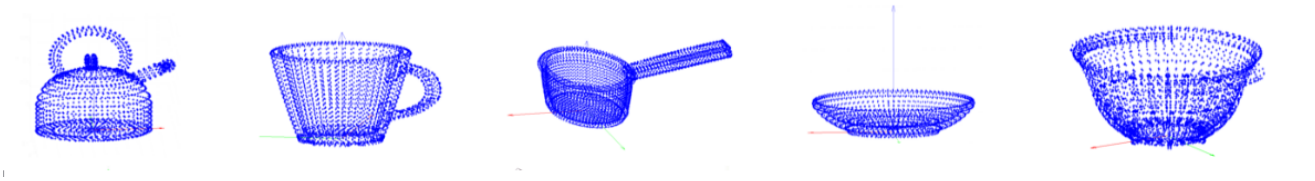
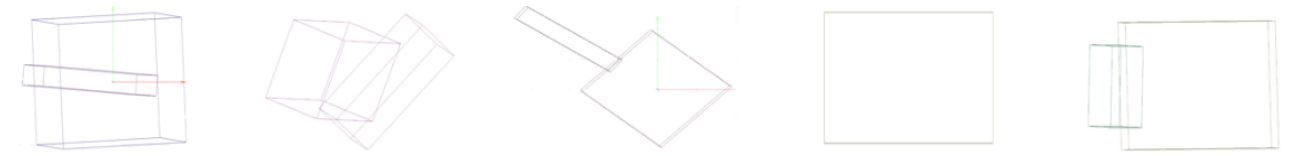


Fig. 6. In order to generate more poses to grasp each box, the hand is rotated and translated along x axis of the box.

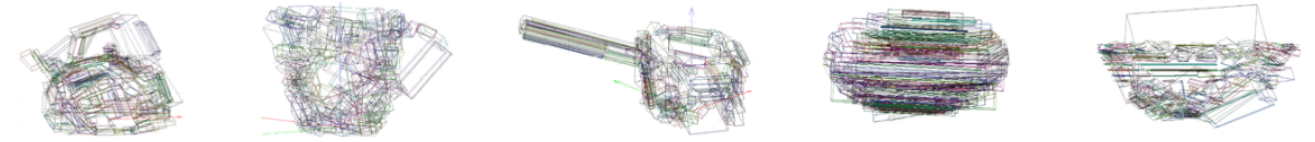
Preprint submitted to 2015 IEEE/RSJ International Conference on Intelligent Robots and Systems. Received March 4, 2015.



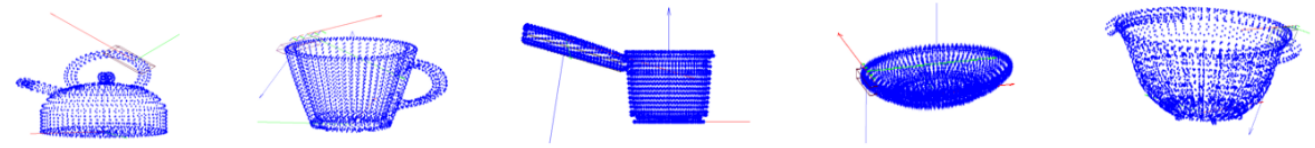
(a) Original point clouds of the objects



(b) Fist MVBB generated



(c) Fine MVBB generation using $t = 0.0001$



(d) First MVBB to grasp suggested for the MVBB method

(e) Simulation if the first grasp suggested by the MVBB method

Fig. 7. Set of objects used for simulations

final configuration of the hand in of one of the simulations where the box-guided grasp suggestion results in the hand grasping the kettle from the handle.

A. *Observations*

Form the simulations performed in this work we observe that most of the failed simulations were caused by the hand being fixed during the whole grasping procedure. Observing how humans grasp an object, we can affirm that they normally adapt the position of the hand with respect to the object to avoid the object being ejected. This suggests to implement a strategy to try to reproduce the same behavior. This could be implemented observing the evolution of the contact forces as the object moves.

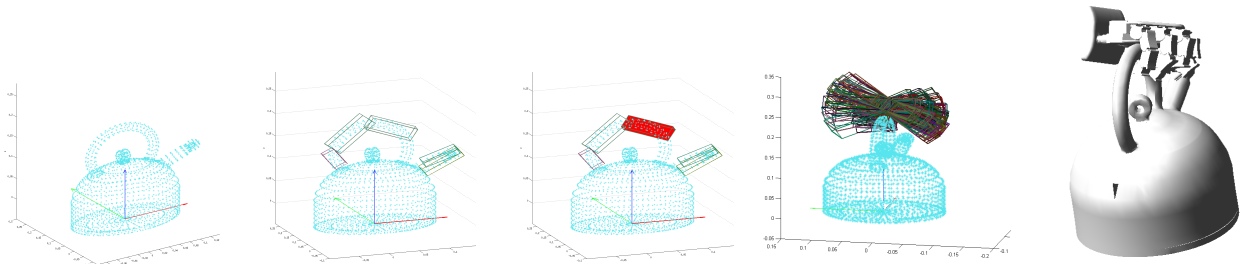
Another point to take into account is how strongly to close the hand to grasp an object. In some simulations, as an example when grasping the plate, the hand closes too much. Therefore, even when the grasp is successful the grasp is still not human-like, as the fingers are too wrinkled. See Fig. 10. A strategy to load the motor current in simulations is envisioned for future research.

Fig. 8 shows the x_t vs α_t space. Green points represent the successful grasp and red points the unsuccessful ones for experiment with the kettle. This plot suggest that a second phase of the procedure can be included to bias new configurations to the regions where there exist more successful grasp. The size of this region can be considered also as a measurement of the grasp quality and robustness of the box.

V. Conclusions

Inspired by the new developments in robotic hands, in this paper we presented an algorithm to propose hand poses for grasp a objects. The algorithm consists in, firstly approxi-

Preprint submitted to 2015 IEEE/RSJ International Conference
on Intelligent Robots and Systems. Received March 4, 2015.



(a) (b) (c) (d) (e)

Fig. 9. Grasp planning using the method presented in this paper was executed for the kettle.

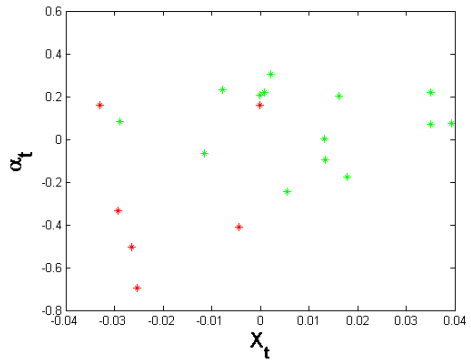


Fig. 8. 2D Space representing all possible variations for a box. Green points are successful grasps while red unsuccessful.

mating the object with a set of MVBB. Then, as the main contribution of this paper, we present a method to select a MVBB in the object and then align the hand. Performing dynamic simulations of the PISA/IIT soft hand we register, with the proposed algorithm, an increase in the probability of success of a proposed pose from 18.1% to 77.61% with respect to the method presented in [1].

Applying the algorithm presented in this paper to a point cloud coming from a 3D sensor and trying to generate hand postures in real-time is left for near future work.

ACKNOWLEDGMENTS

This work is supported by the EC under the CP-IP grant no. 600918 “PaCMan”, within the FP7-ICT-2011-9 program “Cognitive Systems”, ERC Advanced Grant no. 291166 “SoftHands” - A Theory of Soft Synergies for a New Generation of Artificial Hands-, under grant agreements no.611832 “Walk-Man” and by CONACYT through the scholarship 266745/215873.

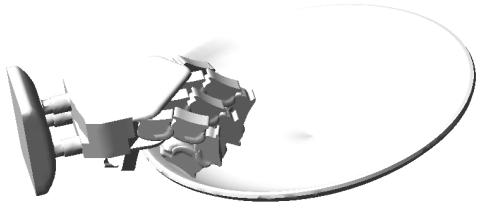


Fig. 10. This figure shows non human like but successful grasp. This problem is generated because the hand is always closed to the maximum.

REFERENCES

[1] M. Bonilla, E. Farnioli, C. Piazza, M. G. Catalano, G. Grioli, M. Garabini, M. Gabiccini, and A. Bicchi, “Grasping with soft hands,” in *International Conference on Humanoid Robots IEEE-RAS 2014*, Madrid, Spain, November 18 - 20, In Press.

[2] K. Shimoga, “Robot grasp synthesis algorithms: A survey,” vol. 15, no. 3, pp. 230–266, 1996.

[3] A. T. Miller, S. Knoop, H. Christensen, and P. Allen, “Automatic grasp planning using shape primitives,” 2003.

[4] C. Borst, M. Fischer, and G. Hirzinger, “Grasp planning: How to choose a suitable task wrench space,” 2004.

[5] C. Goldfeder, P. K. Allen, C. Lackner, and R. Pelossof, “Grasp planning via decomposition trees,” 2007.

[6] G. Biegelbauer and M. Vincze, “Efficient 3d object detection by fitting superquadrics to range image data for robots object manipulation,” 2007.

[7] L. Chevalier, F. Jaillet, and A. Baskurt, “Segmentation and superquadric modeling of 3d objects,” *Journal of WSCG*, vol. 11, no. 1, pp. 1–8, 2003.

[8] K. Huebner, S. Ruthotto, and D. Kragic, “Minimum volume bounding box decomposition for shape approximation in robot grasping,” in *Robotics and Automation, 2008. ICRA 2008. IEEE International Conference on*, May 2008, pp. 1628–1633.

[9] S. Geidenstam, K. Huebner, D. Banksell, and D. Kragic, “Learning of 2D grasping strategies from box-based 3D object approximations,” in *Proceedings of Robotics: Science and Systems*, Seattle, USA, June 2009.

[10] K. Huebner and D. Kragic, “Selection of robot pre-grasps using box-based shape approximation,” in *Intelligent Robots and Systems, 2008. IROS 2008. IEEE/RSJ International Conference on*, Sept 2008, pp. 1765–1770.

[11] M. G. Catalano, G. Grioli, E. Farnioli, A. Serio, C. Piazza, and A. Bicchi, “Adaptive synergies for the design and control of the pisa/iit softhand,” *International Journal of Robotics Research*, vol. 33, p. 768–782, 2014.

[12] R. Deimel and O. Brock, “A novel type of compliant, underactuated robotic hand for dexterous grasping,” in *Proceedings of Robotics: Science and Systems*, Berkeley, USA, July 2014.

[13] “Pacman grasp dataset,” 2015. [Online]. Available: https://github.com/CentroEPiaggio/unipi-grasp-datasets/tree/master/scenario1

[14] MSC Software. (2015) Adams. [Online]. Available: http://web.mscsoftware.com/

A.3 Article: One shot learning and generation of dexterous grasps for novel objects

Authors M. Kopicki, M. Adjible, A. Leonardis, R. Detry, J.L. Wyatt

Info Under review

Abstract This paper presents a method for one-shot learning of dexterous grasps, and grasp generation for novel objects. A model of each grasp type is learned from a single kinesthetic demonstration, and several types are taught. These models are used to select and generate grasps for unfamiliar objects. Both the learning and generation stages use an incomplete point cloud from a depth camera no prior model of object shape is used. The learned model is a product of experts, in which experts are of two types. The first is a contact model and is a density over the pose of a single hand link relative to the local object surface. The second is the hand configuration model and is a density over the whole hand configuration. Grasp generation for an unfamiliar object optimises the product of these two model types, generating thousands of grasp candidates in under 30 seconds. The method is robust to incomplete data at both training and testing stages. When several grasp types are considered the method selects the highest likelihood grasp across all the types. In an experiment, the training set consisted of five different grasps, and the test set of forty-five previously unseen objects. The success rate of the first choice grasp is 84.4% or 77.7% if seven views or a single view of the test object are taken, respectively. A video summarizing the work is available at `https://www.youtube.com/watch?v=FXchUFERErU`.

Relation with the deliverable A new method is proposed for learning grasps and generalize them to novel objects.

Attachment (following pages until next annex)

The International Journal of Robotics Research
–(–):1–24

DOI:–
http://mms.sagepub.com

One shot learning and generation of dexterous grasps for novel objects

Marek Kopicki, Maxime Adjigble, Rustam Stolkin, Ales Leonardis
School of Computer Science, University of Birmingham, Edgbaston, Birmingham, B15 2TT.

Renaud Detry
University of Liège, Belgium

Jeremy L. Wyatt*
CN-CR, University of Birmingham, Edgbaston, Birmingham, B15 2TT and IRI, University Polytechnic de Catalunya, Barcelona

Abstract

This paper presents a method for one-shot learning of dexterous grasps, and grasp generation for novel objects. A model of each grasp type is learned from a single kinesthetic demonstration, and several types are taught. These models are used to select and generate grasps for unfamiliar objects. Both the learning and generation stages use an incomplete point cloud from a depth camera – no prior model of object shape is used. The learned model is a product of experts, in which experts are of two types. The first is a *contact model* and is a density over the pose of a single hand link relative to the local object surface. The second is the *hand configuration model* and is a density over the whole hand configuration. Grasp generation for an unfamiliar object optimises the product of these two model types, generating thousands of grasp candidates in under 30 seconds. The method is robust to incomplete data at both training and testing stages. When several grasp types are considered the method selects the highest likelihood grasp across all the types. In an experiment, the training set consisted of five different grasps, and the test set of forty-five previously unseen objects. The success rate of the first choice grasp is 84.4% or 77.7% if seven views or a single view of the test object are taken, respectively.

Keywords
learning, dexterous grasping

* Corresponding author; e-mail: jlw@cs.bham.ac.uk

Fig. 1. Leftmost image: Objects used, the four objects on the left were used solely for training, the remaining forty three objects on the right were solely used as novel test objects. **Rightmost image:** The Boris manipulation platform on which the experiments reported were carried out.

1. Introduction

Transferring dexterous grasps to novel objects is an open problem. In this paper we present a method that achieves this using as little as one training example per grasp type. The method enables learning and transfer of five grasp examples, including power and pinch-type grasps, to 43 unfamiliar test objects. Grasps generalise to test objects of quite different shape to the training object, for instance from a bowl to a kettle. The approach selects as well as adapts a grasp type from several learned types, simply selecting the type that enables the most similar grasp to training. The method copes with partial and noisy shape information for the test objects, and also generates different grasps for the same object presented in various orientations. The method requires no knowledge of the human defined object category either when learning or performing transfer.

Previous work in learning generalisable grasps falls broadly into two classes. One class of approaches utilises the shape of common object parts or their appearance to generalise grasps across object categories (Saxena et al., 2008; Detry et al., 2013; Herzog et al., 2014; Kroemer et al., 2012). This works well for low DoF hands. Another class of approaches captures the global properties of the hand shape either at the point of grasping, or during the approach (Ben Amor et al., 2012). This global hand shape can additionally be associated with global object shape, allowing generalisation by warping grasps to match warps of global object shape (Hillenbrand & Roa, 2012). This second class works well for high DoF hands, but generalisation is more limited. We achieve the advantages of both classes, generalising grasps across object categories with high DoF hands.

Our main technical innovation to achieve this is to learn two types of models from the example grasp, and then recombine them using a product of experts formulation when inferring a new grasp. Dexterous grasping involves simultaneously satisfying multiple constraints, and our central insight is that a product of experts is a natural way to encode these. Both model types are density functions. The first is a *contact model* of the relation between a rigid link of the hand, and the local object shape near its point of contact. We learn one contact model for each link of the hand involved in the grasp, and these capture local constraints in the grasp. To capture global information we learn a second type of model, a *hand configuration model* from the example grasp. We then use this hand configuration model to constrain the combined search space for the link placements.

Given these two learned models, grasps can be found for novel objects. When presented with a test object, a Monte Carlo procedure is used to combine a contact model with the available point cloud for the new object, to construct a third type of density function (called a *query density*). We build one query density for each hand link, and use them in two ways

to find a grasp for the new object. First we pick a link, and draw a contact point for it on the object from the query density. Then we sample a hand configuration to obtain the remaining link poses via forward kinematics. The whole solution is then refined using local search. The search seeks to maximise the product of experts involving each query density and the hand configuration density.

The paper is organised as follows. We begin with a survey of related work (Sec. 2). We then continue with a description of the representations employed and the learning process (Sec. 3), followed by a description of the process for finding a grasp for a novel object (Sec. 4). We finish with an experimental study (Sec. 5) and a discussion (Sec. 6).

2. Related work

In robotics, grasp planning is driven by two dominant trends. Traditional grasp planning relies on force analysis (FA), which computes the behaviour of an object subject to a grip via the laws of classical mechanics (Bicchi & Kumar, 2000). In recent years, a second trend emerged, whereby a direct mapping from vision to action is either engineered or learned from experience (Bard & Troccaz, 1990; Coelho et al., 2000; Kamon et al., 1996). When comparing one approach to the other, force analysis is the methodical, scrupulous approach, where one attempts to model the physical processes that occur at the interface of the object and the gripper. Given a model of the shape, weight distribution, and surface friction of an object, a model of the shape, kinematics, and applicable forces/torques of a gripper, and a model of object-gripper contacts, force analysis applies the laws of classical mechanics to compute the magnitude of the external disturbances that a grasp can withhold. In turn, from the range of disturbances that a grasp can withhold, authors have defined a number of so-called grasp quality metrics (Shimoga, 1996), amongst which stands the famous *epsilon* measure of Ferrari & Canny (1992). A grasp is called force-closure if external forces can be balanced by the gripper, thereby restraining the object within the hand.

To plan a grasp via force analysis, several problems must be solved. The robot first needs to build a representation of the object. If the object is seen from a single viewpoint, the robot needs either to access to a representation of the complete object shape, or to hypothesise the shape of the occluded side of the object. The robot must also make a fair assessment of the object's mass, mass distribution and friction coefficient (Zheng & Qian, 2005; Shapiro et al., 2004). Then, it attempts to find a set of points on the object's surface that are such that if the gripper's fingers were contacting the object at those points, the grasp would be stable (Shimoga, 1996; Pollard, 2004; Liu, 2000). To compute the forces that are applicable at those points, the robot must rely on a model of hand-object contacts, in part to assess the amplitude of friction forces. The deformation of the object's surface around a contact point is hard to predict. One often assumes hard contacts with a fixed contact area (Bicchi & Kumar, 2000). One also usually assumes static friction between the hand and the object (Shimoga, 1996). Finally, the robot verifies that the grasp is kinematically feasible, i.e., that the hand can be moved to a configuration that realises those contacts (Rosales et al., 2011). Force analysis is applicable to multi-fingered hands, and its ability to generate complex grasps has been shown in the literature (Boutselis et al., 2014; Gori et al., 2014; Grupen, 1991; Hang et al., 2014; Rosales et al., 2012; Saut & Sidobre, 2012; Xu et al., 2007).

Despite its strong theoretical foundation and conceptual elegance, force analysis has not solved robot grasping entirely. FA is difficult to use in an open-ended environment where perceiving and acting are subject to high degrees of noise. Small errors in estimating the pose of an object or in moving the gripper to its intended position can lead to a grasp whose quality substantially differs from the intended one (Zheng & Qian, 2005). This problem can be mitigated by computing *independent contact regions* (ICRs) (Ponce & Faverjon, 1995), i.e., maximal segments of the object's surface where the fingers can be applied while maintaining force closure. Yet, ICRs still suffer from other shortcomings of force analysis, such as difficulties in estimating shape, mass or friction parameters (Rusu et al., 2009).

Research has shown on several occasions that the correlation between grasp quality metrics and real-world grasp outcomes is limited (Bekiroglu et al., 2011; Kim et al., 2013; Goins et al., 2014). In addition to these limitations, FA

remains a computationally-expensive method. These considerations have encouraged researchers to explore different means of planning grasps. As mentioned above, many have begun studying means of building a direct mapping from vision to action, closer in spirit to the way primates establish grasping plans (Jakobson & Goodale, 1991; Hu et al., 1999; Rizzolatti & Luppino, 2001; Fagg & Arbib, 1998; Borra et al., 2011). The mapping captured implicitly by a learning method has merit of its own. It can in some situations be complementary to force analysis, and in other situations be entirely sufficient to perform robot grasping.

Within the class of methods that do not rely on force analysis, a first group plans grasps by searching for shapes that fit within the robot's gripper (Fischinger & Vincze, 2012; Popović et al., 2010; Trobina & Leonardis, 1995; Klingbeil et al., 2011; Richtsfeld & Zillich, 2008; Kootstra et al., 2012; ten Pas & Platt, 2014). Popović et al. (2010) computed grasps onto object edges detected in 2D images, by defining rules such as "*two parallel edges can be grasped by placing two fingers on the outer sides of both edges*". Klingbeil et al. (2011) searched through range data for sites where the PR2's two-finger gripper fits an object, by considering planar sections of the 3D image and identifying U-shaped boundaries that resemble the inside of the PR2 gripper. Such methods work well with simple grippers, but with more complex grippers, the number of rules that need to be hard-coded for the gripper to work well with objects of different sizes and shapes quickly becomes unmanageable. This problem can be overcome by letting the robot learn the mapping from vision to action (Coelho et al., 2000; Kamon et al., 1996; Morales et al., 2004; Platt et al., 2006; Bard & Troccaz, 1990; Detry et al., 2013; Herzog et al., 2014; Kroemer et al., 2012, 2010; Saxena et al., 2008; Zhang et al., 2011; Kim, 2007), instead of hard-coding it. The vision domain of the mapping has been parametrised by features such as SIFT (Saxena et al., 2008), 3D shape primitives (Platt et al., 2006), or 3D object parts (Kroemer et al., 2012; Detry et al., 2013). The action side of the mapping has been parametrised with a 3D grasping point (Saxena et al., 2008), a 6D gripper pose (Herzog et al., 2014) possibly accompanied by a hand pre-shape (Detry et al., 2013), or gripper-object contact points (Ben Amor et al., 2012). In our work, the robot learns a mapping from simple local 3D shape features to a complete parametrisation of the robot hand pose and its fingers.

Grasp learning algorithms can also classified according to the type of input they require. One class of methods focuses on learning a mapping from an image taken from a single viewpoint, to grasp parameters (Bard & Troccaz, 1990; Detry et al., 2013; Herzog et al., 2014; Kroemer et al., 2012, 2010; Saxena et al., 2008; Zhang et al., 2011; Kim, 2007). The image is provided by a depth sensor such as the Kinect, or by a stereo camera, and by nature it covers only one side of the object. Another class assumes the existence of a full 3D model of the object's shape (Hillenbrand & Roa, 2012; Ben Amor et al., 2012). Assuming a complete object model facilitates the planning problem, but it makes perception more challenging, as the robot is required to circle around a novel object before grasping it, and in many cases even then a complete model will not be obtained. Our method is designed to work with a setup that resides between these two classes: grasps are computed from an image captured from a single standpoint, by fixing the camera to the robot's arm and merging several views acquired from various extensions of the arm. We present experiments where the robot uses one to seven images captured from viewpoints spanning up to approximately 200° around the object.

Methods close in spirit to our own include the work of Hillenbrand & Roa (2012), who addressed the problem of transferring a multi-finger grasp between two objects of known 3D shape. A known object's geometry is warped until it matches that of a novel object, thereby also warping grasp points on the surface of the known object onto candidate grasp points on the novel object. Ben Amor et al. (2012) exploit this warping method to transfer grasps taught by a human hand (using a data glove) to contact points for a robot hand on a novel object. We compute a full hand grasping configuration for a novel object, using a grasp model that is learned from a single or a few example grasps. Our method performs best when a nearly complete shape model of the target object is obtainable by sensing, but it is also applicable to partially-modelled objects, based one one view recovering as little as 20% of the object surface in our experiments. One difference in performance compared to the approach of Hillenbrand & Roa (2012) is that they transfer grasps within the same human defined object shape category (e.g. from one mug to another), whereas we are able to transfer grasps to different human defined object categories. Saxena et al. (2008) learned a three-finger grasp success classifier from a bank of photometric

and geometric object features including symmetry, centre of mass and local planarity. Kroemer et al. (2012) and Detry et al. (2013) let the robot learn the pose and pre-shape of the hand with respect to object parts, and relied on compliance or force sensing to close the hand. Alternatively, Kroemer et al. (2010) also relied on control policies to adapt the fingers to the visual input. In our work, the robot plans a set of configurations for each finger individually, using local surface data, then it searches within those configurations for one that complies to hand kinematics. The result is an ability to plan dexterous multi-fingered grasps, while allowing for generalization of grasp models to objects of novel shape. In our earlier work Kopicki et al. (2014) we showed only how to solve the problem of adapting a particular grasp type, given a prior point cloud model of the object. Thus this current work goes beyond our previous work in that we now also: i) present a method to select between adapted grasp types, enabling automatic grasping of a much wider range of objects, and of objects presented in many orientations; ii) present extensive results for learning and testing without prior point clouds; iii) present testing with a variety of numbers of views of the test object.

3. Representations

This section describes the representations underpinning our approach. First we describe the kernel density representation that underpins all the models. The representation of the surface features necessary to encode the contact models follows. Finally we describe the form of the contact model and the hand configuration model. In the rest of the paper we assume that the robot's hand is formed of N_L rigid *links*: a palm, and a number of finger phalanges or links. We denote the set of links $L = \{L_i\}$. The representations are summarised at a high level in a video attached as Extension 2.

3.1. Kernel Density Estimation

Much of our work relies on the probabilistic modelling of surface *features*, extracted from 3D object scans. Features are composed of a 3D position, a 3D orientation, and a 2D local surface descriptor. Let us denote by $SO(3)$ the group of rotations in three dimensions. A feature belongs to the space $SE(3) \times \mathbb{R}^2$, where $SE(3) = \mathbb{R}^3 \times SO(3)$ is the group of 3D *poses* (a 3D position and 3D orientation), and surface descriptors are composed of two real numbers.

This paper makes extensive use of probability density functions (PDFs) defined on $SE(3) \times \mathbb{R}^2$. This section explains how we define these density functions. We represent PDFs non-parametrically with a set of K features (or particles) x_j

$$S = \left\{x_j : x_j \in \mathbb{R}^3 \times SO(3) \times \mathbb{R}^2\right\}_{j\in[1,K]}. \tag{1}$$

The probability density in a region of space is determined by the local density of the particles in that region. The underlying PDF is created through *kernel density estimation* (Silverman, 1986), by assigning a kernel function $\mathcal{K}$ to each particle supporting the density, as

$$\mathbf{pdf}(x) \simeq \sum_{j=1}^{K} w_j \mathcal{K}(x|x_j, \sigma), \tag{2}$$

where $\sigma \in \mathbb{R}^3$ is the kernel bandwidth and $w_j \in \mathbb{R}^+$ is a weight associated to x_j such that $\sum_j w_j = 1$. We use a kernel that factorises into three functions defined on the three components of our domain, namely $\mathbb{R}^3$, $SO(3)$, and $\mathbb{R}^2$. Let us denote the separation of feature x into $p \in \mathbb{R}^3$ for position, a quaternion $q \in SO(3)$ for orientation, $r \in \mathbb{R}^2$ for the surface descriptor. Furthermore, let us denote by μ another feature, and its separation into position, orientation and surface descriptor. Finally,

we denote by σ a triplet of real numbers:

$$x = (p, q, r), \tag{3a}$$
$$\mu = (\mu_p, \mu_q, \mu_r), \tag{3b}$$
$$\sigma = (\sigma_p, \sigma_q, \sigma_r). \tag{3c}$$

We define our kernel as

$$\mathcal{K}(x|\mu, \sigma) = \mathcal{N}_3(p|\mu_p, \sigma_p)\Theta(q|\mu_q, \sigma_q)\mathcal{N}_2(r|\mu_r, \sigma_r) \tag{4}$$

where μ is the kernel mean point, σ is the kernel bandwidth, and where $\mathcal{N}_n$ is an n-variate isotropic Gaussian kernel, and Θ corresponds to a pair of antipodal von Mises-Fisher distributions which form a Gaussian-like distribution on $SO(3)$ (for details see (Fisher, 1953; Sudderth, 2006)). The value of Θ is given by

$$\Theta(q|\mu_q, \sigma_q) = C_4(\sigma_q)\frac{e^{\sigma_q\ \mu_q^T q} + e^{-\sigma_q\ \mu_q^T q}}{2} \tag{5}$$

where $C_4(\sigma_q)$ is a normalising constant, and $\mu_q^T q$ denotes the quaternion dot product.

We note that thanks to the nonparametric representation used above, conditional and marginal probabilities can easily be computed from Eq. (2). The marginal density $\mathbf{pdf}(r)$ is computed as

$$\mathbf{pdf}(r) = \iint \sum_{j=1}^{K} w_j \mathcal{N}_3(p|p_i, \sigma_p)\Theta(q|q_i, \sigma_q)\mathcal{N}_2(r|r_i, \sigma_r)\mathrm{d}p\mathrm{d}q = \sum_{j=1}^{K} w_j \mathcal{N}_2(r|r_j, \sigma_r), \tag{6}$$

where $x_j = (p_j, q_j, r_j)$. The conditional density $\mathbf{pdf}(p, q|r)$ is given by

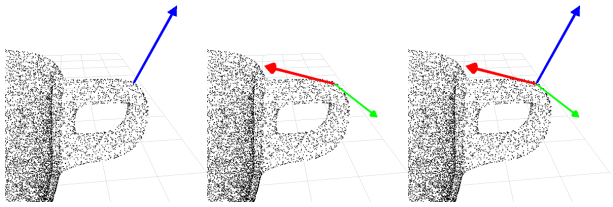
$$\mathbf{pdf}(p, q|r) = \frac{\mathbf{pdf}(p, q, r)}{\mathbf{pdf}(r)} = \frac{\sum_{j=1}^{K} w_j \mathcal{N}_2(r|r_j, \sigma_r)\mathcal{N}_3(p|p_j, \sigma_p)\Theta(q|q_j, \sigma_q)}{\sum_{j=1}^{K} w_j \mathcal{N}_2(r|r_j, \sigma_r)}. \tag{7}$$

3.2. Surface Features

This section explains how the surface features discussed above are acquired from real object data. All objects considered in the paper are represented by point clouds constructed from one or multiple shots taken by a depth camera. A depth camera captures a set of points distributed in a 3D space along the object's visible surface. We directly augment these points with a surface normal and a curvature descriptor. As a result, the point clouds discussed below are composed of points that belong to $SE(3) \times \mathbb{R}^2$. As in the previous section, we denote a point of $SE(3) \times \mathbb{R}^2$ by x, and its separation into position-orientation-curvature components as p, q, and r. For compactness, we also denote the pose of a feature (its position and orientation) as v. As a result, we have

$$x = (v, r), \qquad v = (p, q). \tag{8}$$

The surface normal at p is computed from the nearest neighbours of p using a PCA-based method (e.g. (Kanatani, 2005)). Surface descriptors corresponds to the local *principal curvatures* (Spivak, 1999). The curvature at point p is encoded along two directions that both lie in the plane tangential to the object's surface, i.e., perpendicular to the surface normal at p. The first direction, $k_1 \in \mathbb{R}^3$, is a direction of the highest curvature. The second direction, $k_2 \in \mathbb{R}^3$, is perpendicular to k_1. The curvatures along k_1 and k_2 are denoted by $r_1 \in \mathbb{R}$ and $r_2 \in \mathbb{R}$ respectively, forming a 2-dimensional feature vector $r = (r_1, r_2) \in \mathbb{R}^2$. The surface normals and principal directions allow us to define the 3D orientation q that is associated to a point p. Fig. 2 illustrates a point's surface normal and curvature.



(a) (b) (c)

Fig. 2. An example object point cloud (black dots) with a selected point, its surface normal (blue axis), direction of the first principal curvature k_1 (red axis), and direction of the second principal curvature k_2 (green axis). These form a frame of reference depicted in the rightmost image.

The procedure described above allows the computation of a set of K_O features $\{(v_j, r_j)\}$ from a given object point cloud. In turn, the set of features defines a joint probability distribution, further referred to as the *object model*:

$$O(v, r) \equiv \mathbf{pdf}^O(v, r) \simeq \sum_{j=1}^{K_O} w_j \mathcal{K}(v, r | x_j, \sigma_x) \tag{9}$$

where O is short for $\mathbf{pdf}^O$, $x_j = (v_j, r_j)$, $\mathcal{K}$ is defined in Eq. (4) with bandwidth $\sigma_x = (\sigma_v, \sigma_r)$, and where all weights are equal $w_j = 1/K_O$.

We note that the values computed for surface normals and curvatures are subject to ambiguities. For instance, there are always two ways of defining the directions of vectors k_1 and k_2 given a surface normal: (k_1, k_2) and $(-k_1, -k_2)$. For a sphere or a plane there are an infinite number of orientations about the normal. Finally for a point lying on a near-flat surface, the orientations of k_1 and k_2 within the tangent plane are also uncertain because of sensor noise. We account for these ambiguities/uncertainties at the stage of point cloud processing, by randomly sampling a direction or orientation amongst solutions. In this way, the ambiguity/uncertainty of normals and curvatures is represented by the statistics of the surface features that become the input data to the object model density. We now describe how we model the relationship of a finger link to the surface of the training object.

3.3. Contact Model

A contact model M_i encodes the joint probability distribution of surface features and of the 3D pose of the i-th hand link. Let us consider the hand grasping some given object. The (object) contact model of link L_i is denoted by

$$M_i(U, R) \equiv \mathbf{pdf}_i^M(U, R) \tag{10}$$

where M_i is short for $\mathbf{pdf}_i^M$, R is the random variable modelling surface features, and U models the pose of L_i *relative* to a surface feature. In other words, denoting realisations of R and U by r and u, $M_i(u, r)$ is proportional to the probability of finding L_i at pose u relative to the frame of a nearby object surface feature that exhibits feature vector equal to r.

Given a set of surface features $\{x_j\}_{j=1}^{K_O}$, with $x_j = (v_j, r_j)$ and $v_j = (p_j, q_j)$, a contact model M_i is constructed from features from the object's surface. Surface features close to the link surface are more important than those lying far from the surface. Features are thus weighted, to make their influence on M_i decrease with their squared distance to the i^{th} link (Fig. 4). Additionally, features that are further than a cut-off distance δ_i from L_i are ignored. The weight is given by

$$w_{ij} = \begin{cases} \exp(-\lambda ||p_j - a_{ij}||^2) & \text{if } ||p_j - a_{ij}|| < \delta_i \\ 0 & \text{otherwise,} \end{cases} \tag{11}$$

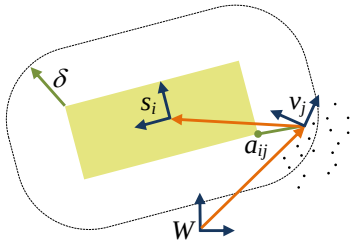
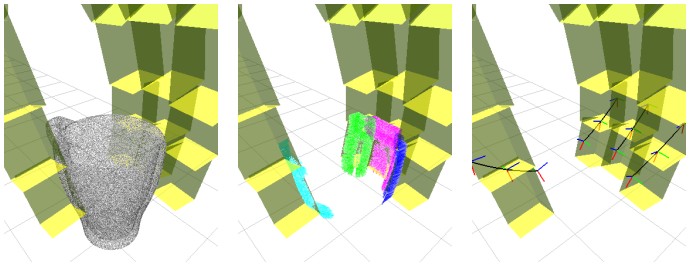


Fig. 3. Contact model. The figure shows the i-th link L_i (in yellow) and its pose s_i. The black dots are features from the surface of an object. The distance a_{ij} between feature v_j and the closest point on the link's surface is shown in green. The rounded rectangle illustrates the cut-off distance δ_i. The poses v_j and s_i are expressed in the world frame W. The top orange arrow illustrates u_{ij}, i.e., the pose of L_i relative to v_j.



(a) (b) (c)

Fig. 4. Example top grasp of a mug represented by a point cloud (a). The dotted coloured regions are rays between features and the closest hand link surfaces (b). The black curves with frames at the fingertips represent the range of hand configurations in Eq. (16) (c).

where $\lambda \in \mathbb{R}^+$ and a_{ij} is the point on the surface of L_i that is closest to p_j.

Let us denote by $u_{ij} = (p_{ij}, q_{ij})$ the pose of L_i relative to the pose v_j of the j^{th} surface feature. In other words, u_{ij} is defined as

$$u_{ij} = v_j^{-1} \circ s_i, \tag{12}$$

where s_i denotes the pose of L_i, $\circ$ denotes the pose composition operator, and v_j^{-1} is the inverse of v_j, with $v_j^{-1} = (-q_j^{-1} p_j, q_j^{-1})$ (see Fig. 3). The contact model is estimated as

$$M_i(u, r) \simeq \frac{1}{Z} \sum_{j=1}^{K_{M_i}} w_{ij} \mathcal{N}_3(p|p_{ij}, \sigma_p) \Theta(q|q_{ij}, \sigma_q) \mathcal{N}_2(r|r_j, \sigma_r) \tag{13}$$

where Z is a normalising constant, $u = (p, q)$, and where $K_{M_i} \leq K_O$ is a number of features which are within cut-off distance δ_i to the surface of link L_i. If the number of features K_{M_i} of contact model M_i is not sufficiently large, contact model M_i is not instantiated and is excluded from any further computation. Consequently, the overall number of contact models N_M is usually smaller than the number of links N_L of the robotic hand. We denote the set of contact models learned from a grasp example g as $\mathcal{M}^g = \{\mathcal{M}_i^g\}$. The contact models are quite different for the different links within a grasp. This can be seen by comparing the marginalised contact models $M(r)$ for two example training grasps and two links in Fig. 5.

The parameters λ and $\sigma_p, \sigma_q, \sigma_r$ were chosen empirically and kept fixed in all experiments reported in Sec. 5. The time complexity for learning each contact model from an example grasp is $\Omega(TK_O)$ where T is the number of triangles in the tri-mesh describing the hand links, and K_O is the number of points in the object model.

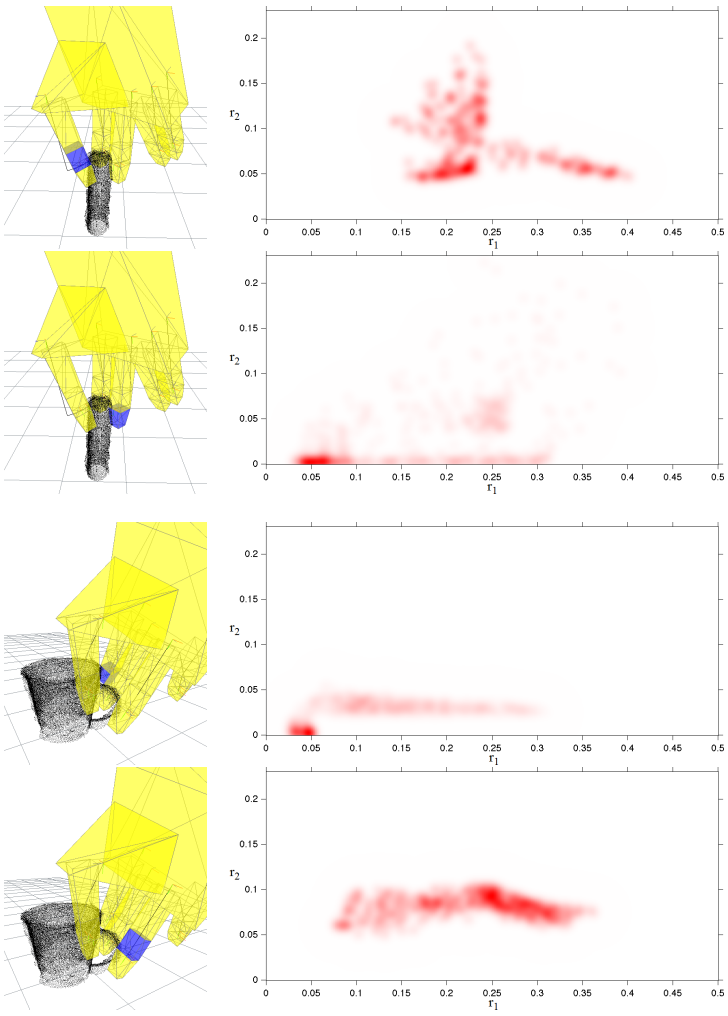


Fig. 5. Illustration of the contact model. Each image from the left column shows an example training grasp and a single selected link in blue colour. The corresponding image from the right column shows a distribution of the curvature descriptor for the features involved in the contact model of the selected link. The top two rows show contact models for two links for a pinch grasp on a vitamin tube, while the bottom two rows show contact models for two links from a handle grasp.

3.4. Hand Configuration Model

The hand configuration model, denoted by C, encodes a set of configurations of the hand joints $h_c \in \mathbb{R}^D$ (i.e., joint angles), that are particular to a grasp example. The purpose of this model is to allow us to restrict the grasp search space (during grasp transfer) to hand configurations that resemble those observed while training the grasp.

In order to boost the generalisation capability of the grasping algorithm the hand configuration model encodes the hand configuration that was observed when grasping the training object, but also a set of configurations recorded during the approach towards the object. Let us denote by h_c^t the joint angles at some small distance *before* the hand reached the training object, and by h_c^g the hand joint angles at the time when the hand made contact with the training object. We consider a set of configurations interpolated between h_c^t and h_c^g, and extrapolated beyond h_c^g, as

$$h_c(\gamma) = (1-\gamma)h_c^g + \gamma h_c^t \tag{14}$$

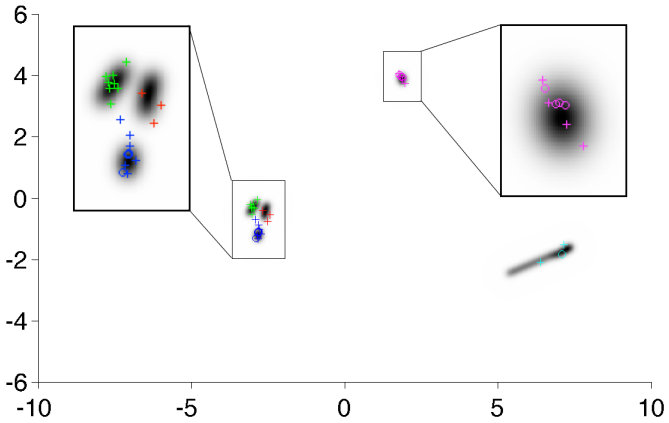


Fig. 6. Configuration models learned from real data (see Sec. 5). A configuration model is a PDF defined on the space of hand joint angles. The 20 degrees of freedom of our robot's hand make it difficult to plot a configuration model. Instead, we applied weighted PCA to the hand data collected during five different training grasps, and plot a KDE of the two principal components. The resulting density is shown in black in the figure. The coloured crosses show the configuration of grasps on test objects computed in Sec. 5 from the learned grasp models. Crosses coloured in red, green, blue, magenta and cyan respectively correspond to transferred grasps of type "handle", "pinch", "pinch with support", "power", and "powertube".

where $\gamma \in \mathbb{R}$. For all $\gamma < 0$, configurations $h_c(\gamma)$ are beyond h_c^g (see Fig. 4). The hand configuration model C is constructed by applying kernel density estimation to

$$\mathcal{H}_c = \{h_c(\gamma) : \gamma \in [-\beta, \beta], \beta \in \mathbb{R}^+\}, \tag{15}$$

as

$$C(h_c) \equiv \sum_{\gamma \in [-\beta,\beta]} w(h_c(\gamma)) \mathcal{N}_D(h_c | h_c(\gamma), \sigma_{h_c}) \tag{16}$$

where $w(h_c(\gamma)) = \exp(-\alpha \|h_c(\gamma) - h_c^g\|^2)$ and $\alpha \in \mathbb{R}^+$. α and β were hand tuned and kept fixed in all the experiments. The hand configuration model computation has time complexity $\Omega(d_h K_C)$ where d_h is the number of dimensions of the configuration vector, and K_C is the size of the set of values of γ used in Eq. (16). Fig. 6 shows a plot of the configuration models learned in our experiments.

4. Inferring Grasps for Novel Objects

After acquiring the contact model and the configuration model, the robot is now presented with a new query object to grasp. The aim is that the robot finds a generalisation of a training grasp such that its links are well-placed with respect to the object surface, while preserving similarity to the example grasp. We infer generalised grasps for every example grasp, and pick the transfer grasp that is most likely according to the learned models.

First of all we combine each of the contact models with the query object's perceived point cloud, to obtain a set of *query densities*, one for each link that has an associated contact model. The i-th query density Q_i is a density modelling where the i-th link can be placed, with respect to the surface of a new object (see Fig. 7). From the query densities, a hand pose is generated as follows. We randomly pick a link i. We randomly sample, from the corresponding query density Q_i, a pose for link i. We sample, from the configuration model C, a hand configuration that is compatible with the pose selected for link i, and then we compute from forward kinematics the 3D poses of all the remaining hand links. We refine the grasp by performing a simulated annealing search in the hand configuration space, to locally maximise the grasp likelihood

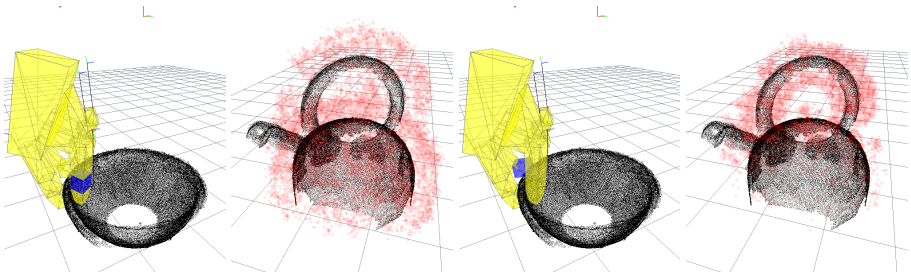


Fig. 7. Visualisation of two query densities (panels 2 and 4 from the left) for two contact models (panels 1 and 3) of a pinch with support grasp. The links to which the contact models are associated are in blue. A query density is a distribution over poses of the corresponding link (red cloud) for a new "query" kettle.

measured as the product of the hand configuration density and the query densities for all the hand links. We repeat the entire process a number of times, and select the most likely grasp that is also kinematically feasible.

The optimisation procedure generates many possible grasps, each with its likelihood. Each grasp has a set of link poses that independently comply with the contact models, while jointly complying with the hand configuration model. The following subsections explain in detail how to estimate query densities for a given query object, and how grasp optimisation is carried out. Extension 2 contains a high level video description of the grasp inference process as described in detail here.

4.1. Query Density

This section explains how query densities are constructed. A query density results from the combination of a contact model for a specific finger link with an object point cloud O for the new object. The purpose of a query density is both to generate and evaluate poses of the corresponding finger link on the new object. The i-th query density Q_i models the pose s *in the world frame* of the i-th link L_i.

A query density should be defined in a way that achieves good generalisation from training to test objects. To achieve this there are three relevant random variables to consider: the random variable V denoting a point on the object's surface, expressed in the world frame; the random variable for surface curvature of such a point R; and the random variable denoting finger link pose U relative to a local frame on the object. We define a joint density over all four variables, $\mathbf{pdf}_i(s,u,v,r)$. The pose distribution of robot link L_i is then defined by marginalisation with respect to u, v and r:

$$\mathbf{pdf}_i(s) = \iiint \mathbf{pdf}_i(s,u,v,r)\mathrm{d}v\mathrm{d}u\mathrm{d}r \tag{17}$$

Since $s = v \circ u$ it is completely determined by v and u. Thus we may factorise Eq. (17) as follows:

$$\mathbf{pdf}_i(s) = \iiint \mathbf{pdf}(s|u,v)\mathbf{pdf}_i(u,v,r)\mathrm{d}v\mathrm{d}u\mathrm{d}r \tag{18}$$

where $\mathbf{pdf}(s|u,v)$ is a Dirac delta function. We can factor Eq. (18) again by assuming that v (the density in the world frame of a surface point) and u (the distribution of the finger link pose relative to its closest surface point) are conditionally independent given r (the local curvature for a surface point):

$$\mathbf{pdf}_i(s) = \iiint \mathbf{pdf}(s|u,v)\mathbf{pdf}_i(u|r)\mathbf{pdf}(v|r)\mathbf{pdf}(r)\mathrm{d}v\mathrm{d}u\mathrm{d}r \tag{19}$$

where $\mathbf{pdf}_i(u|r) = M_i(u|r)$ and $\mathbf{pdf}(v|r) = O(v|r)$, since the object model does not depend on link L_i. The remaining question is how to determine the density over the curvatures r. Clearly to encourage generalisation curvatures should be preferred that are present in both the contact model and the object model, thus we set $\mathbf{pdf}(r) = M_i(r)O(r)$. Thus the i-th query density Q_i can be approximated by a single integral:

Algorithm 1: Pose sampling (M_i, O)

For samples $j = 1$ to K_{Q_i}
  Sample $(\hat{v}_j, \hat{r}_j) \sim O(v, r)$
  Sample from conditional density $(\hat{u}_{ij}) \sim M_i(u|\hat{r}_j)$
  Compute sample weight $w_{ij} = M_i(\hat{r}_j)$
  $\hat{s}_{ij} = \hat{v}_j \circ \hat{u}_{ij}$
  separate $\hat{s}_{ij}$ into position $\hat{p}_{ij}$ and quaternion $\hat{q}_{ij}$
return $\{(\hat{p}_{ij}, \hat{q}_{ij}, w_{ij})\}$, $\forall j$

$$Q_i(s) = \mathbf{pdf}_i(s) = \iiint T(s|u,v) M_i(u|r) O(v|r) M_i(r) O(r) \mathrm{d}v\mathrm{d}u\mathrm{d}r \tag{20a}$$

$$= \iiint T(s|u,v) O(v,r) M_i(u|r) M_i(r) \mathrm{d}v\mathrm{d}u\mathrm{d}r \tag{20b}$$

where $T(s|u,v) \equiv \mathbf{pdf}(s|u,v)$ which is the Dirac delta function mentioned above. Eq. (20) defines the density that must be computed for each link prior to grasp optimisation (Sec. 4.2). This query density (20) can be approximated by K_{Q_i} kernels centred on the set of weighted, sampled finger link poses for link L_i returned by Algorithm 1:

$$Q_i(s) \simeq \sum_{j=1}^{K_{Q_i}} w_{ij} \mathcal{N}_3(p|\hat{p}_{ij}, \sigma_{p_i}) \Theta(q|\hat{q}_{ij}, \sigma_{q_i}) \tag{21}$$

with j-th kernel centre $(\hat{p}_{ij}, \hat{q}_{ij}) = \hat{s}_{ij}$, and where all weights were normalised $\sum_j w_{ij} = 1$. The number of kernels $K_{Q_i} = K_Q$ were chosen equal for all query densities and grasp types (unless otherwise stated), also the bandwidths σ_{p_i} and σ_{q_i} in Eq. (21) were hand tuned and kept fixed in all the experiments. Fig. 7 depicts two example query densities created for two contact models of a handle grasp.

When a test object is presented a set of query densities $\mathcal{Q}^g$ is calculated for each training grasp g. The set $\mathcal{Q}^g = \{Q_i^g\}$ has $N_Q^g = N_M^g$ members, one for each contact model M_i^g in $\mathcal{M}^g$. The computation of each query density has time complexity $\Omega(K_{M_i} K_Q)$ where K_{M_i} is the number of kernels of the i-th contact model density (13), and K_Q is the number of kernels of the corresponding query density.

4.2. Grasp Optimisation and Selection

During testing the robot will have at its disposal N_G grasp types $\mathcal{G} = \{\mathcal{Q}^g, C^g\}$. We now describe how these are used to generate a set of ranked grasps for a new object by Algorithm 2. There is an initial grasp generation phase. This is followed by interleaved grasp optimisation and selection steps.

Grasp Generation A initial set of grasps is generated for each grasp type g by randomly picking a query density Q_i^g and then sampling a pose s_i from it. Together with a sample h_c from C^g this defines a complete hand pose h. A set of initial solutions across all grasp types $\mathcal{H}^1 = \{h_j^g\}$ is generated, where h_j^g means the j^{th} initial solution for grasp type g. To represent each of these grasp solutions let us denote by $s_{1:N_L} = (s_1, ..., s_{N_L})$ the configuration of the hand in terms of a set of hand link poses $s_l \in SE(3)$. Let us also denote by $h = (h_w, h_c)$ the hand pose in terms of a wrist pose $h_w \in SE(3)$

Algorithm 2: Grasp Optimisation and Selection ($\{\mathcal{Q}^g, C^g\}, \forall g, \mathcal{K}_{selection}$)

For each grasp g
  For $j = 1$ to N
    Randomly select a query density Q_i^g from $\mathcal{Q}^g$
    Sample the pose s_i of the i^{th} link from Q_i^g
    Sample a hand configuration h_c from $C^g(h_c)$
    Compute the remaining hand link poses and thus overall hand configuration and pose h_j^g using forward kinematics
  end
end
$\mathcal{H}^1 = \{h_1^1, \ldots, h_j^1, \ldots, h_N^1, h_1^2, \ldots, h_N^2, h_1^3, \ldots, \ldots, h_N^{N_g}\}$
For $k = 1$ to K
  if $k \in \mathcal{K}_{selection}$
    rank $\mathcal{H}^k$ by Eq. 26 and retain top $p\%$
  end
  for $m = 1$ to $|\mathcal{H}^k|$
    $\mathcal{H}_m^k$ = perform a step of simulated annealing on $\mathcal{H}_m^k$ using Eq. 23 as the objective function.
  end
  $\mathcal{H}^{k+1} = \mathcal{H}^k$
end
rank $\mathcal{H}^{K+1}$ by Eq. 26
return $\mathcal{H}^{K+1}$

and joint configuration $h_c \in \mathbb{R}^D$. Finally, let $k^{\text{for}}(\cdot)$ denote the forward kinematic function of the hand, with

$$s_{1:N_L} = k^{\textbf{for}}(h), \quad s_l = k_l^{\text{for}}(h) \tag{22}$$

Having generated an initial solution set $\mathcal{H}^1$ stages of optimisation and selection are interleaved.

Grasp Optimisation Steps The objective of the grasp optimisation steps is, given a candidate grasp and a grasp model g, to find a grasp that maximises the product of the likelihoods of the query densities and the hand configuration density

$$\underset{(h)}{\text{argmax}}\, \mathcal{L}^g(h) = \underset{(h)}{\text{argmax}}\, \mathcal{L}_C^g(h)\mathcal{L}_Q^g(h) = \underset{(h_w,h_c)}{\text{argmax}}\, C^g(h_c) \prod_{Q_i^g \in \mathcal{Q}^g} Q_i^g\left(k_i^{\text{for}}(h_w, h_c)\right) \tag{23}$$

where $\mathcal{L}^g(h)$ is the overall likelihood, where $C^g(h_c)$ is the hand configuration model (16), Q_i^g are query densities (21). Improvement is by simulated annealing (SA) (Kirkpatrick et al., 1983). The SA temperature T is declined linearly from T_1 to T_K over the K steps. In each time step, one step of simulated annealing is applied to every grasp m in $\mathcal{H}^k$.

Grasp Selection Steps During periodic, predetermined selection steps, grasps are ranked and only the most likely $p\%$ retained for further optimisation. During these selection steps the criterion in (23) is augmented with an additional expert $W(h_w, h_c)$ penalising collisions in a soft manner. This penalises grasps which are likely to lead to grasp failure. This soft collision expert has a cost that rises exponentially with the greatest degree of penetration through the object point cloud by any of the hand links. We thus refine Eq. 23:

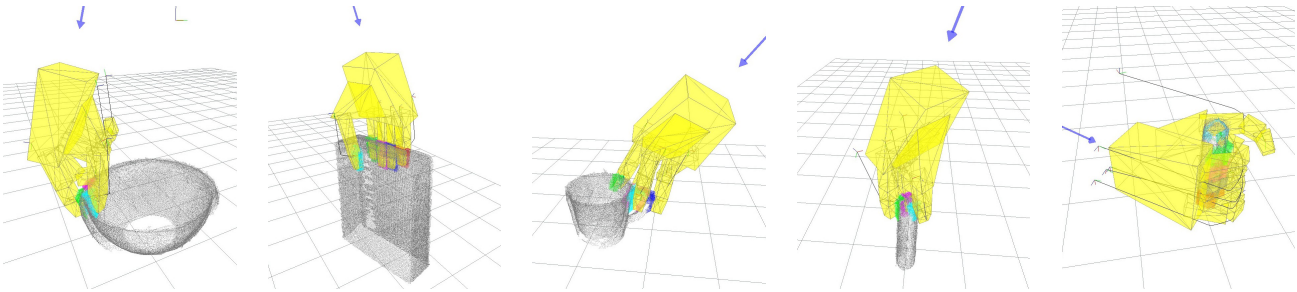


Fig. 8. The five training grasps. From left to right these are *pinch with support*, *power-box*, *handle*, *pinch*, and *power-tube*. The grey lines show the sequence of finger tip poses on the demonstrated approach trajectory. The whole hand configuration is recorded for this whole approach trajectory. The initial pose and configuration we refer to as the pre-grasp position. For learning the contact models and the hand configuration model only the final hand pose (the yellow hand pose) is used. The point clouds are the result of registration of seven views with a wrist mounted depth camera taken during training.

$$\mathcal{L}^g(h) = \mathcal{L}^g_W(h)\mathcal{L}^g_C(h)\mathcal{L}^g_Q(h) \tag{24}$$

$$= W(h_w, h_c)C^g(h_c) \prod_{Q_i \in \mathcal{Q}^g} Q^g_i \left(k^{\text{for}}_i (h_w, h_c)\right) \tag{25}$$

where $\mathcal{L}^g(h)$ is now factorised into three parts, which evaluate the collision, hand configuration and query density experts, all at a given hand pose h. A final refinement of the selection criterion is due to the fact the number of links involved in a grasp varies across grasp types. Thus the number of query densities $N^{g_1}_Q$, $N^{g_2}_Q$ for different grasp models $g_1 \neq g_2$ also varies, and so the values of $\mathcal{L}^{g_1}$ and $\mathcal{L}^{g_2}$ cannot be compared directly. Given the grasp with the maximum number of involved links $N^{\max}_Q$, we therefore normalise the likelihood value (24) with

$$\|\mathcal{L}^g(h)\| = \mathcal{L}^g_W(h)\mathcal{L}^g_C(h) \left(\mathcal{L}^g_Q(h)\right)^{\frac{N^{\max}_Q}{N^g_Q}} . \tag{26}$$

It is this normalised likelihood $\|\mathcal{L}^g\|$ that is used to rank all the generated grasps across all the grasp types during selection steps.

After Algorithm 2 has yielded a ranked list of optimised grasp poses, they are checked for reachability given other objects in the workspace, and unreachable poses are pruned. The remaining best scoring hand pose $h*$ is then used to generate a collision free approach trajectory to the pre-grasp wrist pose (also determined by the training grasp – see Fig. 8) and the trajectory and grasp are executed.

During grasp optimisation and selection the evaluation of product (23) accounts for over 95% of the computation time during the entire grasp inference process. A single evaluation of product (23) has time complexity $\Omega(N_Q K_Q) + \Omega(d_{h_c} K_C)$ where d_{h_c} is the dimensionality of h_c. The time complexity can be reduced to $\Omega(N_Q \log K_Q) + \Omega(d_{h_c} \log K_C)$ using k-nearest neighbour search methods (Weber et al., 1998). However, because of the overhead associated with such search structures, this approach is only justified for large values of K_Q and K_C.

5. Experimental Method

The grasp transfer performance was studied by training five models with the grasps of Fig. 8, and testing on 45 grasps of 43 unfamiliar objects (two objects were presented in two different poses each). All the training and testing reported here was performed with the Boris robot platform depicted in Fig. 1.

Views	Kernels	Initial grasp candidates	Steps K	Selection steps	Selected %	Final grasp candidates	(T_1, T_K)
1	5,000	50,000	500	1, 50	10%	500	(1,0.1)
7	2,000	2,500	500	None	100%	2,500	(1,0.1)

Table 1: Algorithm parameterisation for the two experimental conditions (1 and 7 views).

5.1. Training

Training proceeded as follows. Each training object was placed on the table, and seven views of the object were taken using a depth camera (PrimeSense Carmine 1.09). The resulting view specific depth clouds were then combined to form a single point cloud model. Stitching the point clouds together is trivial, since we know the exact pose of the camera at each frame from the robot's forward kinematics. Each grasp was then demonstrated kinesthetically by a human operator. The whole hand pose was recorded at five points along the trajectory on the final approach from a pre-grasp position selected by the operator (see Fig. 8). The hand configuration model and the contact models for each finger segment were learned from the final configuration of this trajectory. The remaining four configurations on the approach trajectory are only used to interpolate the hand configuration during execution of the approach for the transferred grasps, and are not used in model learning or grasp inference. Only kinematic information was used during training, and no force sensing of contacts was recorded.

5.2. Testing

The testing phase proceeded as follows: An object was selected from the test set, and placed on the table. Two experimental conditions were tested, where either 1 or 7 views of the test object were taken using a depth camera. The simulated annealing procedure was run using the parameters in Tab. 1. In each case the final grasp candidates were ranked by likelihood, and pruned for kinematically infeasible grasps due to collisions with the table surface. The grasp selected was the first ranked grasp. The grasp was then executed on the robot using a PRM path planner with optimisations to reach the pre-grasp position (Kopicki, 2010), and using the generated grasp trajectory thereafter. The robot hand is a DLR-HIT2 hand, which uses active compliant control based on motor current sensing at 1 kHz. The success of the grasp was determined by whether the robot could raise the object and hold it for 10 seconds. This procedure was followed for all 45 test objects for both viewing conditions. In addition for seven objects under the seven view condition the first ranked grasp of the next best grasp *type* was also tested, and for one object the first ranked grasp of the third best grasp *type* was tested. This led to a total of 98 grasps being executed across the two conditions, of which 90 were the first choice grasps. Grasp generation took an average of 23 seconds for the 1 view condition, and 12 seconds for the 12 view condition on a Intel Core i7 4-core 2.6GHz processor. Query density computation took an average of 0.7 seconds and 0.23 seconds respectively.

5.3. Results

Tab. 2 shows the grasp transfer success rate. When 7 views were taken of the test object, of the 45 first choice test grasps made 38 were successful, and 7 failed, giving a success rate among first choice grasps of 84.4%. Of the 53 different grasps (45 first choice, 7 second choice and 1 third choice) executed 46 were successful giving a success rate among all grasps of 86.7%. At least one of the first or second choice grasp worked for 95.6% of objects. When only one view was taken of the test object the successfully executed first choice grasps fell to 35, i.e. 77.8%. Fig. 9 shows examples of successful grasps. Each image pair shows the object, the partial point cloud (red), the planned grasp (yellow), and the grasp executed.

Fig. 9. The test objects with a visualisation of some of the successful grasps. Each grasp is shown by a pair of images, with the visualisation of the planned grasp and the obtained point cloud on the left, and the actual grasp execution on the right. Those from the 7 view and 1 view conditions can easily be distinguished by the proportion of the object covered by the recovered point cloud.

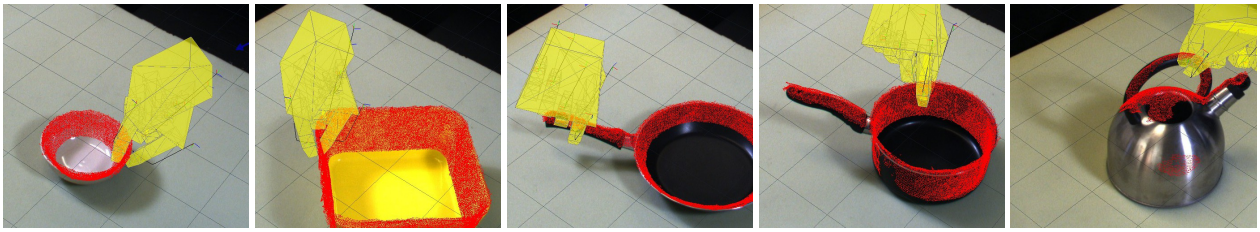


Fig. 10. The test objects with a visualisation of five of the failing grasps. The left four are from the 7 view condition and the kettle is from the 1 view condition.

		Absolute number (% of total) successes/failures			
Views	Testing objects/poses	1st choice successful	1st choice fail.	2nd & 3rd choice succ.	1st or 2nd choice succ.
1	45	35 (77.8%)	10 (22.2%)	n/a	n/a
7	45	38 (84.4%)	7 (15.6%)	8 (100%)	43 (95.6%)

Table 2: Grasp success rates for the two conditions.

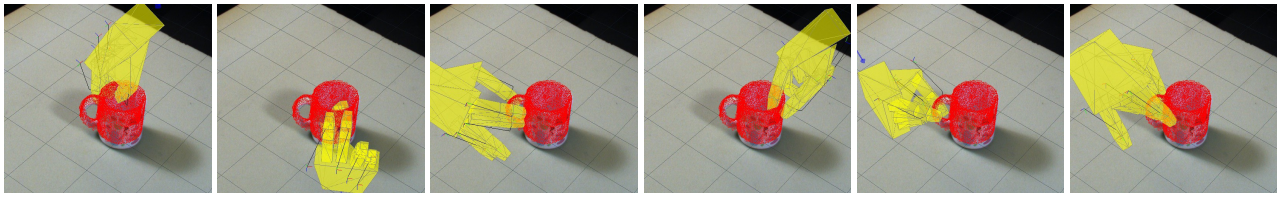


Fig. 11. Grasp variation on mug3. The grasp types used to generate these were (from top to bottom, left to right): pinch, pinch, pinch, pinch with support, handle and handle.

6. Discussion

There are several properties of the grasps generated worthy of further discussion. A complete set of images for all grasps is given in Extension 1. A supporting video with results is given in Extension 2.

Variety of grasp types. For several objects at least two grasp types were tried and executed successfully. A good example is given in Fig. 11, where the mug has six quite different grasps shown from the ranked set. Other examples from Fig. 9 include the pinch and power grasps on a coke bottle, and pinch and handle grasps on a cup. This shows the variety of grasp types the method is able to use when the object is presented in the same orientation. This variety matters for general grasping ability. In the seven view condition, when first grasps failed, the second choice grasp-type always succeeded (except two cases where it wasn't attempted for safety reasons). This means that from first or second choice grasp-types at least one of

Grasp type	1st choice occurrences	
	7 views	1 view
pinch	20	20
pinch w/ support	20	20
handle	3	4
power-box	2	0
power-tube	0	1

Table 3: First choice grasp type distribution for the two conditions.

these worked for 43 of the 45 object-pose combinations. Thus 95.5% of the test objects were successfully grasped by one of the top two grasp types.

Robustness to partial surface data. Fig. 9 shows the recovered point cloud for each test grasp. In the right column it can be seen that successful grasps were made in the face of quite small amounts of surface data being recovered. This is true even of quite complex grasps, such as the handle grasp of the mug.

Robustness to object pose. The method is robust to reorientations of the object. In Fig. 9 the large funnel is presented point up and bowl up respectively, and yet the grasp is adapted from the same base grasp (pinch with support) in both cases. In the case of the guttering the grasps selected are pinch and pinch with support respectively in response to different orientations of the guttering on the table.

Preference for simple grasps. The method is capable of generating a variety of grasp types, but the preferred grasps typically involve fewer finger links. This reflects the greater ease adapting them to more closely match the conditions of the original grasp: fewer finger links involved in the grasp means fewer constraints. This is a different property to the need to rescale grasp likelihood by the number of links involved. In that case as the number of links rises the grasp likelihood falls. That would be the case whether or not two grasps being compared were identical to their training grasps and evaluated on the training objects.

Grasping different object parts. The method generates a large number of grasps. These have a high degree of variation in their pose on the object. This is also shown in Fig. 11. Note that due to the incomplete point cloud some are reasonable even though they are not feasible. Since many missing points are underneath the object these grasps are typically not kinematically feasible either and so are pruned. The variety of grasps generated supports the idea that the method will allow grasping in cluttered scenes, or to find a suitable grasp in the face of task constraints, although testing these hypotheses falls beyond the scope of this paper.

Degree of generalisation. When viewing the transferred grasps next to the example grasp the degree of generalisation is notable. Fig. 12(top) shows three grasps together with the training grasps from which they were adapted. The adaptation from bowl to funnel spout and to a spray bottle shows the generalisation ability of the pinch with support grasp. The grasp of the guttering using a pinch grasp widens the finger spacings significantly with respect to the example grasp on the tube. In addition the global shape of these test objects is different from the training examples. The variety of grasps achievable by adapting one learned grasp is shown by the adaptations of the pinch with support grasp type in Fig. 9. The bucket, funnel (both orientations), guttering, kettle and spray bottle are all adaptations of this grasp type.

Failing grasps. it is worth analysing why grasps fail. Fig. 10 show five failing first choice grasps. The grasp of the bowl failed because the pinch grasp together with the low frictional coefficient of the objects don't give sufficient frictional contact to achieve force closure. In the case of the saucepan the grasp is in the wrong place: the grasp of the rim can't resist the wrench given by the large, heavy object, a grasp around the handle would be better. This was tried for the frying pan, but the wrong type of grasp was used. Instead an adaptation of the power-tube grasp succeeded on the saucepan in Fig. 9. The grasp of the kettle failed because in the single view condition the surface reconstruction is so limited it affects the grasp quality significantly. Finally some grasps, such as the grasp of the yellow container, fail while being superficially very similar to successful grasps of the same object.

7. Conclusions

This paper has presented a method that generalises a single kinesthetically demonstrated grasp to generate many grasps of other objects of different and unfamiliar shapes. One essential element is learning a separate *contact model* for each finger phalange how of its pose relative to the surface is related to local surface feature. This encodes local contact constraints. Another is learning a *hand configuration model* based on sampling poses near to those on the approach trajectory in the training example. This encodes the global handshape.

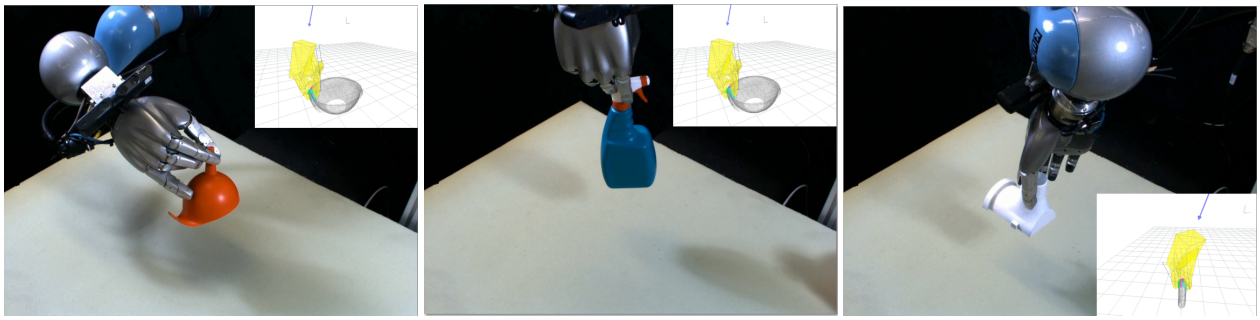


Fig. 12. Examples of transfer to globally very different objects.

The optimisation process is seeded by these two models with many different starting positions, and the resulting optimised grasps are then clustered to yield a variety of different viable grasp candidates. This is advantageous because the candidates can then be further assessed and ranked by stability, reachability, and task suitability. This paper has described simple ranking by similarity. This step could itself be a precursor to other analysis of the ranked grasps.

The empirical studies performed show that: i) the method can learn from one example grasp of a particular type; ii) the system creates grasps for objects with globally different shapes from the training objects; iii) for each new object many different new grasps can be generated, ordered by likelihood, allowing the selection of grasps that satisfy workspace constraints; iv) successful new grasps can also be generated even where shape recovery is incomplete for the new object; v) grasp success rates on test objects are high and robust to partial surface recovery: 84.4% with seven views, and 77.8% with one view.

7.1. Future Work

Many problems remain in dexterous grasping. This work provides a step forward in terms of grasp generation and generalisation. Reasoning about such grasps by other algorithms is the next step. In particular the appeal of dexterous hands is that they enable a variety of ways for the hand to interact with the object, and selecting the initial grasp so as to enable the particular chosen interactions or task is a necessary problem to tackle. Grasp refinement using reinforcement learning is another obvious route to improving the grasps created using the methods here.

References

C. Bard & J. Troccaz (1990). 'Automatic preshaping for a dextrous hand from a simple description of objects'. In *International Workshop on Intelligent Robots and Systems*, pp. 865–872. IEEE.

Y. Bekiroglu, et al. (2011). 'Integrating Grasp Planning with Online Stability Assessment using Tactile Sensing'. In *International Conference on Robotics and Automation*, pp. 4750–4755. IEEE.

H. Ben Amor, et al. (2012). 'Generalization of human grasping for multi-fingered robot hands'. In *International Conference on Intelligent Robots and Systems*, pp. 2043–2050. IEEE.

A. Bicchi & V. Kumar (2000). 'Robotic grasping and contact: a review'. In *International Conference on Robotics and Automation*, pp. 348–353. IEEE.

E. Borra, et al. (2011). 'Anatomical evidence for the involvement of the macaque ventrolateral prefrontal area 12r in controlling goal-directed actions'. *The Journal of Neuroscience* **31**(34):12351–12363.

G. I. Boutselis, et al. (2014). 'Task Specific Robust Grasping For Multifingered Robot Hands'. In *International Conference on Robotics and Automation*, pp. 858–863. IEEE.

J. Coelho, et al. (2000). 'Developing haptic and visual perceptual categories for reaching and grasping with a humanoid robot'. In *Robotics and Autonomous Systems*, vol. 37, pp. 7–8.

R. Detry, et al. (2013). 'Learning a Dictionary of Prototypical Grasp-predicting Parts from Grasping Experience'. In *International Conference on Robotics and Automation*, pp. 601–608. IEEE.

A. H. Fagg & M. A. Arbib (1998). 'Modeling parietal-premotor interactions in primate control of grasping'. *Neural Networks* **11**(7-8):1277–1303.

C. Ferrari & J. Canny (1992). 'Planning optimal grasps'. In *International Conference on Robotics and Automation*, pp. 2290–2295.

D. Fischinger & M. Vincze (2012). 'Empty the basket – a shape based learning approach for grasping piles of unknown objects'. In *International Conference on Intelligent Robots and Systems*, pp. 2051–2057. IEEE/RSJ.

R. A. Fisher (1953). 'Dispersion on a sphere'. In *Proc. Roy. Soc. London Ser. A.*, vol. 217, pp. 295–305. Royal Society.

A. K. Goins, et al. (2014). 'Evaluating the Efficacy of Grasp Metrics for Utilization in a Gaussian Process-Based Grasp Predictor'. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 3353–3360. IEEE/RSJ.

I. Gori, et al. (2014). 'Three-Finger Precision Grasp on Incomplete 3D Point Clouds'. In *IEEE International Conference on Robotics and Automation*, pp. 5366–5373. IEEE.

R. Grupen (1991). 'Planning grasp strategies for multifingered robot hands'. In *IEEE International Conference on Robotics and Automation*, pp. 646–651. IEEE.

K. Hang, et al. (2014). 'Combinatorial Optimization for Hierarchical Contact-level Grasping'. In *IEEE International Conference on Robotics and Automation*, pp. 381–388. IEEE.

A. Herzog, et al. (2014). 'Learning of grasp selection based on shape-templates'. *Autonomous Robots* **36**(1-2):51–65.

U. Hillenbrand & M. Roa (2012). 'Transferring functional grasps through contact warping and local replanning'. In *IEEE/RSJ International Conference on Robotics and Systems*, pp. 2963–2970. IEEE.

Y. Hu, et al. (1999). 'Human Visual Servoing for Reaching and Grasping: The Role of 3-D Geometric Features'. In *IEEE International Conference on Robotics and Automation*, pp. 3209–3216. IEEE.

L. S. Jakobson & M. A. Goodale (1991). 'Factors affecting higher-order movement planning: a kinematic analysis of human prehension'. *Experimental Brain Research* **86**(1):199–208.

I. Kamon, et al. (1996). 'Learning to grasp using visual information'. In *IEEE International Conference on Robotics and Automation*, vol. 3, pp. 2470–2476.

K. Kanatani (2005). *Statistical optimization for geometric computation: theory and practice*. Courier Dover Publications.

J. Kim (2007). 'Example-based Grasp Adaptation'. Master's thesis, Massachusetts Institute of Technology.

J. Kim, et al. (2013). 'Physically Based Grasp Quality Evaluation Under Pose Uncertainty'. *IEEE Transactions on Robotics* **29**(6):1424 – 1439.

S. Kirkpatrick, et al. (1983). 'Optimization by simulated annealing'. *Science* **220**(4598):671–680.

E. Klingbeil, et al. (2011). 'Grasping with application to an autonomous checkout robot'. In *IEEE International Conference on Robotics and Automation*, pp. 2837–2844. IEEE.

G. W. Kootstra, et al. (2012). 'Enabling grasping of unknown objects through a synergistic use of edge and surface information'. *The International Journal of Robotics Research* **34**:26–42.

M. Kopicki (2010). *Prediction learning in robotic manipulation*. Ph.D. thesis, University of Birmingham.

M. Kopicki, et al. (2014). 'Learning dextrous grasps that generalise to novel objects by combining hand and contact models'. In *IEEE International Conference on Robotics and Automation*, pp. 5358–5365. IEEE.

O. Kroemer, et al. (2010). 'Combining Active Learning and Reactive Control for Robot Grasping'. *Robotics and Autonomous Systems* **58**(9):1105–1116.

O. Kroemer, et al. (2012). 'A kernel-based approach to direct action perception'. In *IEEE International Conference on Robotics and Automation*, pp. 2605–2610. IEEE.

Y.-H. Liu (2000). 'Computing n-finger form-closure grasps on polygonal objects'. *The International Journal of Robotics Research* **19**(2):149–158.

A. Morales, et al. (2004). 'Using Experience for Assessing Grasp Reliability'. *International Journal of Humanoid Robotics* **1**(4):671–691.

R. Platt, et al. (2006). 'Learning Grasp Context Distinctions that Generalize'. In *Proceedings of the IEEE-RAS International Conference on Humanoid Robots*, pp. 504 – 511. IEEE.

N. S. Pollard (2004). 'Closure and quality equivalence for efficient synthesis of grasps from examples'. *The International Journal of Robotics Research* **23**(6):595–613.

J. Ponce & B. Faverjon (1995). 'On computing three-finger force-closure grasps of polygonal objects'. *IEEE Transactions on Robotics and Automation* **11**(6):868–881.

M. Popović, et al. (2010). 'A Strategy for Grasping Unknown Objects based on Co-Planarity and Colour Information'. *Robotics and Autonomous Systems* **58**(5):551–565.

M. Richtsfeld & M. Zillich (2008). 'Grasping unknown objects based on 2.5D range data'. In *IEEE International Conference on Automation Science and Engineering*, pp. 691–696. IEEE.

G. Rizzolatti & G. Luppino (2001). 'The cortical motor system'. *Neuron* **31**(6):889–901.

C. Rosales, et al. (2011). 'Synthesizing grasp configurations with specified contact regions'. *The International Journal of Robotics Research* **30**(4):431–443.

C. Rosales, et al. (2012). 'On the synthesis of feasible and prehensile robotic grasps'. In *IEEE International Conference on Robotics and Automation*, pp. 550–556. IEEE.

R. B. Rusu, et al. (2009). 'Perception for Mobile Manipulation and Grasping using Active Stereo'. In *IEEE-RAS International Conference on Humanoids*, pp. 632–638. IEEE.

J. Saut & D. Sidobre (2012). 'Efficient models for grasp planning with a multi-fingered hand'. *Robotics and Autonomous Systems* **60**(3):347–357.

A. Saxena, et al. (2008). 'Learning grasp strategies with partial shape information'. In *Proceedings of AAAI*, pp. 1491–1494. AAAI.

A. Shapiro, et al. (2004). 'On the mechanics of natural compliance in frictional contacts and its effect on grasp stiffness and stability'. In *IEEE International Conference on Robotics and Automation*, vol. 2, pp. 1264–1269. IEEE.

K. B. Shimoga (1996). 'Robot grasp synthesis algorithms: A survey'. *The International Journal of Robotics Research* **15**(3):230.

B. W. Silverman (1986). *Density Estimation for Statistics and Data Analysis*. Chapman & Hall/CRC.

M. Spivak (1999). *A comprehensive introduction to differential geometry*, vol. 1. Publish or Perish Berkeley.

E. B. Sudderth (2006). *Graphical models for visual object recognition and tracking*. Ph.D. thesis, MIT, Cambridge, MA.

A. ten Pas & R. Platt (2014). 'Localizing Handle-like Grasp Affordances in 3D Point Clouds'. In *International Symposium on Experimental Robotics*.

M. Trobina & A. Leonardis (1995). 'Grasping arbitrarily shaped 3-D objects from a pile'. In *IEEE International Conference on Robotics and Automation*, pp. 241–246. IEEE.

R. Weber, et al. (1998). 'A quantitative analysis and performance study for similarity-search methods in high-dimensional spaces'. In *Proceedings of the 24th VLDB Conference*, vol. 98, pp. 194–205.

J. Xu, et al. (2007). 'Force analysis of whole hand grasp by multifingered robotic hand'. In *IEEE International Conference on Robotics and Automation*, pp. 211–216. IEEE.

L. E. Zhang, et al. (2011). 'Grasp Evaluation with Graspable Feature Matching'. In *RSS 2011 Workshop on Mobile Manipulation: Learning to Manipulate*.

Y. Zheng & W.-H. Qian (2005). 'Coping with the grasping uncertainties in force-closure analysis'. *The International Journal of Robotics Research* **24**(4):311–327.

Appendix A: Index to Multimedia Extensions

Extension	Media Type	Description
1	Images	Images of every grasp for both conditions (1 and 7 view)
2	Video	Video with high level explanation and results

A.4 Technical Report: Informational Reward Planning for Dexterous Grasping in Unstructured Environments

Authors Claudio Zito and Jeremy L. Wyatt

Info Unpublished

Abstract Dexterous grasping of objects with uncertain pose is a hard unsolved problem in robotics. In this work we extend our proposed solution to this problem using information gain re-planning. We present a new set of algorithms that enable: i) re-planning while maintaining the configuration of the robot in contact with the object when an unexpected contact occurs, ii) planning of dexterous grasping trajectories for non-convex object shapes by implementing efficient collision detection for point clouds, and iii) use of an active complaint controller to allow continuous contact while minimising the risk of perturbing the object being grasped. First we show how tactile information, acquired during a failed attempt to grasp an object can be used to refine the estimate of that objects pose. Second, we show how this information can be used to re-plan new reach to grasp trajectories for successive grasp attempts. Third, we show how complex, dexterous grasping trajectories can be efficiently planned for a non-convex object described as a point cloud. Finally we present IR3ne, our information reward based algorithm, to show how reach-to-grasp trajectories can be modified, so that they maximise the expected tactile information gain, while simultaneously delivering the hand to the grasp configuration that is most likely to succeed. The method is demonstrated in trials with a simulated robot. Sequential re-planning is shown to achieve a greater success rate than single grasp attempts, and IR3ne, with its trajectories that maximise information gain, requires fewer re-planning iterations than conventional planning methods before a grasp is achieved.

Relation with the deliverable: This report describes our work on Task 4.4, grasping under uncertainty.

Attachment (following pages until next annex)

PaCMan TR-2015-2

Informational Reward Planning for Dexterous Grasping in Unstructured Environments

C. Zito and J.L. Wyatt*
School of Computer Science, University of Birmingham, UK

Abstract

Dexterous grasping of objects with uncertain pose is a hard unsolved problem in robotics. In this work we extend our proposed solution to this problem using information gain re-planning. We present a new set of algorithms that enable: i) re-planning while maintaining the configuration of the robot in contact with the object when an unexpected contact occurs, ii) planning of dexterous grasping trajectories for non-convex object shapes by implementing efficient collision detection for point clouds, and iii) use of an active complaint controller to allow continuous contact while minimising the risk of perturbing the object being grasped.

First we show how tactile information, acquired during a failed attempt to grasp an object can be used to refine the estimate of that object's pose. Second, we show how this information can be used to re-plan new reach to grasp trajectories for successive grasp attempts. Third, we show how complex, dexterous grasping trajectories can be efficiently planned for a non-convex object described as a point cloud. Finally we present *IR3ne*, our information reward based algorithm, to show how reach-to-grasp trajectories can be modified, so that they maximise the expected tactile information gain, while simultaneously delivering the hand to the grasp configuration that is most likely to succeed.

The method is demonstrated in trials with a simulated robot. Sequential re-planning is shown to achieve a greater success rate than single grasp attempts, and IR3ne, with its trajectories that maximise information gain, requires fewer re-planning iterations than conventional planning methods before a grasp is achieved.

Keywords

Dexterous grasping, planning, informational reward, object-pose uncertainty, object-shape uncertainty, pose density from point cloud, collision detection for point clouds.

1. Introduction

In robot grasping, there is typically uncertainty associated with the location of the object to be grasped. This can be due to poor results of vision or depth sensing, or to simply due to incomplete information about object shape caused by incomplete view coverage. If the object is not in its expected location, then a robot equipped with tactile sensors, or torque sensors at finger joints, may gain some additional information to help refine localisation knowledge from tactile contacts (or lack of thereof) occurring during the execution of a reach to grasp trajectory.

* Corresponding author; e-mail: {cxz004, jeremy.l.wyatt}@cs.bham.ac.uk

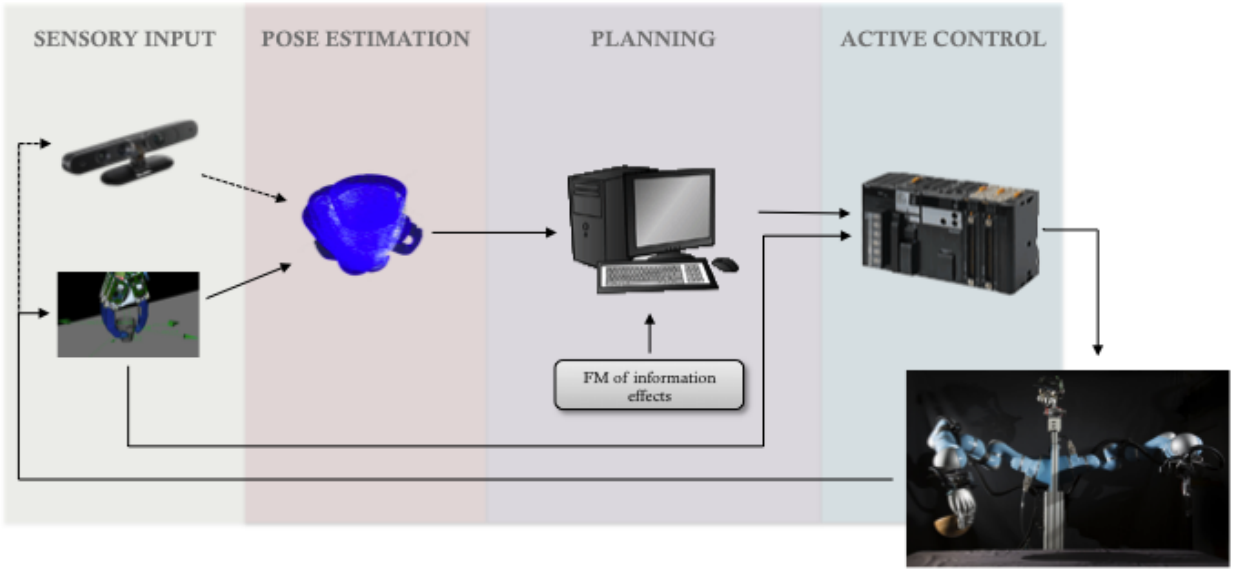


Fig. 1. The architecture of our system is composed of 4 components: sensory input, pose estimation, motion planning and active control. Solid lines highlight the components involved in the re-planning procedure, whilst dashed lines identify interactions that happen only at the first iteration. We employ a depth camera to obtain an incomplete point cloud of the object surface. Using a model fitting procedure a probability density over the object pose is estimated, represented as a particle set. Given this distribution, a trajectory is planned. A forward model (FM) is used to maximise the chance of gathering tactile observations that will reduce pose uncertainty. If unexpected observations occur while the trajectory is executed, then we use them to perform a particle filter update, and the active control enables the robot to keep the contact without applying eccessive force. Re-planning then occurs with the new pose distribution, and a new reach-to-grasp trajectory is constructed. This process is then iterated until a successful grasp is achieved.

In this paper we: i) describe how to efficiently plan collision free, dexterous grasping trajectories for a non-convex object represented as point cloud, ii) describe how to iteratively update the object pose estimate using tactile observations, iii) show how successive grasp trajectories can be planned with respect to these iteratively refined object poses, and iv) show how each reach-to-grasp trajectory can be deliberately planned to maximise information gain from potential tactile contacts, while also planning to terminate at the target grasp given the object is at some assumed pose. This work is an extension of our early work presented in [Zito et al., 2012; 2013].

We achieve these advances by: i) using a hierarchical sample-based path planner, here a Probabilistic Roadmap (PRM) planner, which encodes expected information gain (in a low-dimensional belief space) for each of its trajectory segments, building on the work of [Platt et al., 2001] and [Zito et al., 2013]; ii) refining the belief state for the object pose using an observation model for contact sensing by a multi-finger hand. We show that this approach enables planning for a robot manipulator with 20 DoFs and non-Gaussian object pose uncertainty in 6 dimensions.

We make several assumptions. First we assume a reference model for the object is known. Nevertheless, completeness in this model is not necessary for our algorithm to work. Incomplete point clouds can be used as models. Second we assume that a separate sequence of views given by a depth camera give an incomplete point cloud which can be aligned to this model. The shape incompleteness of both model and observation of the object results in object-pose uncertainty encoded in a belief density for the pose. Given this our approach selects the optimal reach to grasp trajectory with respect to the current belief state. Second we assume that the algorithm takes as input a target grasp configuration on the object (i.e. the set of finger contacts on the model), and a pre-grasp configuration. We use the method described in [**?**] to generate this target grasp for the presented object. Third we assume that the object is stationary and we actively control the compliance of the robot when in contact with the object without altering the object pose. Fourth we assume the availability of a belief state estimate of the object's pose in the form of a particle filter.

The architecture of our system is shown in Fig. 1. In our scenario we employ a depth image obtained from an Asus Xtion Pro depth camera. This gives an incomplete point cloud of the object surface. Using a model fitting procedure similar to the sampling from surflet pairs method presented in [Hillenbrand & Fuchs, 2011], a probability density (or belief state) over the object pose is estimated, represented as a particle set. Given this distribution, a reach-to-grasp trajectory is planned. This trajectory has as its goal configuration the pre-computed grasp under the assumption that the object is at a pose corresponding to the mean position of the particle set. The path to this goal configuration is found using a stochastic motion planner. This planner works with a cost function that allows deviations from the shortest path that maximise the chance of gathering tactile observations that will reduce pose uncertainty in the object location. If unexpected observations occur during the execution of the planned trajectory, then such observations (both tactile contact and no-contact signals) collected at poses along the reach-to-grasp trajectory are used to perform a particle filter update. Re-planning then occurs with the new pose distribution, and a new reach-to-grasp trajectory can be constructed. This process is then iterated until a successful grasp is achieved.

The benefit of planning with beliefs is the ability to reason about the informational effects of sequences of actions. In typical belief space planning this means performing a kind of pre-posterior analysis, in which planned actions (here trajectories) cause imagined observations that are in turn used to perform a Bayes' update of the belief state. As the belief space grows exponentially in the length of the planned action-observation sequence these methods are exact but inefficient.

Our work instead builds on the approach of Platt et al. [Platt et al., 2001; 2012]. That work plans a sequence of actions that will generate observations that distinguish a hypothesised state from competing hypotheses while also reaching a goal position. The informational value of a trajectory is the difference in the expected observations between the hypothesised position and each alternative. This approach allows us to track high-dimensional belief states using an accurate filter defined by the user, but reduces the complexity of planning in belief spaces by approximating the informational value of actions from a low-dimensional subspace of the belief state. Platt et al. applied this to planning for a two degree of freedom manipulator using a laser range finder for observations, and employed an optimisation framework for planning. The algorithm is proved to localise the true state of the system in 1 dimension and to reach a goal region with high probability. In contrast to [Platt et al., 2001], our approach encodes information gathering actions to localise an object to be grasped in 6 dimensions while simultaneously attempting to achieve the grasp. Similarly to [Platt et al., 2001; 2012], our method is guaranteed to converge to the true state of the system in which a reach-to-grasp trajectory succeeds with high probability. Nevertheless, convergence is guaranteed only if the system is not perturbed. In our previous work [Zito et al., 2012; 2013], we thus assumed that the system was i) static (does not change over time) and ii) the robot's palpation skills were sufficiently accurate to retrieve a contact without altering the configuration of the object. For this work we relax the latter assumption by developing an active compliant controller which enables the manipulator to work as a non-linear spring when in contact with the object. We now briefly survey other work on information gathering while grasping.

2. Related work

Other authors have attempted to pose the simultaneous grasping and localisation problem (SLAG) in terms of a partially observable Markov decision process (POMDP). Hsiao et al., [Hsiao et al., 2007], present a simple blocks world problem, wherein a single finger can execute a small selection of actions (left, right, up, down) in response to a small set of possible sensory detections (contact or no contact below, right or left sides of finger). This simple problem is tractable for solving as a POMDP, producing optimal policies for guiding the finger towards a desired location on a 2D blocks world object. The same authors later addressed real-world grasping with an arm and three-finger Barrett hand equipped with tactile fingertip sensors, [Hsiao & Kaelbling, 2010], [Hsiao et al., 2011]. Because the space of possible actions for such a robot is enormous (especially compared to the single finger and 2D blocks-world problem of [Hsiao et al., 2007]), in order to solve the problem as a POMDP, the authors restrict the robot's choice of actions to executing a small number of pre-programmed

reach-to-grasp motions, described relative to the pose of the target object. Thus the POMDP method can tractably be used to select between this small number of actions, in response to tactile contacts detected by the three fingertip sensors during the previous action. Thus various actions are sequentially selected, with successive refinements of the object pose collected from sensing during each action, until eventually one of the actions achieves a successful grasp.

In contrast to [Hsiao & Kaelbling, 2010], our method does not select between pre-defined high level actions, but instead we describe how to optimally plan a novel dexterous reach-to-grasp trajectory, in response to recent sensory observations, which maximises the expected information gain from further potential tactile observations.

Petrovskaya [Petrovskaya & Khatib, 2011] also investigated the use of tactile information, derived from collisions of an end effector with an object, to refine localisation estimates for that object. Similarly to our method, Petrovskaya represents the location of an object to be grasped, as a collection of particles forming a distribution, and this distribution is refined by successive manipulator contacts with the object. However, the work is limited in that Petrovskaya does not address the problem of planning manipulator trajectories to achieve these contacts. Instead, the author simply begins with an assumption that a selection of collisions will occur, and then shows how to use such collisions to update an object location distribution.

More recently, another probabilistic framework for sensor-based grasp planning has been presented in [Nikandrova et al., 2013]. Their approach philosophy differs from ours in the sense that the uncertainty in the pose of the object affects only the choice of the (final) grasp configuration of the robot - in terms of set of finger contacts and wrist pose - which maximises some criterions of stability. The reach to grasp trajectory is compute using a state-of-the-art planner which assume the estimated pose to be accurate. They have proposed two strategies: either based on i) stability maximisation or ii) entropy minimisation. The former approach is implemented as a trial and error procedure in which tactile measurements, at the end of the grasp trajectory, are used to update the belief state - represented as a particle filter - and then the robot is move to a fix position at the beginning of each iteration. The latter approach selects the most informative grasp to reduce entropy and fasten the convergency to a stable grasp. However this requires a pre-posterior analysis which is expensive as the belief space grows. Both approaches have been evaluated in simulation with a proof of concept on a real robotic platform, a Melfa RV-3SB 6DoFs arm and a Robotic 3-finger hand. However the experimental setup consists of a simplify 2D problem in which a 2D Gaussian noise is manually added to the object estimate. In contrast our method can cope with non-Gaussian uncertainty in 6 dimensions generated directly from the visual sensory data.

Prentice et al. [Prentice & Roy, 2008] have shown that motion planning for a mobile robot in belief space can be done efficiently for linear Gaussian systems by using a factored form of the covariance matrix. The authors incorporate an estimation of localisation uncertainty in the cost function of a belief-space variant of the PRM. Similarly to [Prentice & Roy, 2008] our approach is capable of balancing shorter paths against those which reduce belief uncertainty through sensory information gain. Instead our work plans trajectories that are able to distinguish hypotheses by using the expected sequence of observations, which are computed according to the current belief state. Furthermore our approach represents the belief space as non-Gaussian and allows arbitrary implementations of Bayes filters to track belief states.

3. Planning trajectory

3.1. Problem formulation

This section is concerned with the problem of planning control actions to reach a goal state in the presence of incomplete or noisy observations. Let us consider a discrete-time system with continuous non-linear deterministic dynamics,

$$x_{t+1} = f(x_t, u_t)$$

where $x_t \in \mathbb{R}^n$ is a configuration state of the robot and $u_t \in \mathbb{R}^l$ is a action vector, both parametrised over time $t \in \{1, 2, ...\}$. Let $p \in SE(3)$ describe the object pose, given an initial prior belief state b_1 we define a set of k hypotheses as $\{p^i\}_{i=1}^k$, where $p^1 = \arg\max b_1$ and $p^i \sim b_1, i \in [2, k]$. We search for a sequence of actions, $u_{1:T-1} = \{u_1, \ldots, u_{T-1}\}$, that distinguish between observations that would occur if the object were in p^1 from any other p^i pose, with $i \in [2, k]$. At each time step, t, the system makes an observation, $y \in \mathbb{R}^m$, that is a non-linear stochastic function of states and hypotheses. Without losing generality, we define y_t to be a column vector of binary values. Each of these values represents whether or not a contact is observed between a given link of the robot and the hypothesis p^i. However, binary values have been shown to be not very informative during the planning phase. Therefore we define,

$$h(x, p^i) = p(y = 1|x, p^i)$$

as a column vector of scores identifying the likehood of observing a contact, $y = 1$, as function of states and hypotheses. More generally, let $F_t(x, u_{1:t-1})$ be the robot configuration at time t if the system begins at state x and takes action $u_{1:t-1}$. Therefore the expected sequence of observations over a trajectory, $u_{1:t-1}$, is:

$$h_t(x, u_{1:t-1}, p^i) = (h(F_2(x, u_1), p^i)^T, h(F_3(x, u_{1:2}), p^i)^T, \ldots, h(F_t(x, u_{1:t-1}), p^i)^T)^T$$

a column vector which describes the likelihood of observing a contact at any time step of the trajectory $u_{1:t-1}$. We then search for a sequence of actions which maximise the difference between observations that are expected to happen in the sampled states, $p^{2:k}$, when the system is actually in the most likely hypothesis, p^1. In other words, we want to find a sequence of action, $u_{1:T-1}$, that minimises

$$J(x, u_{1:T-1}, p^{1:k}) = \sum_{i=2}^{k} N(h(x, u_{1:T-1}, p^i)|h(x, u_{1:T-1}, p^1), \mathbb{Q}) \tag{1}$$

where $N(\cdot|\mu, \Sigma)$ denotes the Gaussian distribution with mean μ and covariance Σ and $\mathbb{Q}$ is the block diagonal of measurement noise covariance matrices of appropriate size. Rather than optimising (1) we follow the suggested simplifications described in [Platt et al., 2001] by dropping the normalisation factor in the Gaussian and optimising the exponential factor only. Let us define for any $i \in [2, k]$

$$\Phi(x, u_{1:T-1}, p^i) = ||h_t(x, u_{1:T-1}, p^i) - h_t(x, u_{1:T-1}, p^1)||^2_{\mathbb{Q}},$$

then the modified cost function is

$$J(x, u_{1:T-1}, p^{1:k}) = \frac{1}{k} \sum_{i=2}^{k} e^{-\Phi(x, u_{1:T-1}, p^i)} \tag{2}$$

it is worth noting that when there is a significant difference between the sequence of expected observations, $h_t(x, u_{1:T-1}, p^i)$ and $h_t(x, u_{1:T-1}, p^1)$, the function $\Phi(\cdot)$ is large and therefore $J(\cdot)$ is small. On the other hand if the sequences of expected observations are very similar to each other, their distance measurement tends to 0 and $J(\cdot)$ tends to 1. Equation (2) can be minimised using different planning techniques such as Rapidly-exploring Random Trees (RRTs) [LaValle, 1998], Probabilistic Roadmap (PRM) [Kavraki & Svestka, 1996], Differential Dynamic Programming (DDP) [Jacobson & Mayne, 1970] or Sequential Dynamic Programming (SDP) [Betts, 2001]. In Section 4.3, we define a new set of heuristics that can be encoded in a PRM planner for generating more reliable trajectories that explicitly reason over the pose uncertainty, and demonstrate the method with experiments on a virtual testbed.

3.2. Belief Update

After a trajectory is planned our algorithm executes it. If an unexpected observation occurs at execution-time the algorithm refines the current belief state using an accurate, high-dimensional filter provided by the user.

In order to define an unexpected observation, it may be convenient for the reader to think of a reach-to-grasp trajectory as composed of two parts: i) an approach trajectory which leads to a pre-grasp configuration of the robot in which the fingers generally cage the object to be grasped without generating any contact, and ii) a grasping trajectory which moves the robot in contact and eventually generates a force closure grasp. In this way any contact which occurs during the approaching trajectory is considered as an unexpected observation. Similarly a insufficient number of contacts to create a force closure at the end of a grasping trajectory is considered as unexpected.

In our implementation we update the belief using Bayes rule assuming deterministic dynamics. In this case we can write the belief update as

$$b_{t+1} = \frac{P(y_{t+1}|x_t, u_t)b_t}{P(y_{t+1})} \tag{3}$$

3.3. Re-planning

Our algorithm plans trajectories assuming only the maximum likelihood observations given the current belief state. Therefore we need to rely on sensory feedback during the execution of the planned trajectory in order to detect whether or not unexpected observations occur. This triggers a belief update, using the observation gathered at execution-time, and consequently a re-planning phase.

In our previous work [Zito et al., 2012; 2013] the manipulator was moved back to a safe configuration (e.g. outside the uncertain region) and before a new reach-to-grasp trajectory was planned. This approach was necessary for technical reasons due to the robotic platform in use. More precisely, our approach was tested on DLR's Rollin Justin robot. The only way to communicate with this platform is via its own controller. This controller accepts trajectories with extra parameters to set, i.e., compliance or activate/disactivate joint's torques thresholds (or guards) to detect contacts minimising the risk of moving the object. However these parameters cannot be changed on-line during the execution of the trajectory and the controller forbids any movement once that an active guard has been triggered. It is possible however to send trajectory disabling guards. Therefore our proposed solution was to withdraw the robot to a safe configuration with no active guard before the next reach to grasp attempt. This however, is inefficient and fails to exploit the existing contact in completing the grasp.

To overcome these limitations we implemented our latest algorithms for our robotic platform called Boris. Our controller for Boris enables us to modify settings on the fly at execution time. This allows us to make a contact with an object, stop the robot, and then re-plan from the same configuration, maintaining contact with the object.

In our experiments, the algorithm uses torque sensors based on current draw at each joint of the robot's hand to detect whether or not a link of the hand is in contact with the environment. We assume that the sensing abilities of the robot are fine enough to perceive the object without moving it. However even in simulation is difficult to maintain such an assumption. Though our results show that small changes in the configuration of the object do not affect the algorithm which is still able to converge to a force closure grasp.

3.4. Terminal conditions

Our algorithm terminates its execution when no unexpected contacts occur and the target grasp is achieved. Since we do not have mesh models of the object to be grasped we cannot rely on such measures to signal successful termination of the algorithm. Nonetheless, in simulation it is possible to measure the displacement error between the grasp configuration for

the final object-pose estimate and the ground truth. An user-defined threshold of tolerance is used to identify whether the grasp has succeeded.

4. Implementation

Our approach consists in planning informative tactile observations for a dexterous hand while simultaneously reaching a given target grasp. As described previously, our innovations are i) implementing a hierarchical PRM algorithm which allows us to plan dexterous reach-to-grasp trajectories, ii) encoding a new set of heuristics for a randomised motion planner and iii) formalising an observational model for contact sensing by a multi-finger hand. We tested our approach for a robotic manipulator with 21 DoF under non-Gaussian object pose uncertainty in 6 dimensions.

4.1. Observation model

We assume that a robotic manipulator is composed of parts. These parts are considered collections (or chains) of joints, linked together. Without loss of generality, we also assume a single part called the'arm' and a set of m parts called 'fingers'. In addition, we describe the observational model as limited only to a given subset of those parts, i.e. only fingers or a subset of them (e.g. finger tips). Let M be the ordered set of parts which compose the manipulator, then $x(j)$ is the configuration in joint space of the j^{th} part, with $j \in M$. In other words, j is the index of a specific chain. We also define $\bar{M}$ to be the set of indices such that the respective part is used in our observation model. In addition, we use the operator $W(x(j))$ to refer to the workspace coordinates in $SE(3)$ of the j^{th} joint w.r.t. a given reference frame.

Mathematically we formalise the likelihood of observing a contact for each finger of the robot as an exponential distribution over the Euclidean distance, d_{ji}, between the finger tip's pose, $W(x(j))$, and the closest surface of the object assumed to be in pose p^i. Note that, for this work, we limited our observation model to contacts which may occur on the internal surface of fingers. This directly affects our planner which rewards trajectories that would generate contacts on the finger tips rather than on the back side of the fingers. Therefore for any $j \in \bar{M}$ we write

$$p(y(j) = 1|x(j), p^i) = \begin{cases} \eta \exp(-\lambda d_{ji}) & \text{if } d_{ji} \leq d_{max} \\ & \text{and } \langle n_{xj}, \hat{n_{p^i}} \rangle < 0 \\ 0 & \text{otherwise} \end{cases}$$

where $\langle n_{xj}, \hat{n}_{p^i} \rangle$ is the inner product of, respectively, the j^{th} finger tip's normal and the estimated object surface's normal, and d_{max} describes a maximum range in which the likelihood of reading a contact is not zero, and η is a normaliser. This allows us to rewrite the likelihood of reading a contact on the force/torque sensors of the robot, $h(x, p^i), i \in [1, k]$ with $j_1, \ldots, j_m \in \bar{M}$ as follows,

$$h(x, p^i) = [p(y(j_1) = 1|x(j_1), p^i), \ldots, p(y(j_n) = 1|x(j_m), p^i)]^T$$

4.2. Planning a dexterous grasping trajectory for non-convex objects

The implementation of our planner uses a modified version of Probabilistic Roadmap (PRM) planning, [Kavraki & Svestka, 1996], to plan trajectories. Crucially, sampling-based approaches like PRMs rely on the ability to reject points that are in collision. Testing collision between simple geometrical primitives, such as planes, spheres or cubes, is substantially faster than between non-convex polyhedra. On the other hand, if an object to be grasped is wrapped in a simple primitive bounding box –to achieve computational efficiency– many grasps would simply be impossible to achieve. Fig 2 shows the case of a grasp over the rim of a jug. This grasp requires the thumb to penetrate inside the the convex hull of object. Thus, even if this configuration does not produce any true collisions, it would be rejected by a "naive" collision detection.

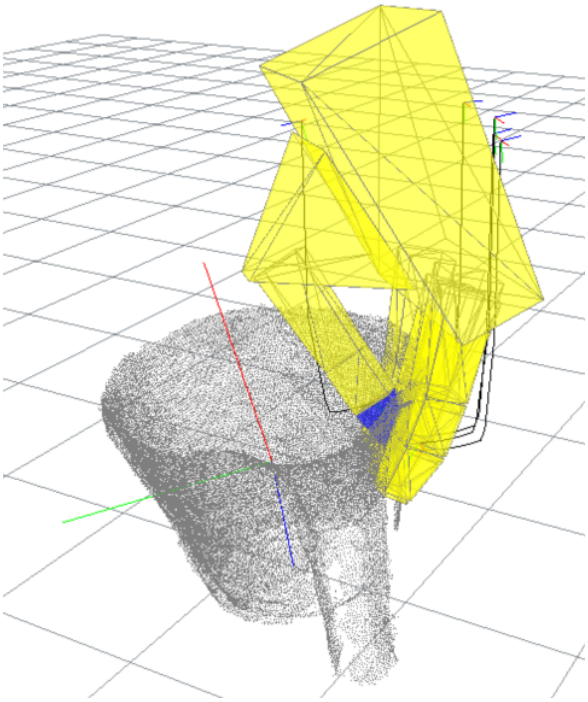


Fig. 2. Rim grasp example on a jug. In this case the target grasp requires to displace the robotic thumb on the internal surface of the object. The grasp configuration is computed using the method described in [**?**].

We propose a fast and efficient collision detection module to cope with object represented by point cloud. Whilst the robot's bodies are represented by a open-chain of convex polyhedra, the object to be grasped is represented by a 2-level structure. The first level of this structure wraps the point cloud in a bounding box. This level can efficiently avoid checking collisions between the robot's links and the object to be grasp that are far apart. The second level contains the point cloud organised in a KD-Tree. Our KD-Tree implementation is based on the FLANN library [Muja & Lowe, 2014]. An example is shown in Fig. 3.

In the planning process we target the object as if it were in its expected location (the mean pose of our density function), therefore only the mean pose is used for collision detection. When at least one bounding box of the robot collides with the bounding box of the object, the second level of the collision detection module is called to determine whether this particular configuration of the robot can be used in the PRM as either a node or in a path to connect two nodes. By querying the KD-Tree it is possible to retrieve the closest points on the surface of the object to the robot's links. Each point is then checked to determine whether it collides or not.

Critically the lack of information about the object's shape may affect collision detection, leading to a reach to grasp trajectory which passes trough the object. In a real scenario, however the tactile observation that will be generated will cause the robot to stop and the current belief state to update. In our current implementation we do not reason about the relative likelihood that the unexpected observation is given by a mis-estimation of the object pose versus lack of shape information; we simply update the object-pose uncertainty. As future work we would like to treat this problem as a SLAM problem, which will enable us to integrate the initial lack of shape information into our object model.

4.3. Planning a trajectory to maximise information gain

The PRM method comprises two phases: i) the learning phase, in which a connected graph G of obstacle-free configurations of the robot is generated and, ii) the query phase, in which a path is searched for a given pair of configurations x_{root}, x_{goal}.

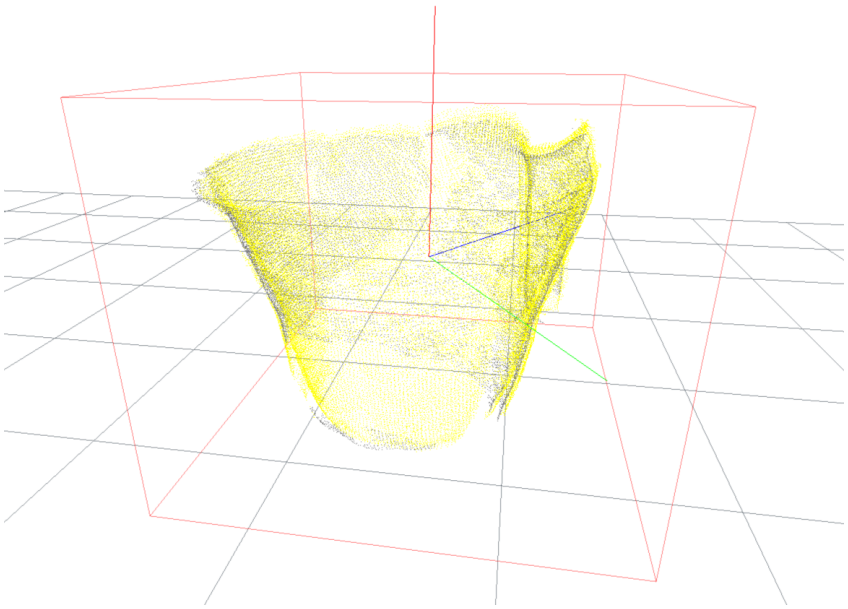


Fig. 3. The image shows the target object to be grasped. In this case the object is a jug. The grey point cloud represents the ground truth pose of the object, as it was acquired by a noiseless input source. The yellow point cloud identifies the best pose estimate. The yellow point cloud is organised as a KD-Tree for faster collision detections. The red box represents the bounding box which avoids unnecessary collision checking between the object and robot's links when they are far apart.

However the computational cost for the learning phase grows fast with respect to the dimensionality of the problem. We therefore incrementally build connections between neighbouring nodes during the query phase. Given a pair $\langle x_{root}, goal \rangle$ which describes the root state in configuration space, $\mathbb{R}^n$, and the goal state in the workspace, $SE(3)$, of the trajectory, we use the A* algorithm to find a minimum cost trajectory in obstacle-free joint space according to:

$$c(x) = c_1(x, x_{root}) + c_2(x, x', \hat{x}_{goal}) \tag{4}$$

where $x, x' \in \mathbb{R}^n$ and $x' \in Neighbour(x)$, $\hat{x}_{goal}$ is a reachable goal configuration for the robot computed by inverse kinematics, $c_1(x, x_{root})$ is the cost-to-reach x from x_{root}, and $c_2(x, x', \hat{x}_{goal})$ is a linear combination of the cost-to-go from x to a neighbouring node x' and the expected cost-to-go from x' to the target. We implemented $c_1(\cdot)$ as a cumulative discount and rewarded travelled distances. In more detail, we define

$$\begin{aligned} c_2(x, x', \hat{x}_{goal}) = \alpha d_{bound}(x, x') + \beta d(x', \hat{x}_{goal}) \\ + \gamma d_{cfg}(x) \end{aligned} \tag{5}$$

where $\alpha, \beta, \gamma \in \mathbb{R}$, $d(\cdot)$ is a distance function in $SE(3)$ which linearly combines rotational and transitional distances in the workspace[1]. For $d_{bound}(\cdot)$, let $\mathcal{B}_n(r) = \{x \in \mathbb{R}^n | x^T x \leq r^2\}$ and $\mathcal{B}(r_l, r_a) = \{A = \left[\begin{smallmatrix} R & p \\ 0 & 1 \end{smallmatrix}\right] \in SE(3) | p^T p \leq r_l^2$ and $1 - \langle Q(R), Q(R) \rangle \leq r_a^2\}$[2] denote repectively the r-ball in $\mathbb{R}^n$ and in $SE(3)$, then $b_{bound}(x, x')$ is defined as

$$d_{bound}(x, x') = \begin{cases} \psi(x, x') & \text{if } W(x) - W(x') \in \mathcal{B}(r_l, r_a) \\ & \text{and } x - x' \in \mathcal{B}_n(r) \\ +\infty & \text{otherwise} \end{cases}$$

[1] For the sake of simplicity, we abuse the mathematical notation by writing $d(x, x')$ instead of d(W(x),W(x')).
[2] We simplified the notation $\mathcal{B}_{SE(3)}(\cdot)$ in $\mathcal{B}(\cdot)$ for pratical reasons.

where $Q(\cdot)$ is the Quaternion operator for $R \in SO(3)$, $\langle q_1, q_2 \rangle$ is the inner product of two quaternions, $r_l, r \in \mathbb{R}$, $r_a \in (0,1)$, and $\psi(x,x') = \zeta d(x,x') + (1-\zeta)||x - x'||_2$ with $\zeta \in (0,1)$. Finally, $d_{cfg}(\cdot)$ is a function which penalises dangerous configurations of the robot (i.e. close to joint limits).

We redefine the heuristic $c_2(\cdot)$ in order to reward informative tactile explorations while attempting to reach the goal state (described as a target configuration of the manipulator).

$$\begin{aligned}\bar{c}_2(x, x', \hat{x}_{goal}, A, p^{1:k}) =& \alpha J(x, x', p^{1:k}) d_{bound}(x, x') \\ &+ \beta d_A(x', \hat{x}_{goal}) + \gamma d_{cfg}(x')\end{aligned} \tag{6}$$

where A is the diagonal covariance matrix of our sampled states, for any column vector $a, \mu \in \mathbb{R}^n$, $d_A(a,\mu) = \sqrt{(a-\mu)^T A^{-1}(a-\mu)}$ is the Mahalanobis distance centered in μ and $J(x, x', p^{1:k}) \in (0,1]$ is a factor which rewards trajectories with a large difference between expected observations if the object is at the expected location, p^1, versus observations that would be expected if the object is at other poses, $p^{2:k}$, sampled from the distribution of poses associated with the object's positional uncertainty:

$$\bar{J}(x, x', p^{1:k}) = \frac{1}{k-1} \sum_{i=2}^{k} e^{-\Phi(x,x',p^i)} \tag{7}$$

where:

$$\Phi(x, x', p^i) = ||h_t(x, x', p^i) - h_t(x, x', p^1)||_2$$

for each $i \in [2,k]$ and $h_t(x, x', p^i)$ is sequence of the probabilities of reading a contact travelling from state x to x'. In our implementation $h_t(x, x', p^i) = h(x', p^i)$. In other words, we evaluate the likelihood of making a contact while moving from state x to x' as the likelihood of making a contact only in the next state x'. Note that our current observational model is designed to conserve (6) as in (5) when the likelihood of observing a tactile contact is zero. In fact, for robot configurations in which the distance to the sampled poses is larger than a threshold, d_{max}, the cost function $J(\cdot)$ is equal to 1. However we also encode uncertainty in the second factor of our heuristics, $d_{\mathcal{A}}(\cdot)$, which evaluates the expected distance to the goal configuration. In this way the planner also copes with pose uncertainty at the early stages of the trajectory, when the robot is still too far away from the object to observe any contacts.

4.4. Planning for Dexterous manipulator

In order to compute a dexterous trajectory which allows us to plan movement for both arm and fingers we need to address the 'curse of dimensionality' or, equivalently, increase the number of sampled configurations to properly cover the configuration space.

Our proposed solution is to build a hierarchical planner. First we generate a PRM only in the arm configuration space in order to find a global path between the $x_{root}, \hat{x}_{goal}$. It is worth noticing that in this phase the rest of the joints of the manipulator are interpolated in order to have a smooth passage from x_{root} to $\hat{x}_{goal}$. We then refine the planned trajectory generating a new PRM in the entire configuration space of the manipulator (e.g. arm + hand joint space) along the global path. In other words, we limited the new PRM to explore only the subspace nearby the configurations which compose the global path. Subsequently an optimisation procedure is executed along the trajectory to generate a smoother transition from one configuration to the next.

This approach enables us to plan dexterous reach-to-grasp trajectories up to 21 DoFs with only 1,000 sampled configurations. Note that this is the same order of magnitude that we use in practice for planning trajectories of 6 DoF robot manipulators.

4.5. Belief update

Once a trajectory is executed and a real (unexpected) observation y is detected, we update our belief state according to Bayes' rule. We represent our belief state as a set of N particles $b_t = \{b_t^z\}_{z=1}^N$. In a particle filter fashion we update the weight of each particle b_t^z, $z \in \{1, \ldots, N\}$, as follows

$$p(y|x, b_t^z) = \prod_{j \in \hat{M}} p(y_t(j)|x_t(j), b_t^z)$$

and then we execute a resampling step which generates a posterior distribution b_{t+1} as new set of particles $\{b_{t+1}^z\}_{z=1}^N$.

In simulation we assume that there are no false dectections. However it is possible to distinguish whether or not a contact occurs between the object to be grasped and the robot's end-effector. For example, in case a contact with the environment is detected we skip the belief update step and we move the robot back to a safe configuration before triggering the re-planning.

5. Results

In this work we aim to show that sequential re-planning is capable of achieving higher grasp success rates than single grasp attempts in presence of non-Gaussian object-pose uncertainty in 6 dimensions. We also show that planning trajectories that maximise information gain requires fewer re-planning iterations to achieve a grasp.

We ran 45 trials in a virtual environment. Each trial has a different initial probability density over the object pose. We tested the ability of different strategies to achieve a grasp configuration. The algorithm has a model of the object to be grasped, as a dense point cloud. We collected the object model with a depth camera from 7 different views. These views have been pre-processed to generate a single point cloud of the object model. The pre-processing aligns the single view point clouds, registering and eliminating outlier points. We used the same object for all the trials. Fig. 2 shows the pre-processed point cloud of the plastic jug used as the object model. Using the procedure described in [Kopicki et al., 2014], we computed a possible grasp on the model.

We tested our algorithms under the hypothesis that the same object is displaced in a different position - but still in the dexterous workspace of the robot - and the robot has to attempt a grasp even if the new point cloud is not as dense as the model point cloud. In simulation, we achieve this by applying a rigid body transformation to the single-view point cloud to move it to a different location within the dexterous workspace of the manipulator. We tested our algorithm in three different conditions. In each condition we randomly selected respectively 1, 3 or 7 single-view point clouds from the 7 single-views that composed the model.

Fig 5 summarises the data collected in our experiments. First we computed, for each condition mention above, how much the combination of the single views covered the model surface. All the results are compared with respect to the model coverage in percentage terms. We ran three different algorithms:

- ELEMENTARY: an open-loop trajectory towards the expect object pose without re-planning.
- MYCROFT: a sequential re-planning algorithm without information gathering.
- IR3NE: a sequential re-planning algorithm with information gathering.

For each strategy we present their success rate in the sense of their ability to converging to a grasp. The top left chart in Fig. 5 shows that the sequential re-planning approaches always led to a grasp, while a single grasp attempt often fails in the presence of uncertainty.

We also show the error in the initial estimate of the object pose with respect to the ground truth, which is available in the simulation environment. We decomposed this error into translational and rotational components. The translational component measures displacement as Euclidean distance in a 3 dimensional space between the estimated location and the ground truth. The rotational or angular displacement is evaluated using quaternions. The bottom line of Fig. 5 shows

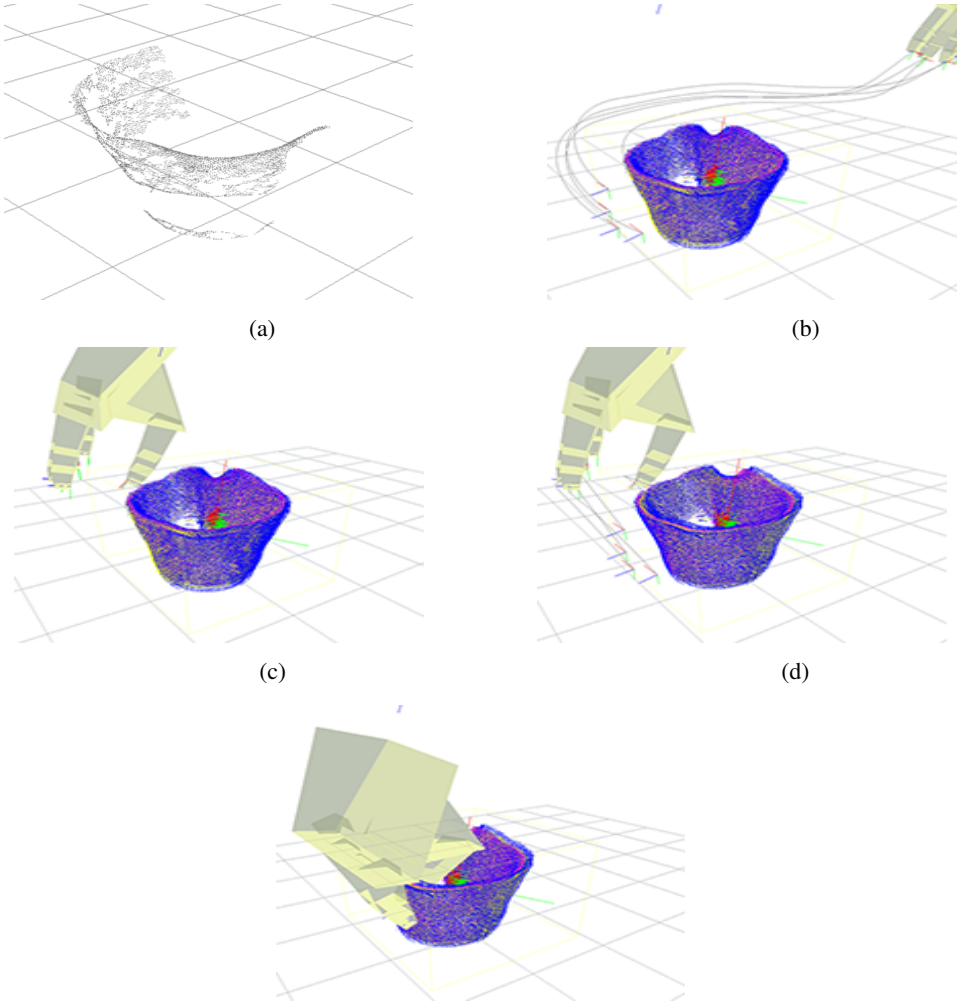


Fig. 4. Example of plan execution for IR3NE. Top row: a partial point cloud is acquired (a), a pose density represented by a set of particles is computed, and a reach-to-grasp trajectory is planned (b). The red point cloud identifies the ground truth which matches the partial point cloud in (a). The blue point clouds are the poses comprising the low dimensional belief state sub-sampled from the corresponding high dimensional belief state. The yellow point cloud represents the estimated pose for the object. Second row: despite the tiny misplacement between the estimated pose and the ground truth, a contact occurs before the grasp configuration with respect to the mean pose (yellow point cloud) could be reached (c). The belief state is therefore updated from the contact (d) and a new trajectory is planned with respect to the new estimate. Bottom row: the hand reaches the target grasp pose (d).

the error reduction, from the initial estimate to the final one, produced by using tactile information acquired during a failed attempt to grasp. Although both sequential re-planning approaches show a similar behaviour it is noticeable that the reach-to-grasp trajectories which maximise information gain (IR3NE) enable us to generate more informative contacts that lead to a better estimate with fewer iterations. Nevertheless, the bottom left chart in Fig. 5 shows that the rotational error seems to increase during the trials. There are two reasons for this: i) the initial rotational uncertainty in our trials was always very low since we chose an asymmetrically shaped object and ii) we stopped the robot at the first contact which most of the time is caused by only a finger, thus this information allows a better estimation in terms of linear accuracy but may reinforce hypotheses in the particle filter with a poor estimate of rotation.

To illustrate the behaviour of our re-planning system we show a typical sequence generated by one simulation trial. We assume a known point cloud model of the object shape, and uncertainty in the object pose. In this trial the robot observes the object as a point cloud and applies a model fitting process described in [Hillenbrand & Fuchs, 2011]. The model

fitting process is stochastic and so the resulting pose of the object is uncertain. We sample multiple poses by repeating this process, and obtain a belief state consisting of the resulting set of possible poses of the object. A trajectory is then planned to achieve a given grasp on the object. At each step a trajectory for the wrist and fingers are generated that will move to the desired grasp, while deviating from a minimum length trajectory to maximise information gathered through tactile observations. The belief state is updated and re-planning occurs each time a tactile contact is made. In the example shown in Fig. 4 the initial belief state over the object pose is quite narrow, however an error of few millimetres in the pose estimate leads to a potentially dangerous contact with the object. Instead, we use the contact to update the belief and plan a new trajectory from the current configuration of the robot. The second trajectory transfers successfully the hand to the target grasp configuration.

6. Conclusion and future work

We have shown how to solve the problem of dexterous grasping of objects with uncertain pose by using information gain re-planning. We have proposed a method for tactile information gain planning for dexterous, high DoF manipulators by showing how to: i) efficiently plan collision free, dexterous grasping trajectories for non-convex objects represented as a point cloud; ii) iteratively update localisation knowledge using tactile observations from a previous grasp attempt; iii) use successive grasp trajectories to plan with respect to these iteratively refined object poses; and iv) deliberately plan each reach-to-grasp trajectory to maximise new tactile information gain, while also moving toward the expected grasp location. We have shown that this approach enables planning for a robot manipulator with 20 DoF and non-Gaussian object pose uncertainty in 6 dimensions. We have also shown how sequential re-planning can achieve better quality grasps than single attempts to directly grasp an object at its expected pose, and that re-planning with trajectories designed to maximise tactile information gain, achieves successful grasps with fewer iterations than sequential attempts to move directly towards the object's expected pose.

This work extends that of [Platt et al., 2001], which offers a way to avoid the complexity of planning in a high dimensional belief space. It does this in two ways, i) by approximating the informational value of actions from a low-dimensional subspace of the belief state; and ii) by embedding that informational value into the physical space, creating a non-Euclidean distance metric in that space. This enables standard motion planning techniques to trade off directly between information gain and achievement of the goal pose for the manipulator. We aim to test our approach on a real robotic platform. This will require adaptation of our current observation model to cope with a robotic hand provided not only with current sensing at the joints, but also with force-torque sensors on the finger tips.

References

J. Betts (2001). *Pratical methods for optimal control using nonlinear programming*. Siam.

U. Hillenbrand & A. Fuchs (2011). 'An experimental study of four variants of pose clustering from dense range data'. In *Computer Vision and Image Understanding*.

K. Hsiao & L. Kaelbling (2010). 'Task-Driven Tactile Exploration'. In *IEEE Proc. of Robotics: Science and Systems (RSS)*.

K. Hsiao, et al. (2007). 'Grasping POMDPs'. In *IEEE Proc. Int. Conf. on Robotic and Automation (ICRA)*.

K. Hsiao, et al. (2011). 'Robust Grasping Under Object Pose Uncertainty'. In *Autonomous Robots*, no. 31 in (2-3), pp. 253–268.

D. Jacobson & D. Mayne (1970). *Differential dynamic programming*. Elsevier.

L. Kavraki & P. Svestka (1996). 'Probabilistic roadmaps for path planning in high-dimensional configuration spaces'. In *IEEE Trans. on Robotics and Automation*.

M. Kopicki, et al. (2014). 'Learning Dexterous Grasps That Generalise To Novel Objects By Combining Hand And Contact Models'. In *IEEE Proc. Int. Conf. on Robotic and Automation (ICRA)*.

S. M. LaValle (1998). 'Rapidly-exploring random trees: A new tool for path planning'. Tech. rep., Computer Science Dept, Iowa State University.

M. Muja & D. Lowe (2014). 'Scalable Nearest Neighbor Algorithms for High Dimensional Data'. *Pattern Analysis and Machine Intelligence (PAMI)* **36**:2227–2240.

E. Nikandrova, et al. (2013). 'Towards informative sensor-based grasp planning'. *Robotics and Autonomous Systems* pp. 340–354.

A. Petrovskaya & O. Khatib (2011). 'Global localization of objects via touch'. *IEEE Trans. on Robotics* **27**(3):569–585.

R. Platt, et al. (2001). 'Simultanious localization and grasping as a belief space control problem'. Tech. rep., CSAIL, MIT.

R. Platt, et al. (2012). 'Non-Gaussian Belief Space Planning: Correctness and Complexity'. In *IEEE Proc. Int. Conf. on Robotic and Automation (ICRA)*.

S. Prentice & N. Roy (2008). 'The belief roadmap: Efficient planning in linear POMDPs by factoring the covariance'. In *12th Int. Symposium of Robotics Research*.

C. Zito, et al. (2013). 'Sequential trajectory re-planning with tactile information gain for dextrous grasping under object-pose uncertainty'. In *IEEE Proc. Intelligent Robots and Systems (IROS)*.

C. Zito, et al. (2012). 'Exploratory reach-to-grasp trajectories for uncertain object poses.'. In *Proc. Workshop on Beyond Robot Grasping: Modern Approaches for Dynamic Manipulation. Intelligent Robots and Systems (IROS)*.

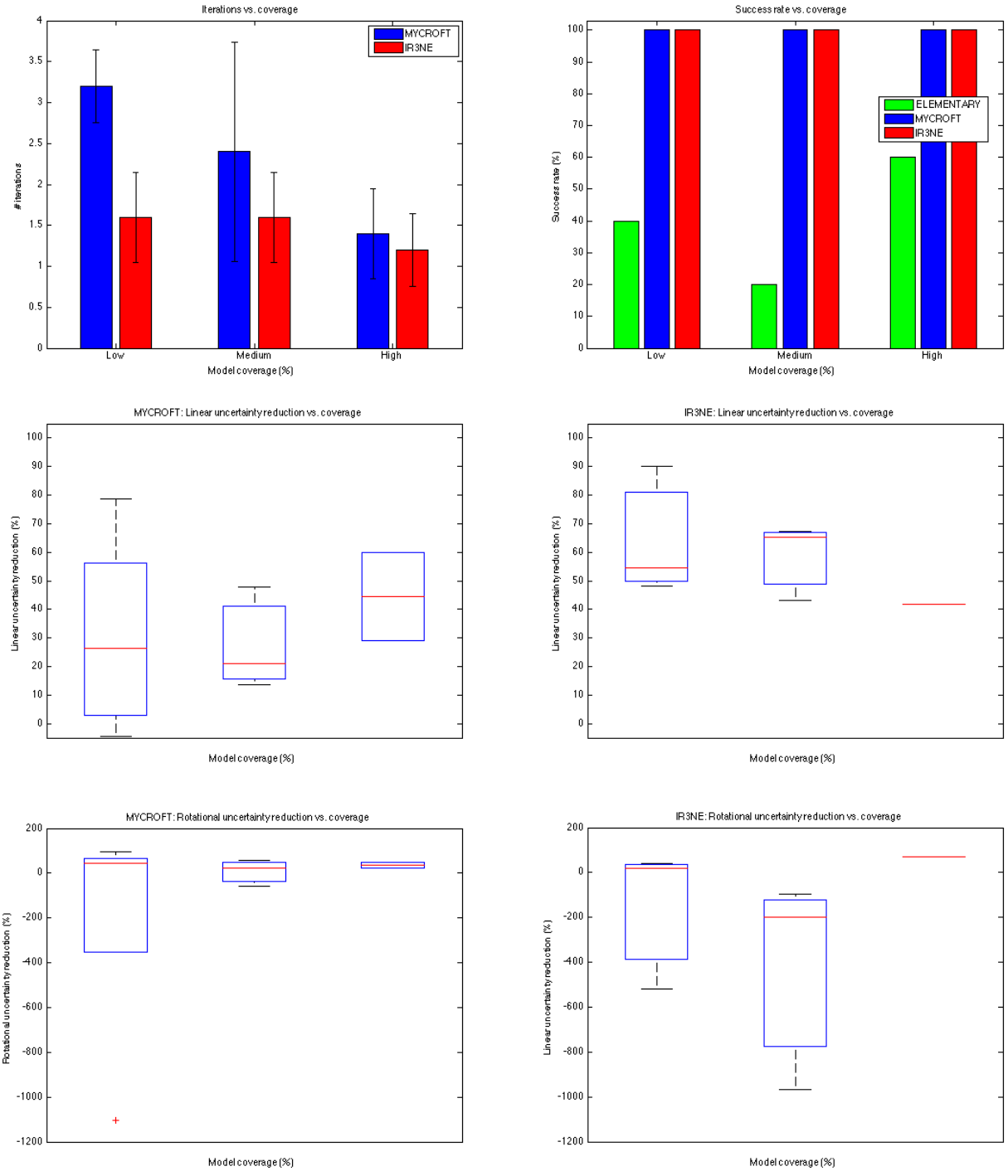


Fig. 5. Simulated results for 45 trials and three strategies: ELEMENTARY (green), MYCROFT (blue), IR3NE (red). All the results are presented plotted against the initial percentage of model coverage given by one or more views by the depth camera. The first chart (top left) presents the average number of iterations required to converge to a grasp. IR3NE shows that maximising information gain requires fewer re-planning iterations to achieve a grasp. The second chart (top right) shows that sequential re-planning is capable of achieving higher success rates than single grasp attempts in presence of uncertainty. The third chart (bottom left) shows how the initial linear uncertainty is reduced by sequential re-planning until the algorithm converges to a grasp. Trajectories that maximise information gain (IR3NE) are capable of localising the objects with a better accuracy. Finally, the last chart (bottom right) presents the reduction in the rotational uncertainty. In this case simple re-planning (MYCROFT) shows better performance, however the rotational uncertainty is usually low for asymmetric objects like the object we used for the experiments.

A.5 Article: Sample-based motion planning for soft robot manipulators under task constraints

Authors M. Bonilla, E. Farnioli, L. Pallottino, A. Bicchi

Info Accepted for publication at the 2015 IEEE Intl. Conf. on Robotics and Automation

Abstract Random sampling-based methods for motion planning of constrained robot manipulators has been widely studied in recent years. The main problem to deal with is the lack of an explicit parametrization of the non linear submanifold in the Configuration Space (CS), due to the constraints imposed by the system. Most of the proposed planning methods use projections to generate valid configurations of the system slowing the planning process. Recently, new robot mechanism includes compliance either in the structure or in the controllers. In this kind of robot most of the times the planned trajectories are not executed exactly by the robots due to uncertainties in the environment. Indeed, controller references are generated such that the constraint is violated to indirectly generate forces during interactions. In this paper we take advantage of the compliance of the system to relax the geometric constraint imposed by the task, mainly to avoid projections. The relaxed constraint is then used in a state-of-the-art sub-optimal random sampling based technique to generate any-time paths for constrained robot manipulators. As a consequence of relaxation, contact forces acting on the constraint change from configuration to configuration during the planned path. Those forces can be regulated using a proper controller that takes advantage of the geometric decoupling of the subspaces describing constrained rigid-body motions of the mechanism and the controllable forces.

Relation with the deliverable planning dual arm manipulation for soft robots.

Attachment (following pages until next annex)

Sample-Based Motion planning for Soft Robot Manipulators Under Task Constraints

M. Bonilla[1], E. Farnioli[1,2], L. Pallottino[1], A. Bicchi[1,2]

***Abstract*— Random sampling-based methods for motion planning of constrained robot manipulators has been widely studied in recent years. The main problem to deal with is the lack of an explicit parametrization of the non linear submanifold in the Configuration Space ($\mathcal{CS}$), due to the constraints imposed by the system. Most of the proposed planning methods use projections to generate valid configurations of the system slowing the planning process.**

Recently, new robot mechanism includes compliance either in the structure or in the controllers. In this kind of robot most of the times the planned trajectories are not executed exactly by the robots due to uncertainties in the environment. Indeed, controller references are generated such that the constraint is violated to indirectly generate forces during interactions.

In this paper we take advantage of the compliance of the system to *relax* the geometric constraint imposed by the task, mainly to avoid projections. The relaxed constraint is then used in a state-of-the-art sub-optimal random sampling based technique to generate any-time paths for constrained robot manipulators. As a consequence of relaxation, contact forces acting on the constraint change from configuration to configuration during the planned path. Those forces can be regulated using a proper controller that takes advantage of the geometric decoupling of the subspaces describing constrained rigid-body motions of the mechanism and the controllable forces.

I. Introduction

In robot motion planning, interacting with the environment is normally considered a task to avoid, however in everyday tasks humans don't do that. Actually, simple tasks as opening a door, sliding an object on a table and moving an object consist on taking advantage of the objects and their constraints with the environment rather than avoiding touching them. In robotics solving the problem of generating motions is not simple mainly because we need to face two main problems: 1) working on high dimensional spaces, which make the problem NP-Hard to solve it optimally, and 2) working under constraints such as closed loop kinematic chains and force/torque limits. The first, is solved in an efficient way randomly sampling the configuration space ($\mathcal{CS}$) of the robot. This is possible thanks to the available explicit description of the $\mathcal{CS}$ of the robot. The second problem is harder due the fact that an explicit description of the admissible $\mathcal{CS}$ is not available. It means that not all random samples of the $\mathcal{CS}$ can be considered as a possible configuration to explore. There exist some approaches to generate motions for robots under environmental constraints, which are based either on the decomposition of the chain in a passive and an active chain [1], or in the projection of any random sample to the valid $\mathcal{CS}$ [2], [3].

Fig. 1. The Motion planning method presented in this paper is developed for systems with compliance either in their structure of via the control loop. In this case the bimanual system includes the compliance in the qbmotors [9] composing the structure which is combined with mechanically embedded compliance of the PISA/IIT SofHands [10]. In this paper we consider that all compliance is in the contact points.

In this paper we propose a new method to generate motions for kinematic chains under constraints. It is based on the relaxation of the constraint to be able to randomly sample an augmented valid $\mathcal{CS}$. Then, using state-of-the-art algorithms as RRT* [4], we guarantee the convergence of the algorithm to a path, connecting two points, which optimizes the distance to the constraint at each point on it.

A. Planning with Task Constraints - State of the Art

Since the introduction of random sampling techniques for path planning, a lot of advances have been made in this field. There exist two main approaches in this topic, the first is the Probabilistic Roadmap (PRM) and the second is the Rapidly-Exploring Random Tree (RRT) introduced in [5] and [6] respectively. These two approaches were designed to plan motions in high dimensional spaces, in fact they are normally applied in the $\mathcal{CS}$ of robot manipulators. The major advances have been focused in the improvement of these methods to mainly include heuristics to speed up the planning time and bias the solutions to get preferred behaviours. For example in [7] exploration and exploitation of $\mathcal{CS}$ is balanced for fast convergence of the planners. In [8] the authors propose to include different heuristics to bias the growth of the trees towards a preferred part in the $\mathcal{CS}$.

The last major contribution in probabilistic motion planing

[1]Research Center "E. Piaggio", Università di Pisa, Largo Lucio Lazzarino 1, 56122 Pisa, Italy
[2]Department of Advanced Robotics, Istituto Italiano di Tecnologia, Via Morego 30, 16163 Genova, Italy

Preprint submitted to 2015 IEEE International Conference on Robotics and Automation. Received October 2, 2014.

was presented in [4] where the authors studied the quality of the paths generated by randomized planners. They proposed a modification of the RRT and PRM algorithms, called RRT* and PRM*, to generate better quality paths. The completeness and sub-optimality of the solutions are guaranteed. Some improvements to speed up the solutions of this planner have been proposed in [11] and [12].

The inclusion of constraints in the model is another research line in the area, for example 1) *nonholonomic constraints* for mobile robots as summarized in [13], 2) *task constraints* where the end effector has to maintain a desired orientation over the whole planned path, [14], and 3) *close kinematic chains* for cooperative robots or parallel manipulators [15]. The latter is sometimes considered a particular case of 2). There is also a research line to include *dynamic constraints* such as joint torque limits. This planning techniques are called kinodynamic motion planning [16]. In this work we will focus the attention to motion planing for systems with tasks space constraints and close kinematic chains.

The main problem in motion planning for closed kinematic chains is that the $\mathcal{CS}$ of the robot is not all available to be explored but just a nonlinear submanifold $\mathbb{M}$, described by the constraint equations, living in it.

All randomized planners include a function called *Sample* where a random point in the configuration space is returned. In case of closed kinematic chains the function *Sample* must return a random point on the aforementioned submanifold. The practical probability of this is 0 because the manifold is a zero measure set in the configuration space.

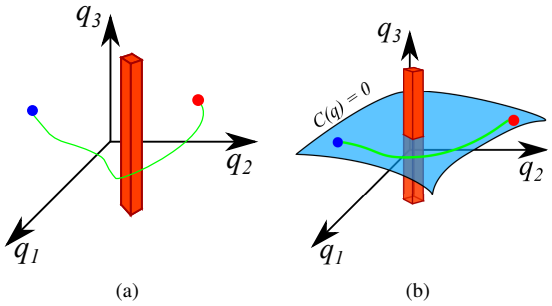
The proposal presented in this paper is to relax the constraints by transforming this set into a volume so that the probability of sampling a point randomly on it is not null. This idea comes from the state of the art robots, see Fig. 1, which have a compliant rather than a rigid structure. From the point of view of path following for systems with compliance is traduced to control references which may or may not be followed by the real system but they are still valid. From this viewpoint, references can violate the constraints imposed by the system structure and by its interaction with the environment.

B. Path Execution

The path execution problem can be addressed with a suitable force/position controller using the theory presented in [17]. In that paper, the authors demonstrate that the object trajectories and the contact forces can be addressed as decoupled control problems. It implies that we can execute any object trajectory coming from planning phase and that contact forces can be steered so as to avoid violation of contact constraints, allowing to regulate a desired force during motion.

II. Organization

Section III formally defines the motion planning problem under task constraints and presents the main contribution of this work. In section IV the algorithm called $soft-RRT^*$ is presented, it describes the strategy implemented to find paths in the relaxed constraint. Section V addresses the problem of the practical implementations of the $soft-RRT^*$ algorithm and introduces a possible solution. After an example presented in section VI, section VII exposes the conclusions and future developments of this work.



(a) (b)

Fig. 2. Differences of motion planning problem without constraints (a) and with constraints (b). Initial position q_{init} in blue, final position q_{final} in red and Planned path in green. Constraint $C(q) = 0$ in baby blue.

III. Problem Definition

Random sampling-based methods for path planning have an excellent performance when they are able to explore the whole $\mathcal{CS}$ of robot manipulators. The good performance comes from the fact that manipulators are able to explore the whole environment performing any kind of motions in any directions in the $\mathcal{CS}$, so any point in it is a valid configuration and can be connected to any other one. However when the system is subject to constraints, such as closed kinematic chains, this fact is not true any more.

In this section we formally introduce the motion planning problem for systems subject to constraints.

A. Motion Planning Problem of Systems Under Constraints

Consider a configuration space $\mathcal{M} \in \mathbb{R}^d$ that is a compact set of configurations q. Let $\mathcal{O} \in \mathcal{M}$ be the obstacle region and $\mathcal{M}_{free} := \mathcal{M} \setminus \mathcal{O}$ the configuration set free of obstacles. Introducing a kinematic constraint $C(q) = 0$ that limits the robot configurations and hence motion, see Fig. 2(b), we define a nonlinear submanifold in $\mathcal{M}$ as $\mathcal{M}_v := \{q : q \in \mathcal{M}_{free}, C(q) = 0\}$ to describe all configuration where non of the links of the mechanism collide neither with objects in the environment not with other links, and satisfy the constraint. The motion planning problem is to find a continuous path $\sigma : [0, 1] \rightarrow \mathcal{M}_v$; with $\{\sigma(0) = q_{init},\ \sigma(1) = q_{final}\}$. As mentioned, the main challenge in applying sampling based motion planning algorithms to closed kinematic chains is that the probability of getting a random point laying on the submanifold is zero, see Fig. 2(b).

B. Relaxing Constraints

As mentioned in the introduction we consider systems with compliance. In this paper we introduce compliance in the planning phase as a parameter to relax the constraint, so $C(q) \leq \epsilon$, using this approach the rigid kinematic constraint is replaced with a compliant one. In the case in which the

Preprint submitted to 2015 IEEE International Conference
on Robotics and Automation. Received October 2, 2014.

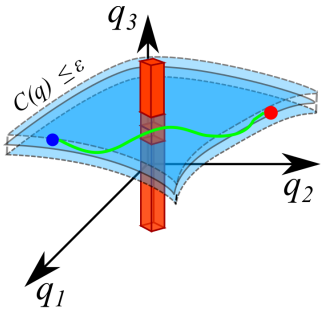


Fig. 3. Motion planning problem under relaxed constraints. Initial position q_{init} in blue. Final position q_{final} in red. Planned path in green. Constraint $C(q)$ in baby blue.

tight constraint is violated a proportional force f_h arises between the two parts in contact. With this parameter the sub-manifold describing the relaxed constraint can be considered as a space with the same dimension of $\mathcal{CS}$. Thanks to this we can use rejection techniques to randomly sample the $\mathcal{CS}$ valid, now defined as $\mathcal{M}_r := \{q : q \in \mathcal{M}_{free}, C(q) = \epsilon\}$, and thus speed up the planning process. Now the planning problem is to find a continuous path $\sigma : [0, 1] \rightarrow \mathcal{M}_r$, with, $\{\sigma(0) = q_{init},\ \sigma(1) = q_{final}\}$. This relaxed problem is graphically described in Fig. 3.

IV. Randomized Planning Algorithm

The random based-sampling algorithm used in this paper is the $soft-RRT^*$ reported in the algorithm 1. The difference with the original RRT* algorithm is that instead of just checking for collision we also check if the new configuration is inside the relaxed constraint. This is performed in the

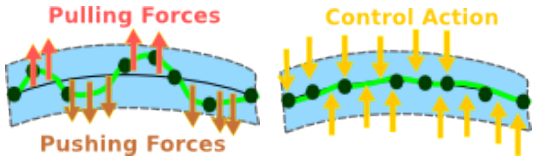
Algorithm 1 $\mathcal{T} = (V, E) \leftarrow$ soft-RRT*(x_{init})

```
T ← InitTree();
for i = 1 to N do
    x_rand ← Sample(i);
    x_nearest ← Nearest(V, x_rand);
    x_new ← Steer(x_nearest, x_rand);
    if Constraints(X_nearesr, x_new) then
        X_near ← Near(T, x_new)
        x_min ← BestParent(X_near, x_new)
        T ← T ∪ (x_new, x_min)
        T ← Rewire(T, X_near, x_new)
    end if
end for
return G = (V, E).
```

Constraints function where the constrained optimization problem described in the subsection IV-B is solved. Notice that this function is also called inside the **BestParent** and **Rewire** functions, typical of RRT*.

A. Biased Random Sampling

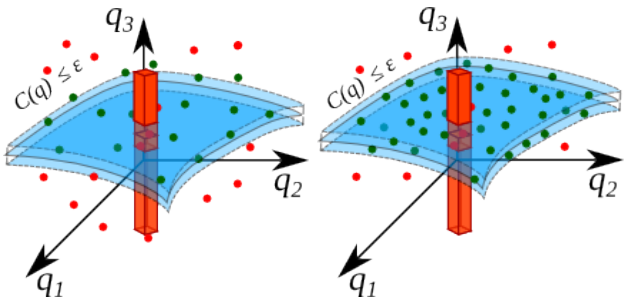
The first step in randomized path planners is perfomed in the function **Sample** and it consists on generating a new sample in $\mathcal{M}$. Typically, random configurations are taken



(a) Undesired forces arising from planning on the relaxed contraint

(b) The task of the controller is to project the undesired forces back to the manifod

Fig. 4. Lateral view of the relaxed constraint. In green are the pushing and pulling forces against the constraint. The black dots are the nodes extracted from the tree generated by the $soft - RRT^*$.



(a) (b)

Fig. 5. Graphical explanation of the difference of using uniform distribution a) and applying the algorithms presesnted in [22] b) to get new random sampling configuration in $\mathcal{M}$.

using a uniform distribution to explore equally all regions in $\mathcal{CS}$. Doing the same in our problem, the probability of getting a new point in $\mathcal{M}_r$ can be computed as

$$\rho = volume(\mathcal{M}_r)/volume(\mathcal{M}), \tag{1}$$

where the volume of $\mathcal{M}$ is defined by the mechanism, more precisely by the range of motion of all joints. In the other hand, the volume $\mathcal{M}_r$ is proportionally to the relaxing parameter ϵ. As a consequence, the probability of getting a new point goes to 0 as ϵ approaches 0, in other words it means that bigger is ϵ, higher the probability of getting a new configuration in $\mathcal{M}_r$.

It is evident that if ϵ is small most of the new samples will be rejected in the **Constraints** function because they are not in the relaxed constraint. To minimize the impact of this fact, in the $soft - RRT^*$ we used the algorithm presented in [22] to bias the random sampling procedure to converge to a uniform distribution not in $\mathcal{M}$ but in $\mathcal{M}_r$. This idea is graphically presented in the Fig. 5.

B. The Equilibrium Manifold

In order to guarantee the equilibrium of the object being manipulated by the multi-robot system we recall the kineostatic analysis presented in our previous work [18]. In that work we presented the equilibrium manifold of the multi-robot system subject to synergistic underactuation and variable stiffness in the joint actuation. Since we are performing motion planning in the $\mathcal{CS}$, we need a fully actuated system to allow to the system to follow the planned

path. Thus, the main equations describing the equilibrium configuration of the system are:

$$w + G(u)f_h = 0, \quad (2)$$
$$\tau - J^T(q,u)f_h = 0, \quad (3)$$
$$f_h - K_c p_h = 0, \quad (4)$$
$$\tau - K_q(q_r - q) = 0, \quad (5)$$

where $x \in \mathbb{R}^{2d+12}$ is an equilibrium residual vector containing the joint positions $q \in \mathbb{R}^d$, joint torques $\tau \in \mathbb{R}^d$, contact forces $f_h \in \mathbb{R}^6$ and object positions $u \in \mathbb{R}^6$. $\Phi(x) \in \mathbb{R}^{2d+12}$ includes the equations describing the equilibrium manifold. $G(u)$ is the so called grasp matrix of the system, $w \in \mathbb{R}^6$ is the external wrench in the object, $J^T \in \mathbb{R}^{c\times d}$ is the multi-robot Jacobian matrix, $K_{pc} \in \mathbb{R}^{c\times c}$ is the contact stiffness matrix and $K_q \in \mathbb{R}^{d\times d}$ is the joint stiffness matrix. Dimension c is the number of contact constraints.

Equations (2) to (5) describe the equilibrium manifold in the system that by adding quotes into the contact forces f_h becomes the same as the one used in the planning phase. In order to find equilibrium configurations we firstly get the random sampling configurations q_r and then we solve the following optimization problem to the get the rest of the variables

$$\min_{q,u,f_h,\tau} \quad x^T \mathbf{W} x \quad \text{subject to} \quad \Phi(x,q,u,f_h,\tau) = 0, \quad (6)$$

Once the planning problem is solved a proper controller able to let the robot follow the planned path must be determined. The main challenge comes from the fact that the closed kinematic constraint has been relaxed, so undesired contact forces arise from interactions, a graphical example is shown in Fig. 4(a). The real-time controller must ensure that the nominal constraint is satisfied during the whole execution, see Fig. 4(b). Indeed, if only the relaxed constraint is verified the object handled by the robot does not fall but can be damaged since high squeezing forces can appear. On the other hand, whenever the nominal closed kinematic constraint is verified this can not occur.

V. Execution of the Planned Path

The problem that arise when relaxing constraints is that from the point of view of implementation, the constraint violation can be dangerous, see Fig. 6, since undesirable interaction forces may be indirectly induced into the system. In this section we introduce a control to solve this problem.

A. Control

In order to address the problem of regulating contact forces and, at the same time, executing the planned path, a force/position controller can be implemented. There are many control approaches to do that, for example in [19] an adaptive hybrid control scheme for multiple geometric constraints based on the joint-space orthogonalization method (JSOM) is proposed, in [20] the authors propose a general framework for multi-contact motion/force control. In both

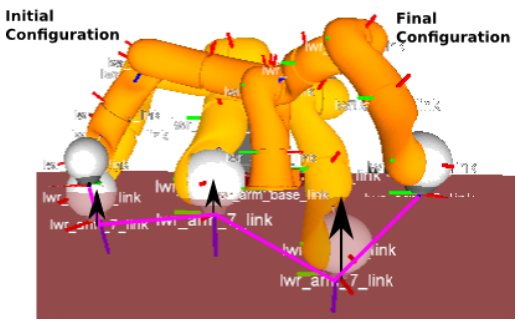


Fig. 6. In this example the kuka robot has to move from the initial to the final configuration maintaining the contact with the plane (red object). The resulting path from applying the $soft - RRT^*$ is shown in pink and the forces during motions in black arrows.

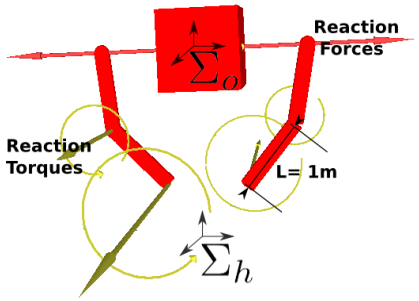


Fig. 7. The two finger hand used for the presented example. In this pictures the object position u, the fingers configurations q, the joint torques τ and contact forces f_h are expressed graphically.

cases the main considerations is that contacts are performed with rigid environments. However, new robot developments, like Soft Robots, are designed to work in uncertain environments and compliant task spaces. A general analysis of manipulation systems with general kinematics and compliant contact models is presented in [17] and complemented in [21], the main result of the last two contributions is a geometric description and an algorithm to provide a basis to describe the feasible motions that can be executed by the system, and forces that can be controlled to avoid violation of the contact constraints, both in a decoupled way. In practice it means that it is possible to control all object displacements given a fixed force reference and vice versa, where the first is useful to correct the relaxations in the planning phase.

VI. Simulations

In this section we show some simulations of the motion planning method presented in this paper. As an example we consider a two finger planar hand with two degrees of freedom in each finger, see Fig. 7.

This systems has 7 degrees of freedom in total but the $\mathcal{CS}$ for planning purposes is of dimension 4 since we are not sampling the object configurations. Algorithms have been implemented in C++ and use ROS for visualization purposes. All tests were performed in a $2.4Ghz$ quad-core computer with $3Gb$ of RAM memory and Ubuntu 14.04

Preprint submitted to 2015 IEEE International Conference
on Robotics and Automation. Received October 2, 2014.

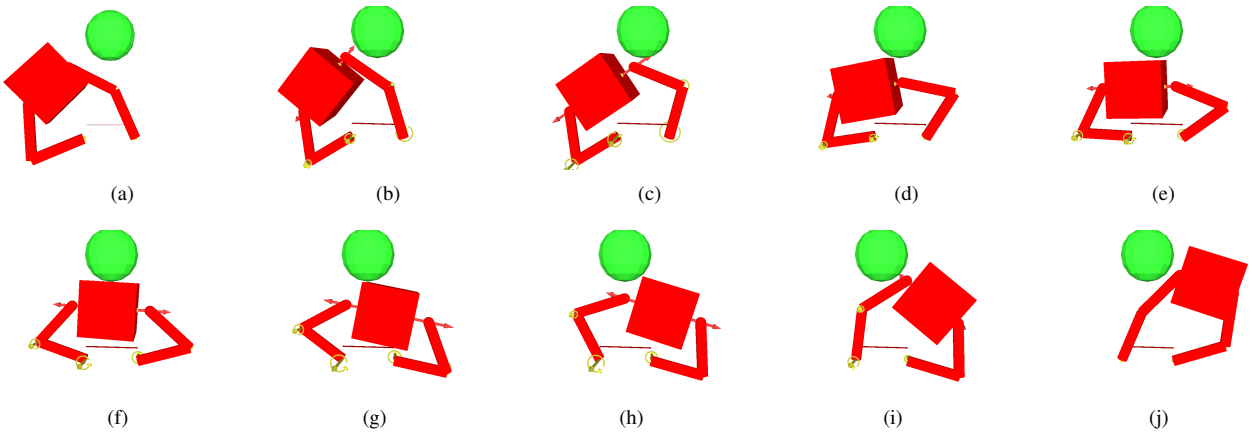


Fig. 9. Final path from the presented experiment. a) Initial position and b) final position.

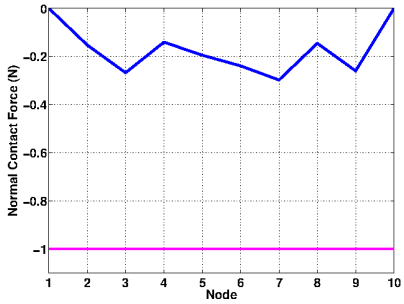


Fig. 8. Normal contact forces (blue) resulting from the relaxation of the constraint during planning. The maximum allowed forces ϵ are in magenta.

operative system. Fig. 9(a) shows the starting position and Fig. 9(j) shows goal position of the hand, the objective is to find a path connecting this two points avoiding the spherical obstacle in green and maintaining the contact forces within ϵ. Figs. 9(b) to 9(i) show some snapshots of the planned path resulting from the execution of algorithm 1, we can observe how the hand avoids the obstacle. Interaction forces arising from the planning phase, which in the case of the 2D example presented in this section are normal to the contact constraints and of magnitude proportional to ϵ, are shown in Fig. 8. Notice that the relaxation parameter ϵ is never overtaken.

VII. Conclusions

In this paper we propose a motion planning method for Soft Robots moving with task constraints. The approach consists in the combination of constraint relaxation and random sampling sub-optimal planner. The first one helps to speed up the planning phase considering the closed loop imposed in the system because of the interaction of the manipulators and the object. The second one allows us to explore the complete configuration space of the system and, at the same time, take into account optimality of the planned trajectories. Combining the first two strategies we are able to fast plan motions for multiple robot manipulators working cooperatively, however due to the constraint relaxation interaction forces my appear during executions of the planned path. To deal with this, as a future work we can implement a control strategy, as the ones presented in V, to online regulate the contact forces while executing trajectories coming from planning phase.

Acknowledgments

This work is supported by the EC under the CP-IP grant no. 600918 “PaCMan”, within the FP7-ICT-2011-9 program “Cognitive Systems”, ERC Advanced Grant no. 291166 “SoftHands” - A Theory of Soft Synergies for a New Generation of Artificial Hands-, under grant agreements no.611832 “Walk-Man” and by CONACYT through the scholarship 266745/215873.

References

[1] J. Cortes, T. Simeon, and J.-P. Laumond, “A random loop generator for planning the motions of closed kinematic chains using PRM methods,” in *Robotics and Automation, 2002. Proceedings. ICRA ’02. IEEE International Conference on*, vol. 2, pp. 2141–2146 vol.2, 2002.

[2] M. Stilman, “Global manipulation planning in robot joint space with task constraints,” *Robotics, IEEE Transactions on*, vol. 26, no. 3, pp. 576–584, 2010.

[3] D. Berenson, *Constrained Manipulation Planning*. PhD thesis, 2011.

[4] S. Karaman and E. Frazzoli, “Incremental sampling-based algorithms for optimal motion planning,” *CoRR*, vol. abs/1005.0416, 2010.

[5] L. Kavraki, P. Svestka, J.-C. Latombe, and M. Overmars, “Probabilistic roadmaps for path planning in high-dimensional configuration spaces,” *Robotics and Automation, IEEE Transactions on*, vol. 12, no. 4, pp. 566–580, 1996.

[6] S. M. Lavalle, “Rapidly-exploring random trees: A new tool for path planning,” tech. rep., 1998.

[7] M. Rickert, O. Brock, and A. Knoll, “Balancing exploration and exploitation in motion planning,” in *Robotics and Automation, 2008. ICRA 2008. IEEE International Conference on*, pp. 2812–2817, 2008.

[8] C. Urmson and R. Simmons, “Approaches for heuristically biasing RRT growth,” in *Intelligent Robots and Systems, 2003. (IROS 2003). Proceedings. 2003 IEEE/RSJ International Conference on*, vol. 2, pp. 1178–1183 vol.2, 2003.

[9] M. G. Catalano, G. Grioli, M. Garabini, F. Bonomo, M. Mancini, N. G. Tsagarakis, and A. Bicchi, “Vsa - cubebot. a modular variable stiffness platform for multi degrees of freedom systems,” in *2011 IEEE International Conference on Robotics and Automation*, (Shangai, China), pp. 5090 – 5095, May 9 - 13 2011.

[10] M. G. Catalano, G. Grioli, E. Farnioli, A. Serio, C. Piazza, and A. Bicchi, "Adaptive synergies for the design and control of the pisa/iit softhand," *International Journal of Robotics Research*, vol. 33, p. 768–782, 2014.

[11] B. Akgun and M. Stilman, "Sampling heuristics for optimal motion planning in high dimensions," in *Intelligent Robots and Systems (IROS), 2011 IEEE/RSJ International Conference on*, p. 26402645, 2011.

[12] O. Salzman and D. Halperin, "Asymptotically near-optimal RRT for fast, high-quality, motion planning," *arXiv preprint arXiv:1308.0189*, 2013.

[13] S. M. Lavalle, J. J. Kuffner, and Jr, "Rapidly-exploring random trees: Progress and prospects," in *Algorithmic and Computational Robotics: New Directions*, p. 293308, 2000.

[14] M. Stilman, "Task constrained motion planning in robot joint space," in *Intelligent Robots and Systems, 2007. IROS 2007. IEEE/RSJ International Conference on*, p. 30743081, 2007.

[15] J. Corts and T. Simon, "Sampling-based motion planning under kinematic loop-closure constraints," in *Algorithmic Foundations of Robotics VI*, p. 7590, Springer, 2005.

[16] A. Masoud, "Kinodynamic motion planning," *IEEE Robotics & Automation Magazine*, vol. 17, pp. 85–99, Mar. 2010.

[17] D. Prattichizzo and A. Bicchi, "Consistent task specification for manipulation systems with general kinematics," *in ASME Journal of Dynnamic System Measurements and Control. Accepted*, pp. 760–767, 1997.

[18] E. Farnioli, M. Gabiccini, M. Bonilla, and A. Bicchi, "Grasp compliance regulation in synergistically controlled robotic hands with vsa," in *IEEE/RSJ International Conference on Intelligent Robots and Systems, IROS 2013*, (Tokyo, Japan), pp. 3015 –3022, November 3-7 2013.

[19] Y.-H. Liu, V. Parra-Vega, and S. Arimoto, "Decentralized cooperation control: joint-space approaches for holonomic cooperation," in *Robotics and Automation, 1996. Proceedings., 1996 IEEE International Conference on*, vol. 3, pp. 2420–2425 vol.3, Apr 1996.

[20] R. Featherstone, S. Thiebaut, and O. Khatib, "A general contact model for dynamically-decoupled force/motion control," in *Robotics and Automation, 1999. Proceedings. 1999 IEEE International Conference on*, vol. 4, pp. 3281–3286 vol.4, 1999.

[21] D. Prattichizzo and A. Bicchi, "Dynamic analysis of mobility and graspability of general manipulation systems," *IEEE Transactions on Robotics and Automation*, vol. 14, pp. 241–258, 1998.

[22] J. Bialkowski, M. Otte, and E. Frazzoli, "Free-configuration biased sampling for motion planning," in *Intelligent Robots and Systems (IROS), 2013 IEEE/RSJ International Conference on*, pp. 1272–1279, 2013.

A.6 Article: High-Level Planning for Dual Arm Goal-Oriented Tasks

Authors H. Marino, M. Ferrati, A. Settimi, C. Rosales, M. Gabiccini

Info Submitted to the 2015 IEEE/RSJ Int. Conf. on Intelligent Robots and Systems

Abstract In this paper, we design a simple methodology for high-level planning, corroborated with low-level planning and execution, to solve a task having specified a priori a set of interaction primitives. We provide a realistic implementation focusing on the simple task of moving an object from a given initial to a desired final configuration, possibly exploiting the interaction between the object and the tabletop, as well as direct handoff between two hands when convenient. The interaction primitives are the different ways in which the object can be grasped and moved around, possibly different for each end-effector, and the planning step is carried out considering the table itself as an end-effector (with the non-movable attribute) with its own set of primitives. This results in a sequence of actions that are then translated into collision-free paths and, subsequently, low level commands for the robot. Experimental results with a bi-manual robotic platform show the viability of our approach, which can be well extended to multi-arm systems.

Relation with the deliverable passing objects from one hand to another (Task 5.3)

Attachment (following pages until next annex)

High-Level Planning for Dual Arm Object Passing Tasks

Hamal Marino, Mirko Ferrati, Alessandro Settimi, Carlos Rosales, and Marco Gabiccini

***Abstract*— In this paper, we design a simple methodology for high-level planning, corroborated with low-level planning and execution, to solve a task having specified a priori a set of *interaction primitives*. We provide a realistic implementation focusing on the simple task of moving an object from a given initial to a desired final configuration, possibly exploiting the interaction between the object and the tabletop, as well as direct handoff between two hands when convenient. The interaction primitives are the different ways in which the object can be grasped and moved around, possibly different for each end-effector, and the planning step is carried out considering the table itself as an end-effector (with the *non-movable* attribute) with its own set of primitives. This results in a sequence of actions that are then translated into collision-free paths and, subsequently, low level commands for the robot. Experimental results with a bi-manual robotic platform show the viability of our approach, which can be well extended to multi-arm systems.**

Keywords: Dual Arm Manipulation; High-Level Planning; Bimanual Object Passing; Handoff

I. Introduction

In recent years, many approaches have been developed to allow a robot to execute a task by giving only high-level commands, and plan both the sequence of actions needed to complete the task, as well as the low-level sequence of movements to perform each single action.

In [1] a couple of robots demonstrates the capability of translating a natural language plan for making pancakes (found on the web), into a sequence of movements: one robot goes around the kitchen looking for ingredients, while the other is fixed at a table and prepares the pancake as soon as everything is placed inside its reachable workspace by the first robot.

In [2], a long-term planning strategy is devised such that, when used to displace an object, it also reasons about moving obstacles out of the way.

In [3], predefined grasp primitives are attached to objects which implement functional categories in a framework similar to object-specific reasoning introduced earlier in [4]; it uses PDDL language [5] and a semantic planner to decompose a given task into a series of subtasks.

H. Marino and A. Settimi are with the Dep. of Advanced Robotics, Istituto Italiano di Tecnologia, via Morego, 30, 16163 Genova, and with the Res. Center "E. Piaggio", University of Pisa, 56122 Pisa, Italy (hamal.marino@centropiaggio.unipi.it, alessandro.settimi@for.unipi.it)

M. Ferrati and C. Rosales are with the Res. Center "E. Piaggio", University of Pisa, 56122 Pisa, Italy (mirko.ferrati@gmail.com, carlos.rosales@for.unipi.it)

M. Gabiccini is with the DICI and Res. Center "E. Piaggio" University of Pisa, 56122 Pisa, Italy, and with the Dep. of Advanced Robotics, Istituto Italiano di Tecnologia, via Morego, 30, 16163 Genova (phone: +39-050-221.80.77, fax: +39-050-221.80.65, email: m.gabiccini@ing.unipi.it)

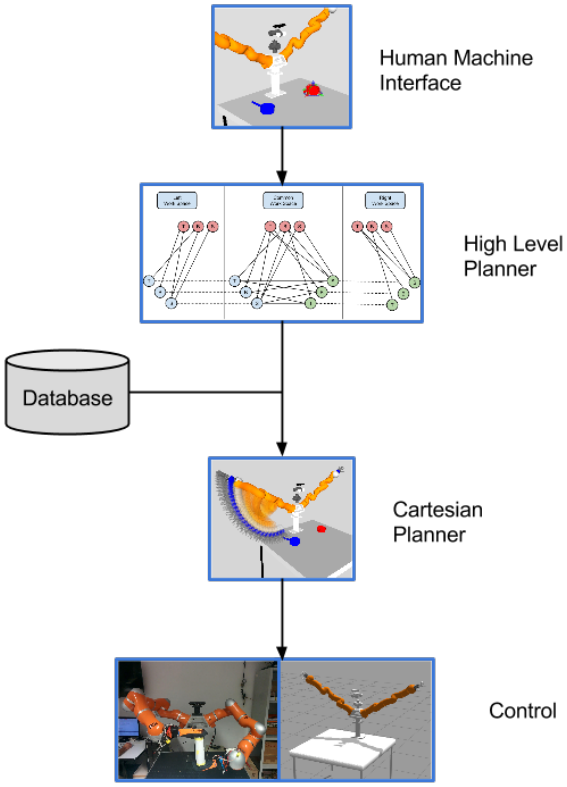


Fig. 1: Overview of the complete system.

[6] uses a hybrid task and path planner for positioning an object from a given initial to a desired final configuration, using pick and place actions that involves both arms of a bimanual robot. For all of these actions, the outcome is an abstract list of subtasks. The geometric level describing how a robot should actually perform each subtask can be carried out using probabilistic path planning methods, such as RRT [7] or PRM [8] to find a feasible, non-colliding path to each needed configuration.

[3] and [6] also use a geometric backtracking algorithm which allows for task-level replanning when a collision-free path has not been found, going back in the semantic plan, as much as needed, to find an alternative route to the goal.

Although all the previous are very promising approaches, we think they lack the possibility to include handoffs in the planning, which could be rather useful (if not necessary) in many situations.

This aspect is tackled instead in [9], where grasps are generated online once the object is perceived, as well as a way of performing the handoff. The disadvantage of this

Preprint submitted to 2015 IEEE/RSJ International Conference on Intelligent Robots and Systems. Received March 5, 2015.

approach is that it is limited to computing a single handoff transition. Therefore, the approach cannot generalize to more complex configuations when more than a single regrasping is needed.
This is not the case for [10] which, using pre-computed grasps and allowed grasp transitions (i.e., ways to perform the handoff), is capable of planning for multiple handoffs using an A* algorithm. The algorithm uses custom heuristics including distance of the object from its goal position and total number of handoffs penalties; the direct use of geometric information, although reduced to 3D, in the A* planner makes it computationally heavy. We believe that a better trade-off between speed and optimality can be obtained by implementing a high-level planner to decide which sequence of handoffs have to be performed.
Motivated by the idea of adding handoffs in a planning scheme which first considers task level planning and then geometric planning, we describe, in this paper, a methodology to solve a task given a set of *interaction primitives*.

We include, among the primitives, the generic action of passing the object from one end-effector to another, where the notion of end-effector is extended to consider also fixed, known environmental constraints when it is possible to exploit them: we will define them as *non-movable* end-effectors.

Possible grasps for each end-effector, as well as transitions between pairs of grasps, are pre-computed and stored in a database, avoiding an excessive increase in computational burden for the planning step.

Such a database can be generated via experiments, e.g. using a motion tracking system (PhaseSpace, [11]) to record hand and object pose during a grasp performed by a human "wearing" the robotic hand, simulations [12], or using diverse grasp synthesis algorithms, e.g. [13].

After highlighting the problem, we focus on the description of the proposed approach (Sec. II), and provide an in-depth view on the database structure (Sec. III), as well as the high-level planning strategy (Sec. IV). More on the low-level execution (Sec. IV-E) and other implementation details (Sec. V) follow. Experimental results (Sec. VI) and conclusions (Sec. VII) close the paper, discussing interesting extensions.

II. Problem Statement

The problem we look into to illustrate our methodology is "moving an object from a given initial pose to a desired target pose". Instead of specifying a way to achieve this result, we define a series of *interaction primitives* which are the ways in which an object can be manipulated from each end-effector. Given this simple specification, a sequence of actions may be needed in order to accomplish the task: a semantic planning step is used first, resulting in a feasible sequence of actions to be performed in order to reach the goal. A low-level planner has then to find a collision-free joint space trajectory for the robot in order to obtain the desired motion.

Specifically, we treat primitives that are: (i) handoffs, and (ii) picking/placing an object, incorporating (with the *movable* flag set to false) the environment such as a support surface in the high-level planning. It is worth remarking that a picking/placing primitive can be considered rightfully as a handoff between a movable end-effector (the hand) and a non-movable one that can apply only specific grasps to objects (see Fig. 2).

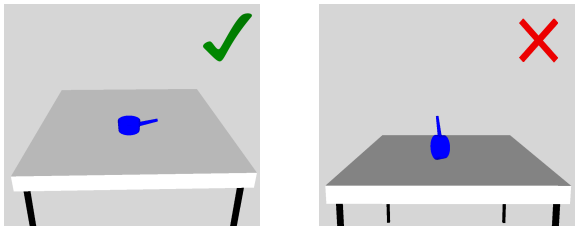


Fig. 2: A pot on a table. On the left a feasible bottom grasp, on the right an unfeasible side grasp.

Our working assumptions are that:

- vision gives to the planner all the necessary information, handling object recognition and pose estimation with sufficient accuracy;
- all objects appearing in the scene are either known or considered as fixed obstacles;
- information about the desired target configuration is given from a higher-level input, e.g. from a user (see Fig. 3).

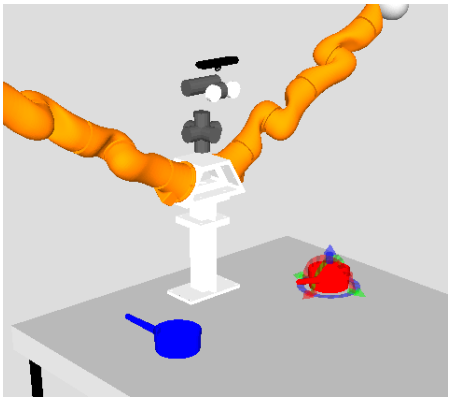


Fig. 3: The scene in Rviz. The blue object (left) represents the initial object configuration which is assumed to be given by the vision system; the red one (right) represents the goal selected by the user.

III. Database Description

The algorithm we devise relies on key information stored in a database in order to perform the planning step. A closer

look at the tables therein is taken in the following. A very simple, exemplary database is shown in Fig. 4.

A. Object Table

All known objects are included in the database; for each of them, various object geometric information such as multiple 3D views which can be used for object recognition and pose estimation using a 3D vision library are stored.

B. Grasp Table

A grasp is defined as a Cartesian trajectory of the end-effector expressed in the object frame with the corresponding joint trajectory if any. A good estimate of the final object pose within the hand is relevant to accomplish the final and intermediate goals.

Then, a grasp is associated with a single object, however, an object can have more than one grasp associated.

The generation of this table can be done in several ways. It can be produced either using simulation-based synthesis approaches [12] as well as

For each object, a series of grasps are stored for each kind of end-effector (e.g. a left hand and a right hand; different hands can be considered, too). Each grasp consists of a cartesian trajectory of the end-effector expressed in object frame, as well as final in-hand object pose.
Specifically, we use simulated synthesis of the grasps as in [12].

C. End-Effectors, Workspaces, and Reachability Tables

Each end-effector is an entity which can act on an object: there are both "movable" end-effectors (such as robot hands) and "non-movable" ones (such as a fixed surface in the environment which can be exploited). From a semantic planning perspective, there is no distinction between movable and non-movable end-effectors, both types can interact with an object in specific ways, and can exchange the object with specific grasp transitions.
In order to generate the semantic workspaces table, the environment has to be divided in smaller regions (with a grid procedure or other segmentation algorithm), each one associated to a semantic workspace. These workspaces are semantically adjacent if their corresponding regions are adjacent in the 3D space. Both geometric and adjacency information is stored.
A workspace is considered to be reachable with an end-effector only if its associated region can be reached dexterously (i.e., with good mobility); this could be checked using, e.g., Capability Maps [14].

D. Allowed Grasp Transitions

We define possible transitions among different grasps with an associated *interaction primitive*, when there is no kinematic overlap between two different end-effectors performing the grasps at the same time.

The transitions we consider in this work are:

- pick and place actions when a non-movable end-effector is involved
- a sequence of grasp-ungrasp actions when both end-effectors are movable

To check whether, given a pair of grasps, a transition between them could exist, it is possible to check for kinematic feasibility (absence of collisions between the end-effectors) when both grasps are performed at the same time; alternatively, [15] developed an automatic algorithm for directly synthesizing grasps which allow for in-air re-grasping for fully actuated hands.

Grasp_id	EE_id	Name	Allowed_transitions
g1	e1	TopTable	g5 , g6 , g8 , g9
g2	e1	SideTable	g5 , g8
g3	e1	BottomTable	g4 , g5 , g7 , g8
g4	e2	TopLeftArm	g3, g8 , g9
g5	e2	SideLeftArm	g1 , g2 , g3 , g7, g9
g6	e2	BottomLeftArm	g1 , g7 , g8
g7	e3	TopRightArm	g3 , g5 , g6
g8	e3	SideRightArm	g1 , g2 , g3 , g4 , g6
g9	e3	BottomRightArm	g1 , g4 , g5

EE_id	Reachable Workspaces	Movable	Name
e1	w1,w2,w3	FALSE	Table
e2	w1,w2	TRUE	LeftArm
e3	w2,w3	TRUE	RightArm

Workspace_id	Adjacency	Geometry	Name
w1	w2	[x1,y1...xn,yn]	Left
w2	w1,w3	[x1,y1...xn,yn]	Middle
w3	w2	[x1,y1...xn,yn]	Right

Fig. 4: Example of a DataBase overall structure.

IV. High-Level Action Planning

The high-level semantic planning involves four distinct phases:

- construction of a graph with states and transitions
- conversion of cartesian information about current and target configurations to the semantic domain
- planning on a graph of a minimum cost path from initial to target state
- conversion of the semantic plan into a cartesian plan, which can then be executed from the robot

A *state* S is a possible (end-effector, grasp, workspace) tuple.

Preprint submitted to 2015 IEEE/RSJ International Conference
on Intelligent Robots and Systems. Received March 5, 2015.

A. Graph Construction

Using the full DB structure as illustrated in Sec. III, a graph is generated. For the examplary content illustrated in Fig. 4, the generated graph can be seen in Fig. 5.
Each node of the graph represents a state $S_{e,g,w}$ of the object, where e is the end-effector which grasps the object with the grasp g in the workspace w. The arcs represent two different types of transitions:

1) from S_{e,g,w_i} to S_{e,g,w_j}: change of workspace with the same grasp, where w_i and w_j must be adjacent (in the sense of Sec. III-C) and e must be movable. This is represented by the dashed lines in Fig. 5.
2) from $S_{e_i,g_k,w}$ to $S_{e_j,g_l,w}$: change of end effector inside a certain workspace, where a transition between grasp g_k and g_l must exist (as described in Sec. III-D). This is represented by the continues lines in Fig. 5.

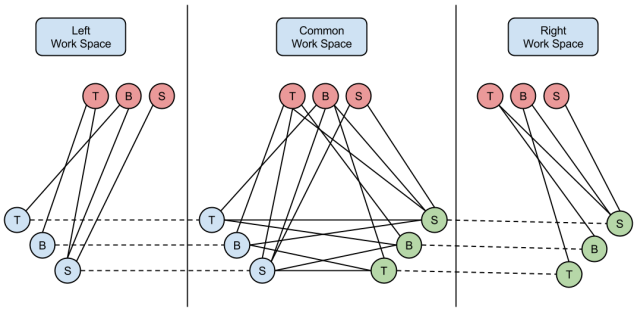


Fig. 5: Graph generated from the exemplary DB 4: 3 end-effectors, 3 workspaces, and 9 grasps. Left hand grasps, right hand grasps and table grasps are respectively represented with cyan, green and red circles while grasp types are indicated with T,B,S (top, bottom and side) inside the circles.

B. Cartesian-to-Semantic Conversion

Given the workspaces geometry associated with semantic workspace definitions, semantic states S_i and S_t representing initial and target object configurations are obtained as follows:

1) the workspaces containing the required poses are found (w_i, w_t)
2) by assumption, we consider only the non-movable end-effectors present in each of the workspaces (e_i, e_t)
3) a search for grasps (g_i, g_t) that fulfill the initial and final pose is performed.

This search involves all possible grasps associated to the end-effectors (e_i, e_t), which are checked for similarity w.r.t. the initial and target poses of the object; given the "non-movable" condition, this check is performed considering invariance to translations, and rotations around an axis normal to the end-effectors.

C. Planning on Graph

Using the graph generated in Section IV-A, Dijkstra algorithm [16] is used to find the high-level shortest path from S_i to S_t. We show in Fig. 6 a possible plan generated by hand, while in Fig. 7 the solution found by Dijkstra for the same case.
Another plan with different initial and target states that uses the table as intermediate end-effector in order to minimize path length is shown in Fig. 8. Each shortest path represents a set of workspace/grasp semantic transitions, which need to be converted in a sequence of cartesian commands for the robot.

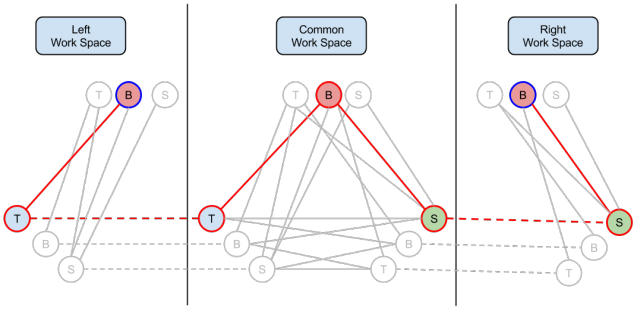


Fig. 6: Possible planning on the graph generated previously: highlighted are the initial and final states (in blue) and the path in between (red).

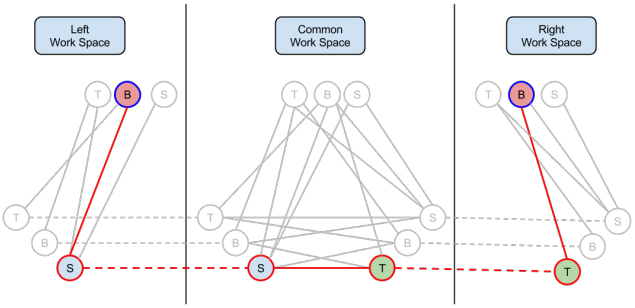


Fig. 7: Planning given by Dijkstra algorithm, the transition on the table is avoided because it would require more grasp transitions for the task.

D. Semantic-to-Cartesian Conversion

The high-level sequence of transitions obtained from the planning is simplified, by eliminating the transitions which are not useful, e.g. an end-effector moving through multiple workspaces $S_{e,g,w_1}...S_{e,g,w_n}$ becomes a single cartesian command to the final workspace w_n location. Next, each non-fixed workspace location of the object (initial and target configurations are given) are converted into a cartesian pose for the object such that all the end-effectors involved in that transition can reach it.
For this purpose, when one of the involved end-effectors

Preprint submitted to 2015 IEEE/RSJ International Conference on Intelligent Robots and Systems. Received March 5, 2015.

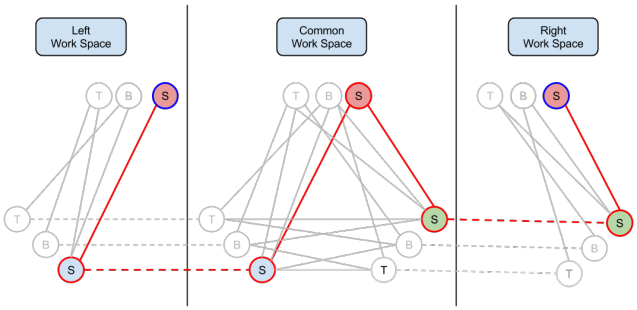


Fig. 8: Planning given by Dijkstra algorithm, the transition on the table is used because it is the shortest one in this example of initial and final states.

is not movable (e.g., the table), the position of the object is computed only considering possible rotations around the local axis aligned with the table normal (z-axis in world frame). Instead, when both end-effectors are movable, an initial guess of the object pose is considered to be where the end-effectors are mostly aligned with a "preferred" direction, and a local IK search is performed until a feasible robot configuration is found. Finally, each arc in the semantic plan is associated to low-level commands, in particular a change of workspace generates a *move* command for an end-effector, while a change of grasp generates a sequence of *grasp-ungrasp* commands for the end-effectors involved.

Summarizing, the performed steps before execution are:

- eliminating redundant semantic steps,
- translating semantic workspaces into cartesian poses,
- computing reachable object poses for each transition considering involved end-effectors,
- using the transition primitive information to plan the actions needed (i.e., *move*, *grasp-ungrasp*).

E. Execution of the Plan

The execution of the commands requires the use of a low level Inverse Kinematics planning. If more than a single end-effector *move* command has to be performed, they are performed simultaneously computing collision-free trajectories that involves the various end-effectors. This can be efficiently done e.g. using a sample-based random planner such as RRT [7]. Grasp trajectories stored in the database (Sec. III-B), which are in a frame local to the object, are translated into world coordinates and executed as well.

V. Implementation Details

Both simulated and real environment have been used to perform dual-manipulation experiments with the Vito robot[1]. The implementation of the planning algorithm, as well as the instructions to download and install the simulation package are available online as open source software[2]. A high-level scheme of the full pipeline of the experiment is depicted in Fig. 9.

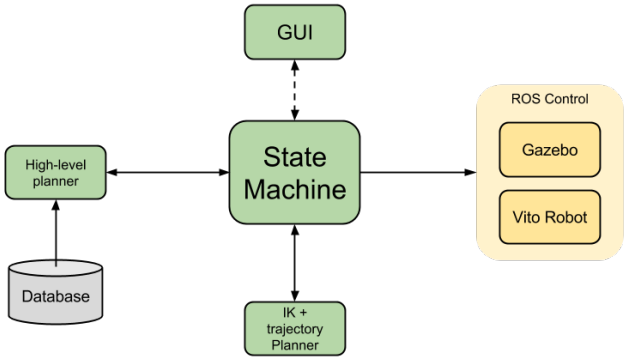


Fig. 9: High-level scheme of the full experiment pipeline with highlighted specific software blocks.

As shown in Fig. 9 the core of the framework is represented by the State-Machine, its internal structure is reported in Fig. 10. In the various states, the State-Machine communicates with different modules depending on its needs (e.g. in the *Getting Info* state it asks the vision for the starting position and the GUI for the target position).

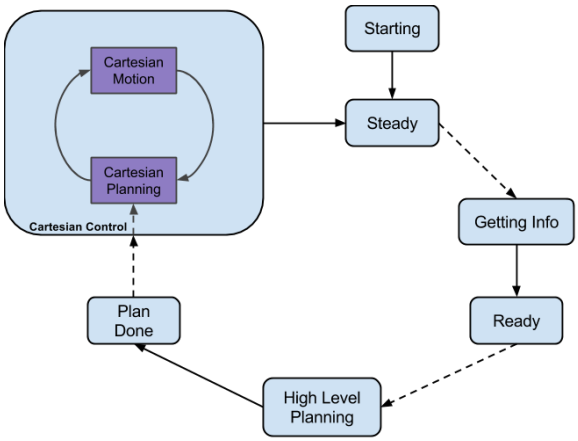


Fig. 10: The *State-Machine* structure is reported. Only the transition representing the main execution flow are in figure.

The user can, through the *GUI*, send commands to the *State-Machine* to perform a dual manipulation task. In particular, the dashed arrows in Fig. 10 represent the input from the user that can hence:

- get the object start position from the vision and set the desired final position for it,
- ask to an high level plan for the desired task,
- use the cartesian plan corresponding to the semantic generated one to perform the task with the robot (simulated or real one).

[1]http://www.pacman-project.eu/software

[2]https://bitbucket.org/dualmanipulation/dualmanipulation

The *Cartesian* control state has a sub-state-machine that iteratively plan cartesian movements and execute them up to completion of the plan generated in the *High Level Planning* state.

The experiment software, written in C++, runs on two Linux Ubuntu 14.04 PCs, using ROS [17] as a middleware for communication between vision, high- and low-level planning, and control.

VI. Experimental Results

A. Set-Up

The robot used for the validation of the approach is the Vito robot in Centro "E. Piaggio" (see Fig. 11, Fig. 12, Fig. 13).

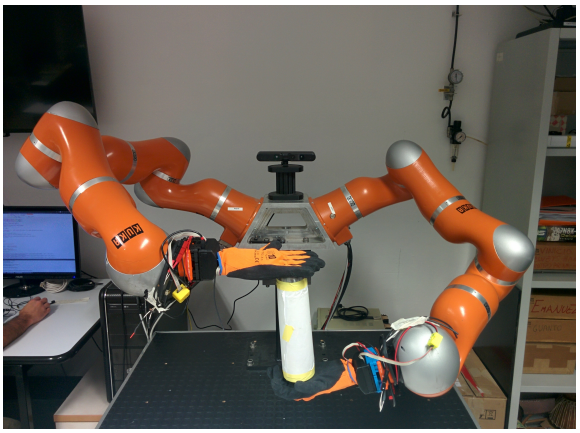


Fig. 11: Vito robot.

Two Kuka Light-Weigth Robots (LWR) mounted on a fixed torso have been equipped with a left and a right Pisa-IIT Soft-Hands.

B. Performed Experiments

In the proposed experiments a cylinder has been used. While the starting position (blue) is given by the vision system, the target position (red) is chosen by the user using the GUI[3].

In the first experiment (Fig. 12) the object starting pose is in the workspace of the right arm and the user selects the goal position in the workspace of the left arm with a similar orientation. The high level planner leads to the following behavior:

1) the robot picks up the cylinder from the top with the right hand,
2) the object is brought in the center of the common workspace,

[3]Videos regarding the robot Vito performing the described experiments will be available at *https://www.youtube.com/user/centroepiaggio*.

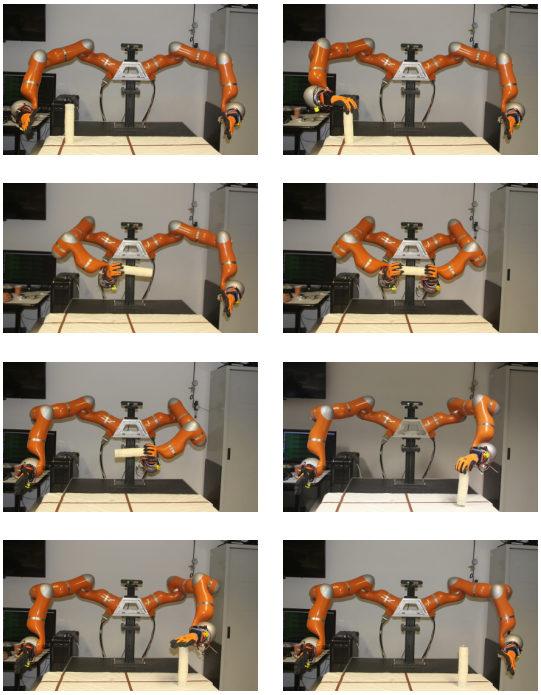


Fig. 12: Experiment 1

3) an aerial handoff is performed with the left hand grasping the object from the bottom,
4) the right hand realeases the object and the robot puts the cylinder in the target position, inside the left workspace.

Concerning the second experiment (Fig. 13), the cylinder is positioned again in the right arm workspace, this time laying on a side; the user selects the goal position in the workspace of the left arm with a similar orientation. The resulting behavior is slightly different:

1) the robot performs a side grasp on the object with the right hand,
2) then, it brings the cylinder to the common workspace, and the table is used as temporary support (i.e. the table performs a side grasp),
3) with another side grasp the robot uses the left arm to grasp the object and brings it to the target workspace.

For the sake of clarity, the reader can refer to Fig. 8 for a simplified semantic plan, which conceptually corresponds to the results depicted in Fig. 13.

VII. Conclusions

In this work we proposed a multi layer planner to solve complex object passing tasks using both handoffs and picking/placing primitives, executed by multiple end-effectors. The use of a high level semantic graph coupled with low level cartesian planner and precomputed grasp transitions allows to handle complex plans without incurring in heavy computational load. The implementation was tested on a

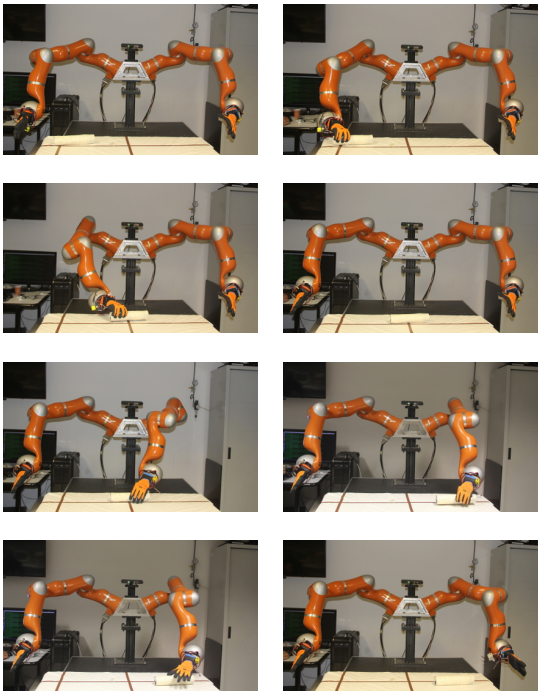


Fig. 13: Experiment 2

dual-arm robot using a cylinder with few associated grasps in order to test both the planner and the whole system in configurations that can be easily solved by a human, but that can still provide a testbed for the overall integration.

Ongoing development aims at adding arc costs which considers grasp and transition quality, in order to improve the planner decisions. Moreover, future works should extend the concept of "interaction primitives" to include dynamic primitives and extrinsic manipulation such as [18].

To allow for possible corrections during the procedure execution, we plan to include an Execution Monitoring System as suggested in [4], in order to check for correctness at every stage of the plan to be executed and considering replanning when needed.
Possible monitoring to check whether the object was successfully grasped could include visually checking the object position or reading current consumption from hand joint, thus understanding if the hand is applying force to the object or not.

ACKNOWLEDGMENTS

This work is supported by the European commission projects PaCMan EU FP7-ICT 600918.

REFERENCES

[1] M. Beetz, U. Klank, I. Kresse, A. Maldonado, L. Mosenlechner, D. Pangercic, T. Ruhr, and M. Tenorth, "Robotic roommates making pancakes," in *Humanoid Robots (Humanoids), 2011 11th IEEE-RAS International Conference on*. IEEE, 2011, pp. 529–536.

[2] L. P. Kaelbling and T. Lozano-Pérez, "Hierarchical task and motion planning in the now," in *Robotics and Automation (ICRA), 2011 IEEE International Conference on*. IEEE, 2011, pp. 1470–1477.

[3] D. Leidner, C. Borst, and G. Hirzinger, "Things are made for what they are: Solving manipulation tasks by using functional object classes," in *Humanoid Robots (Humanoids), 2012 12th IEEE-RAS International Conference on*. IEEE, 2012, pp. 429–435.

[4] L. Levison, "Connecting planning and acting via object-specific reasoning," Ph.D. dissertation, Citeseer, 1996.

[5] M. Ghallab, C. Knoblock, D. Wilkins, A. Barrett, D. Christianson, M. Friedman, C. Kwok, K. Golden, S. Penberthy, D. E. Smith *et al.*, "Pddl-the planning domain definition language," 1998.

[6] L. Karlsson, J. Bidot, F. Lagriffoul, A. Saffiotti, U. Hillenbrand, and F. Schmidt, "Combining task and path planning for a humanoid two-arm robotic system." Citeseer, 2012.

[7] S. M. LaValle, "Rapidly-exploring random trees: A new tool for path planning," 1998.

[8] 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*, vol. 12, no. 4, pp. 566–580, 1996.

[9] J.-P. Saut, M. Gharbi, J. Cortés, D. Sidobre, and T. Siméon, "Planning pick-and-place tasks with two-hand regrasping," in *Intelligent Robots and Systems (IROS), 2010 IEEE/RSJ International Conference on*. IEEE, 2010, pp. 4528–4533.

[10] B. Cohen, M. Phillips, and M. Likhachev, "Planning single-arm manipulations with n-arm robots," in *Proceedings of Robotics: Science and Systems*, 2014.

[11] Phase Space. (2015) Phase space motion capture. [Online]. Available: http://www.phasespace.com/

[12] M. Bonilla, E. Farnioli, C. Piazza, M. Catalano, G. Grioli, M. Garabini, M. Gabiccini, and A. Bicchi, "Grasping with soft hands," in *Humanoid Robots (Humanoids), 2014 14th IEEE-RAS International Conference on*, Madrid, Spain, November 2014.

[13] C. Goldfeder, P. K. Allen, C. Lackner, and R. Pelossof, "Grasp planning via decomposition trees."

[14] F. Zacharias, C. Borst, and G. Hirzinger, "Capturing robot workspace structure: representing robot capabilities," in *Intelligent Robots and Systems, 2007. IROS 2007. IEEE/RSJ International Conference on*. Ieee, 2007, pp. 3229–3236.

[15] B. Balaguer and S. Carpin, "Bimanual regrasping from unimanual machine learning," in *Robotics and Automation (ICRA), 2012 IEEE International Conference on*. IEEE, 2012, pp. 3264–3270.

[16] E. W. Dijkstra, "A note on two problems in connexion with graphs," *Numerische mathematik*, vol. 1, no. 1, pp. 269–271, 1959.

[17] "ROS: Robot operating system," www.ros.org.

[18] N. C. Dafle, A. Rodriguez, R. Paolini, B. Tang, S. S. Srinivasa, M. Erdmann, M. T. Mason, I. Lundberg, H. Staab, and T. Fuhlbrigge, "Extrinsic dexterity: In-hand manipulation with external forces," in *Robotics and Automation (ICRA), 2014 IEEE International Conference on*. IEEE, 2014, pp. 1578–1585.

A.7 Article: A computational framework for environment-aware robotic manipulation planning

Authors M. Gabiccini, A. Artoni, G. Pannocchia, J. Gillis

Info Under review

Abstract In this paper, we present a computational framework for direct trajectory optimization of general manipulation systems with unspecified contact sequences, exploiting *environmental constraints* as a key tool to accomplish a task. Two approaches are presented to describe the dynamics of systems with contacts, which are based on a penalty formulation and on a velocity-based time-stepping scheme, respectively. In both cases, object and environment contact forces are included among the free optimization variables, and they are rendered consistent via suitably devised sets of complementarity conditions. To maximize computational efficiency, we exploit sparsity patterns in the linear algebra expressions generated during the solution of the optimization problem and leverage Algorithmic Differentiation to calculate derivatives. The benefits of the proposed methods are evaluated in three simulated planar manipulation tasks, where essential interactions with environmental constraints are automatically synthesized and opportunistically exploited.

Relation with the deliverable grasp and manipulation planning for systems with a-priori unspecified contact sequences.

Attachment (following pages until next annex)

A Computational Framework for Environment-Aware Robotic Manipulation Planning

Marco Gabiccini Alessio Artoni Gabriele Pannocchia Joris Gillis

***Abstract*— In this paper, we present a computational framework for direct trajectory optimization of general manipulation systems with unspecified contact sequences, exploiting *environmental constraints* as a key tool to accomplish a task. Two approaches are presented to describe the dynamics of systems with contacts, which are based on a penalty formulation and on a velocity-based time-stepping scheme, respectively. In both cases, object and environment contact forces are included among the free optimization variables, and they are rendered consistent via suitably devised sets of complementarity conditions. To maximize computational efficiency, we exploit sparsity patterns in the linear algebra expressions generated during the solution of the optimization problem and leverage Algorithmic Differentiation to calculate derivatives. The benefits of the proposed methods are evaluated in three simulated planar manipulation tasks, where essential interactions with environmental constraints are automatically synthesized and opportunistically exploited.**

I. INTRODUCTION

Careful observation of how humans use their hands in grasping and manipulation tasks clearly suggests that their limbs extensively engage functional interactions with parts of the environment. The physical constraints imposed by the manipulandum and the environment are not regarded as obstacles, but rather as opportunities to guide functional hand pre-shaping, adaptive grasping, and affordance-guided manipulation of objects. The exploitation of these opportunities, which can be referred to as *environmental constraints* (EC), enables robust grasping and manipulation in dynamic and highly variable environments. When one considers the exploitation of EC, i.e. when manipulation actions are performed *with the help* of the environment, the boundary between grasping and manipulation is blurred, and traditional categories such as grasp and manipulation analysis, trajectory planning and interaction control appear somewhat artificial, as the problem we aim to solve seems to inextricably entangle all of them.

In this paper, we set out to formulate environment-aware manipulation planning as a nonlinear optimal control problem and discretize it according to a *direct transcription* scheme [1]. In Sec. III, two approaches to describe the dynamics of systems with contacts are proposed and evaluated: in the first one, continuous contact reaction forces are generated by nonlinear virtual springs, and the requirement to avoid sliding contacts is handled in an apparently original way; in the second one, contact collisions are approximated as impulsive events causing discontinuous jumps in the velocities according to a modified version of the Stewart-Trinkle time-stepping scheme [2]. The introduction of two different models is motivated by the relative ease for the first (second) one to enforce EC exploitation primitives that avoid (profitably exploit) sliding motions during interaction.

Both formulations lead to a nonlinear programming (NLP) problem (Sec. IV) that we solve by using the Interior-Point (IP) method implemented in IPOPT [3] and discussed in Sec. V. To improve computational efficiency, we also annotate sparsity in the linear algebra expressions and leverage algorithmic differentiation (AD) [4] to calculate derivatives both quickly and accurately: the adoption of the CasADi framework [5], described in Sec. V, provides a profitable interface to all the above tools with a consistent API.

In Sec. VI, we evaluate our approaches in three simulated planar manipulation tasks: (i) moving a circular object in the environment with two independent fingers, (ii) rotating a capsule with an underactuated two-fingered gripper, and (iii) rotating a circular object in hand with three independent fingers. Tasks (i) and (ii) show that our algorithm quickly converges to locally optimal solutions that opportunistically exploit EC. Task (iii) demonstrates that even dexterous fingertip gaits can be obtained as a special solution in the very same framework. Conclusions are drawn in Sec. VII.

It is worth noting that with our method, approach planning, grasping, manipulation, and environment constraint exploitation phases occur automatically and opportunistically for a wide range of tasks and object geometries, with no *a-priori* specification of the ordering of the different stages required.

II. RELATED WORK

A. Exploitation of Environmental Constraints

The concept of exploiting environmental constraints is well-rooted in robotics. Pioneering work was performed already in the eighties in the context of motion planning [6] and manipulation [7]. However, these concepts did not have the proper influence on many of the recent developments on either area, perhaps due to the inadequacy of the mechanical impedance properties of contemporary industrial manipulators to achieve sliding motion primitives stably, thus precluding the adoption of strategies that exploit environmental constraints, e.g. by sliding one object over another [8]. Also the idea of programming using environmental constraints is well entrenched in robotics literature, starting with the seminal work [9], proceeding with [10], and culminating in the *iTaSC* framework [11].

The exploitation of complex interactions with the environment in a manipulation task also plays a central role

M. Gabiccini, A. Artoni and G. Pannocchia are with Department of Civil and Industrial Engineering, Univ. of Pisa, Italy ({firstname.lastname}@unipi.it). J. Gillis is with the Department of Electrical Engineering, KU Leuven, Belgium (joris.gillis@kuleuven.be). Corresponding author is M. Gabiccini.

in automation and manufacturing to design fixtures [12] and part feeders [13]. However, these works are highly specialized and they are limited to the case of handling a single object geometry.

B. Traditional Grasp Planners

State-of-the-art general grasping algorithms and dexterous mechanical hands lie at the opposite end of the spectrum: they are designed to perform grasping and manipulation of a wide range of object geometries and for many different tasks. Traditional grasp planners (such as OpenRAVE [14] and GraspIt! [15]) rely on precise finger-to-object contact points while avoiding the surrounding environment. In real-world scenarios these models, as well as the motion of the hand, will be highly uncertain leading to poor grasping performance for grasps that were deemed highly robust based on theoretical considerations.

The recent paper [16] has proposed a pipeline for automated grasp synthesis of common objects with a compliant hand by developing a full-fledged multi-body simulation of the whole grasping process in the presence of EC: however, to date, the approach seems time consuming and the sequence of primitive actions needed to perform complex tasks must be scripted in advance. Also recently, both grasp planning algorithms and grasping mechanisms have begun to take advantage of EC [17], albeit not systematically. While sequences of EC exploitation primitives have been shown to be robust and capable [18], [19], there has been no comprehensive research on how to enumerate and describe these primitives or how to sequence them into task-directed manipulation plans.

C. General Purpose Planning Algorithms

State-of-the-art sampling-based planning algorithms like RRT*[20] seem not tailored for situations where *a-priori* unknown contact interactions may cause a combinatorial explosion of system configurations. Recent extensions, initially presented in [21] for RRT, and successively devised for RRT* in [22], were able to cope with systems described by complex and underactuated dynamics. In [23], the authors presented an approach to moving an object with several manipulators. This problem presents some similarities to ours, since a sequence of phases has to be both planned and solved. The strong assumption that the plan called by the high-level scheduler will succeed — which is not always possible — has been removed in the recent contribution [24].

However, situations where intermitted contact sequences are not easily enumerated from the outset, but are key for the success of environment-aware manipulation plans, still appear to be out of their reach.

D. Machine Learning Approaches

Although significant progresses have been made in this area in recent years [25], learning robot motion skills still remains a major challenge, especially for systems with multiple intermittent contacts. Policy search is often the preferred method as it scales gracefully with system dimensionality, even if its successful applications typically rely on a compact representation that reduces the numbers of parameters to learn [26], [27], [28]. Innovative policy classes [29] have led to substantial improvements on real-world systems. However, designing the right low-dimensional representation often poses significant challenges. Learning a state representation that is consistent with physics and embeds prior knowledge about interactions with the physical world has recently been proposed in [30] and seems a promising venue to find effective methods to help improve generalization in reinforcement learning: however, the simulated robotic tasks solved by this methods are still far in complexity from environment-aware manipulation scenarios.

The recent contribution [31] developed a policy search algorithm which combines a sample-efficient method for learning linear-Gaussian controllers with the framework of guided policy search, which allows the use of multiple linear-Gaussian controllers to train a single nonlinear policy with any parametrization, including complex and high-dimensional policies represented by large neural networks. In [32], this method has been recently applied, with some modifications that make it practical for deployment on a robotic platform, to solve contact-rich manipulation tasks with promising results.

E. Optimization-based Trajectory Planning

Various research groups are currently pursuing direct trajectory optimization to synthesize complex dynamic behaviors for systems with intermittent contacts. Those working in locomotion [33] mainly adopt a multi-stage hybrid-mode approach (usually employing multiple-shooting), where the optimization is constrained to operate within an *a priori* specification of the mode ordering. Interestingly, a recent contribution [34] explored the synthesis of optimal gaits for legged robots without the need to specify contact sequences. Certainly, the adoption of such an approach seems the only viable solution for a multi-fingered hand manipulating an object also by exploiting EC. Along this line, it is worth mentioning the contact-invariant approach originally proposed in [35] to discover complex behaviors for humanoid figures and extended to the context of manipulation in [36]. The previously described trajectory optimization method has been recently employed to gradually train a neural network by following an Alternating Direction Method of Multipliers (ADMM) strategy with interesting results [37].

The approach presented in [38] inspired our work and is the one which is definitely closest. However, remarkable differences in the formulation of the dynamics for systems with contacts, in the choice of the solution algorithm, solver and framework, and in the focus of the paper — here, EC exploitation is sought as a key factor — render our work significantly different.

III. DYNAMICS OF SYSTEMS WITH CONTACTS

A. Penalty-based contact model

To our eyes, manipulation planning has to rely on a dynamic model of the system, namely of manipulandum, manipulator, and the environment, and of their mutual interactions through contact. We consider here deterministic systems with continuous state and control spaces, denoted by x and u respectively. The dynamic evolution of our controlled

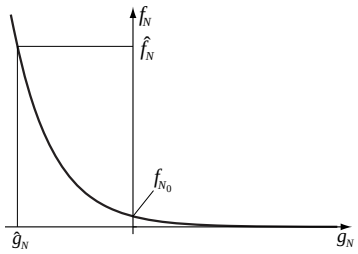


Fig. 1. Normal contact force as a function of the normal gap.

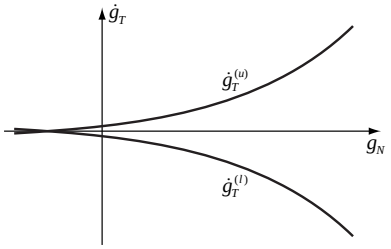


Fig. 2. Example of sliding velocity funnel.

(non-autonomous) system can be described in continuous time t by a set of Ordinary Differential Equations (ODEs):

$$\dot{x}(t) = F(x(t), u(t)) \quad \text{or} \quad \tilde{F}(\dot{x}(t), x(t), u(t)) = 0 \tag{1}$$

(explicit dependence on t is omitted hereafter). Together with an initial value $x(0) = x_0$, equation (1) defines an initial value problem. In general, additional algebraic dependencies may exist among $\dot{x}$, x and u leading to a dynamic system governed by differential-algebraic equations (DAEs). In the presence of contact, for instance, and if contact forces are included among the controls u, contact interactions establish functional dependencies between u and x through a set of algebraic equations/inequalities. As an example, if f_N is the normal contact force and $g_N = g_N(x)$ the normal gap (shortest distance) between a finger and the object being manipulated, the complementarity and non-negativity conditions $0 \leq f_N \perp g_N(x) \geq 0$ must hold. The contact model described in this section is based on a special treatment of the contact forces and the relative velocities that arise during interaction between manipulandum, manipulator, and environment. Such a model has proved to be successful in solving trajectory planning problems for manipulation tasks with *no sliding*, in the presence of EC.

For normal contact forces, the underlying idea is borrowed from classical penalty-based approaches, where contact interactions are modeled by spring-dampers. In our model, no damping is introduced, while a nonlinear exponential spring relates the normal contact force and the normal gap through the constitutive equation (Fig. 1)

$$f_N(g_N(x)) = f_{N_0} \left(\frac{\hat{f}_N}{f_{N_0}} \right)^{g_N(x)/\hat{g}_N} \tag{2}$$

where $g_N(x)$ is the normal gap function and $\hat{g}_N$ the negative normal gap value (penetration) corresponding to the normal force value $\hat{f}_N$. The (fixed) parameters $(\hat{f}_N, \hat{g}_N)$ provide a straightforward way to properly "calibrate" the model and adapt it to the problem at hand. Relation (2) is a relaxation of the above-stated complementarity condition: it avoids discontinuities while being sufficiently representative of physical reality. In order to describe (unilateral) contacts with friction, the classical Coulomb model is adopted. The focus is placed here on point-contact with *static* friction, whereby normal force f_N, tangential force f_T and the coefficient of static friction μ_s are related by the well known relation

$$f_T \leq \mu_s f_N \tag{3}$$

No-sliding conditions must be enforced by requiring that sliding velocities at contact points be zero. The normal gap $g_N(x)$ and the sliding velocity $\dot{g}_T(\dot{x}, x)$ (i.e., the time derivative of the tangential gap [39]) would call for an additional complementarity condition. We devised a smooth relaxation of this discontinuous condition through the *sliding velocity funnel* shown in Fig. 2 and described by:

$$\dot{g}_T^{(l)}(g_N) \leq \dot{g}_T \leq \dot{g}_T^{(u)}(g_N) \tag{4}$$

where the functions $\dot{g}_T^{(l)}(g_N)$ and $\dot{g}_T^{(u)}(g_N)$ are lower and upper bounds, respectively, for the sliding velocity. It is reasonable to have a symmetric funnel, hence $\dot{g}_T^{(l)}(g_N) = -\dot{g}_T^{(u)}(g_N)$. The bounds are modeled here as:

$$\dot{g}_T^{(u)}(g_N) = \exp(c_1(g_N + c_2)) + c_3 \tag{5}$$

where the three parameters (c_1, c_2, c_3) are used for proper, problem-dependent calibration. No-sliding manipulation planning benefits from this approach as the sliding velocity funnel smoothly and gradually guides the fingers towards the object, eventually driving relative velocities to (nearly) zero at their contact points. The trajectory planning methods employed in this work are based on numerical optimization techniques that require a discrete-time model of the dynamic system, therefore discrete-time versions of eqs. (1)–(4) are adopted in the following. In our implementation, the state vector $x_k = x(t_k)$ collects configuration and velocity of each body, while control vector $u_k = u(t_k)$ includes acceleration of each finger (or actuator) and contact forces at each candidate contact point. The dynamic equation (1) is used in a direct transcription scheme based on *collocation points*: using a single collocation point as midpoint in the interval $[t_k, t_{k+1}]$, and denoting $\bar{t}_k = (t_{k+1} + t_k)/2$, $\bar{x}_k = x(\bar{t}_k)$ and $h = t_{k+1} - t_k$ (fixed), the discretized dynamic equation becomes

$$x_{k+1} = x_k + hF(\bar{x}_k, u_k) \tag{6}$$

In the above implementation, states are linear and controls are constant over each discretization interval. The integration scheme (6) is known as implicit midpoint rule ($O(h^2)$), and it is the special one-point case of the Gauss-Legendre collocation method. As it is a symplectic integrator, it is suitable to cope with stiff, conservative mechanical systems. While eqs. (3)–(4) are straightforward to discretize, eq. (2) needs some attention: as it involves both states and controls, its discretization must adhere to the scheme dictated by eq. (6), to wit

$$f_{N_k} = f_N(g_N(\bar{x}_k)) = f_{N_0} \left(\frac{\hat{f}_N}{f_{N_0}} \right)^{g_N(\bar{x}_k)/\hat{g}_N} \tag{7}$$

Failing to do so (e.g., evaluating g_N at x_k) would result in a "causality violation", with contact forces being inconsistent with the interpenetrations between bodies governed by (6).

B. Velocity-based time stepping scheme

In this section, we present the complementarity formulation of the time-stepping scheme employed for the dynamic modeling of manipulation systems exploiting *sliding over* EC. To keep the treatment general enough, we refer to a 3D system of m bodies with c contacts. We assume a polyhedral approximation of the friction cone, with d friction directions uniformly distributed to positively span the contact tangent plane[1]. For ease of notation, we write $M_k = M(q_k)$ and likewise for other vector/matrix functions. Let h be again the time step, and let $\Delta v := v_{k+1} - v_k$. For $k \in \{0,\dots,N-1\}$, we adopt a Backward-Euler transcription scheme by writing the *kinematic reconstruction* and the *dynamic* equations as

$$q_{k+1} - q_k - h\, v_{k+1} = 0 \tag{8a}$$

$$M_{k+1}\Delta v - h\big[\kappa(q_{k+1}, v_k) + B_{k+1}u_{k+1}\big] - G_{k+1}\lambda_{k+1} = 0, \tag{8b}$$

where: $q \in \mathbb{R}^{6m}$ and $v \in \mathbb{R}^{6m}$ represent system configuration and velocity, respectively, $M \in \mathbb{R}^{6m\times 6m}$ is the generalized mass matrix, $\kappa \in \mathbb{R}^{6m}$ collects centrifugal, Coriolis and gravitational forces, $u \in \mathbb{R}^t$ is the control torque vector, $B \in \mathbb{R}^{6m\times t}$ is the actuation matrix, $G = [N\ T]$ is a *generalized grasp* matrix, where $N \in \mathbb{R}^{6m\times c}$ and $T \in \mathbb{R}^{6m\times nd}$ are normal and tangential wrench bases, and $\lambda = [\lambda_N^\top\ \ \lambda_T^\top]^\top$ is the generalized wrench impulse vector, wherein λ_N and λ_T are the normal and tangential contact wrench impulses. Matrix N appears as $N = [N^{(1)}\cdots N^{(c)}]$, and each column $N^{(i)} \in \mathbb{R}^{6m}$ corresponds to contact i and contains, for each body ℓ connected to contact i, a block of rows of the form $\pm[n_i^\top\ \ (p_{\ell,i}\times n_i)^\top]^\top$[2]. Since there are at most two bodies connected to a contact, each $N^{(i)}$ has at most 12 non-zero elements. Similarly, $T = [T^{(1)}\cdots T^{(c)}]$, and in the generic block $T^{(i)} \in \mathbb{R}^{6m\times d}$, each column $T^{(i,j)} \in \mathbb{R}^{6m}$ contains, for each body ℓ connected to contact i, a block of rows of the form $\pm[t_{i,j}^\top\ \ (p_{\ell,i}\times t_{i,j})^\top]^\top$, where $t_{i,j}$ denotes friction direction j at contact i. Opposite signs must be selected for each of the two bodies connected to contact i, and each column of T will contain, at most, 12 non-zero elements.

In partial accordance to [2], *unilateral* contacts with friction can be described by the following set of *inequality* and *complementarity* conditions

$$0 \le \lambda_{N_{k+1}} \perp g_N(q_{k+1}) \ge 0 \tag{9a}$$

$$0 \le \lambda_{T_{k+1}} \perp \big(\dot g_T(q_{k+1}, v_{k+1}) + E\gamma_{k+1}\big) \ge 0 \tag{9b}$$

$$0 \le \gamma_{k+1} \perp \big[\mu\lambda_{N_{k+1}} - (\lambda_{T_{k+1}}^\top H\, \lambda_{T_{k+1}})^{\frac{1}{2}}\big] \ge 0, \tag{9c}$$

where $g_N(\cdot)$ is the normal gap function and $\dot g_T(\cdot)$ is the time derivative of the tangential gap function [39], γ represents, in most cases[3], an approximation to the magnitude of the relative contact velocity, matrix $E := \mathrm{BlockDiag}(\mathbf{1},\dots,\mathbf{1}) \in \mathbb{R}^{(dc)\times c}$, with $\mathbf{1} \in \mathbb{R}^d$, $\mu \ge 0$ is the coefficient of friction, and $H := \mathrm{BlockDiag}(H^{(1)},\dots,H^{(c)})$, where the $H_{lm}^{(i)} = t_{i,l}^\top t_{i,m}$ $(l,m \in \{1,\dots,d\})$ is the metric form [40, Sec. 2-5] of the basis $t_{i,l}$ that positively spans the tangent plane at contact i. Eqs. (9a) state that bodies cannot interpenetrate ($g_N(q_{k+1}) \ge 0$), normal impulses can only push bjects away ($\lambda_{N_{k+1}} \ge 0$), and that, in order for the impulse to be non-zero in the interval $[t_k, t_{k+1}]$, the normal gap must be closed at t_{k+1}. This condition also implies that collisions are approximated here as inelastic ones, and interacting bodies may end up sticking together. Eqs. (9b) require tangential impulses to be directed along the positive tangential directions ($\lambda_{T_{k+1}} \ge 0$). The complementarity condition in (9b) selects, for sliding contacts, the tangential impulse that opposes the sliding velocity. This constraint is tightly coupled with the complementarity condition in eqs. (9c), and it ensures that, if a contact is sliding, the tangential force will lie on the boundary of the friction cone. It is worth noting that the bracketed term in eq. (9c) allows one to correctly define the Coulomb friction constraints even in sticking conditions, as it is robust to the physiological failure of eq. (9b) in selecting only one non-zero component in each $\gamma_{k+1}^{(i)}$ for adhesive contacts[4].

The choice of fully implicit integration schemes and nonlinear complementarity formulations, as described in Eqs. (8) and (9), can be justified in view of the increased numerical stability and modelling accuracy they bring about, while not hindering the general structure of the problem[5].

IV. TRAJECTORY PLANNING AS AN OPTIMIZATION PROBLEM

A. Penalty-based contact model

Within our direct transcription framework, for $k \in \{0,\dots,N-1\}$, eqs. (6), (7), and the discretized versions of eqs. (3) and (4) constitute a set of (equality and inequality) *nonlinear constraints* for the optimal control problem (OCP) we set out to formulate. Additional constraints include the (fixed and known) initial state values $x_0 = x(0)$ as well as a *terminal equality constraint* on (some) components of x_N: the latter provides a direct way to specify the required final state of the manipulated object at the final time $T = t_N$. Generally, no terminal constraints on the manipulation system configuration are imposed. Lower and upper bounds for $(x_k, \bar x_k, u_k)$ are also included: they act as operational constraints for actuators and the system's workspace, and they are useful to restrain contact forces within safety limits. Other constraints can be introduced to shape emergent behaviors and to render them intrinsically more robust or desirable for several reasons. As an illustrative example, in order to guarantee that any two fingers make always contact with an object in a three-fingered manipulation task, we add the following set of inequalities to the problem: $f_{N_k}^{(1)} + f_{N_k}^{(2)} \ge \varepsilon$, $f_{N_k}^{(2)} + f_{N_k}^{(3)} \ge \varepsilon$, and $f_{N_k}^{(3)} + f_{N_k}^{(1)} \ge \varepsilon$, with $\varepsilon > 0$.

[1] For simplicity of description, we assume that the number of friction directions d is the same at each contact, although this is not necessary.

[2] The positive/negative sign must be chosen if, considering equilibrium of body ℓ, the unit normal vector n_i is facing into/away from body ℓ.

[3] In situations where the relative contact velocity and the friction vector are both zero, $\gamma \ge 0$ can be arbitrary and has no physical meaning.

[4] Replacing the bracketed term in (9c) with $[\mu\lambda_N - E^\top\lambda_T]$, as commonly performed in literature [41], would call for unrealistically strict and physically unmotivated conditions to ensure adhesive friction.

[5] Embedding contact dynamics into the numerical optimization problem as nonlinear constraints, where many other implicit constraints are already present, does not justify explicit or semi-implicit discretization schemes, which are, instead, legitimate when building fast simulators [42, Sec. 5].

We introduce the vector of decision variables $\boldsymbol{v} \in \mathbb{R}^n$, which collects the sequence of unknown $(x_k, \bar{x}_k, u_k)$ (i.e., configurations and velocities in $(x_k, \bar{x}_k)$, contact forces and actuator accelerations in u_k). All equality and inequality constraints can be written compactly as

$$g_{\min} \leq g(\boldsymbol{v}) \leq g_{\max}, \qquad \boldsymbol{v}_{\min} \leq \boldsymbol{v} \leq \boldsymbol{v}_{\max} \tag{10}$$

The general structure of the cost function takes the form

$$f(\boldsymbol{v}) = \sum_{k=0}^{N-1} \sum_{i \in \mathscr{I}} w_i \phi_i(x_k, u_k), \tag{11}$$

where each $\phi_i(\cdot)$ represents a peculiar type of cost. These have to be carefully selected according to the character of the manipulation action we desire to perform, along with the corresponding *weights* w_i (also acting as important scaling factors). Multiple cost terms $\phi_i(\cdot)$ can be used to shape different manipulation behaviors. The following terms (specifically, their squared 2-norm) have proved to be decisive in directing the optimization process: contact forces, and their variations from one interval to the next (to minimize jerk); accelerations of actuators; deviations of actual object trajectories from ideal, smooth trajectories (task-specific).

B. Velocity-based time stepping scheme

Similarly to the penalty-based contact model scheme, the discrete formulation of the dynamics of systems with contacts expressed by eqs. (8) and (9) can be used as a set of nonlinear constraints in an optimal control problem (OCP), involving the sequence of unknown $(q_k, v_{k+1}, \lambda_{k+1}, \gamma_{k+1})$. Additional equality constraints are introduced: for the manipulated object, both initial and final configurations and velocities are imposed, whereas only the initial conditions are specified for the manipulation system.

Inequality constraints are also introduced with similar intentions as in the penalty-based scheme. It is worth noting that, since in our applications the hand is velocity controlled, the hand dynamics is not included in the optimization constraints, and the hand velocities will play the role of control actions. Therefore, limited control authority is imposed as bounds on hand velocities and accelerations in the form $v^{(l)}_{\min} \leq v^{(l)} \leq v^{(l)}_{\max}$ and $a^{l}_{\min} h \leq \Delta v^{(l)} \leq a^{(l)}_{\max} h$, respectively, with l belonging to the index set corresponding to the hand.

Defining $\boldsymbol{v} \in \mathbb{R}^n$ as the multi-stage sequence of configurations, velocities, contact impulses and inputs, $(q_k, v_{k+1}, \lambda_{k+1}, \gamma_{k+1})$, all constraints are still expressed in the form (10), whereas the cost function takes the form

$$f(\boldsymbol{v}) = \sum_{k=0}^{N-1} \sum_{i \in \mathscr{I}} w_i \phi_i(q_k, v_{k+1}, \lambda_{k+1}, \gamma_{k+1}) \tag{12}$$

C. Final optimization problem

From the previous discussion, we now present the nonlinear program (NLP) that has to be solved to generate optimal trajectories. With $\boldsymbol{v} \in \mathbb{R}^n$ previously defined, we consider the following optimization problem

$$\min_{\boldsymbol{v}} f(\boldsymbol{v}), \qquad \text{subject to} \tag{13a}$$

$$g_{\min} \leq g(\boldsymbol{v}) \leq g_{\max} \tag{13b}$$

$$\boldsymbol{v}_{\min} \leq \boldsymbol{v} \leq \boldsymbol{v}_{\max} \tag{13c}$$

in which: $f : \mathbb{R}^n \to \mathbb{R}$ is the objective function, $g : \mathbb{R}^n \to \mathbb{R}^m$ is the nonlinear constraint function, $g_{\min} \in [-\infty, \infty)^m$ and $g_{\max} \in (-\infty, \infty]^m$ (with $g_{\min} \leq g_{\max}$) are, respectively, lower and upper bound vectors of the nonlinear constraints, $\boldsymbol{v}_{\min} \in [-\infty, \infty)^n$ and $\boldsymbol{v}_{\max} \in (-\infty, \infty]^n$ (with $\boldsymbol{v}_{\min} \leq \boldsymbol{v}_{\max}$) are, respectively, lower and upper bound vectors of the decision variables. Problem (13) is a *large-scale*, but *sparse* NLP, that should be solved by structure-exploiting solvers. To this end, as detailed in the next section, we resorted to the IPOPT [43] implementation of the interior-point method within the CasADi framework. As initial guess required by the algorithm, we somewhat crudely mapped the initial state x_0 and a rough estimate of the controls to all the $(N-1)$ variable instances. Obviously, better initial guesses should be provided whenever possible. With the penalty-based approach, (partial) solutions of the NLP (13) have also been used as initial guesses for a subsequent optimization according to a *homotopy* strategy [44, Sec. 11.3], thereby maximizing physical realism while facilitating convergence.

V. NONLINEAR PROGRAMMING VIA AN INTERIOR-POINT ALGORITHM

A. The barrier problem formulation

Problem (13) is equivalently rewritten as

$$\min_{x} f(x), \qquad \text{subject to} \tag{14a}$$

$$c(x) = 0 \tag{14b}$$

$$x_{\min} \leq x \leq x_{\max} \tag{14c}$$

in which x is formed by augmenting the decision variable vector $\boldsymbol{v}$ with suitable slack variables that transform inequality constraints (13b) into equality constraints.[6]

Let $I_{\min} = \{i : x^{(i)}_{\min} \neq -\infty\}$, and $I_{\max} = \{i : x^{(i)}_{\max} \neq \infty\}$. We consider the barrier function

$$\varphi_\mu(x) = f(x) - \mu \left(\sum_{i \in I_{\min}} \ln(x^{(i)} - x^{(i)}_{\min}) + \sum_{i \in I_{\max}} \ln(x^{(i)}_{\max} - x^{(i)}) \right)$$

in which $\mu > 0$ is a (small) barrier parameter. Instead of solving (14), IPOPT performs iterations to achieve an approximate solution of the equality constrained NLP

$$\min_{x} \varphi_\mu(x), \qquad \text{subject to: } c(x) = 0 \tag{15}$$

Note that $\varphi_\mu(x)$ is well defined if and only if $x_{\min} < x < x_{\max}$, i.e. if x is in the *interior* of its admissible region. The value of μ is progressively reduced so that $\varphi_\mu(x) \to f(x)$, and in this way solving (15), in the limit, becomes equivalent to solving (14). Clearly, as $\mu \to 0$, any component of x can approach its bound if this is required by optimality.

B. Interior-point approach to NLP

Any local minimizer to (15) must satisfy the following Karush-Kuhn-Tucker (KKT) conditions [44, Sec. 12.2]

$$\nabla f(x) + \nabla c(x) \lambda - \underline{z} + \bar{z} = 0 \tag{16a}$$

$$c(x) = 0 \tag{16b}$$

$$\underline{z}^{(i)} (x^{(i)} - x^{(i)}_{\min}) - \mu = 0 \quad \forall i \in I_{\min} \tag{16c}$$

$$\bar{z}^{(i)} (x^{(i)}_{\max} - x^{(i)}) - \mu = 0 \quad \forall i \in I_{\max} \tag{16d}$$

[6] With a slight abuse of notation, we still use n and m to denote the dimension of x and $c(x)$, respectively, and $f(x)$ to denote $f(\boldsymbol{v})$.

TABLE I

BASIC ALGORITHM IMPLEMENTED IN IPOPT.

1) Define $\mu_0 > 0$, x_0 ($x_{\min} \leq x_0 \leq x_{\max}$), λ_0, $\underline{z}_0 \geq 0$, $\overline{z}_0 \geq 0$, and form ξ_0 accordingly. Set: $j = 0$, $k = 0$.
2) Given the current iterate ξ_k, compute a Newton step p_k for $F(\xi) = 0$. Compute the new iterate performing a line search: $\xi_{k+1} = \xi_k + \alpha_k p_k$ (for some $\alpha_k > 0$).
3) If $E_0(\xi_{k+1}) \leq \varepsilon$, exit: ξ_{k+1} is a local solution to NLP (14). Otherwise, proceed to Step 4.
4) If $E_{\mu_j}(\xi_{k+1}) \leq \kappa\mu_j$ (for some $\kappa > 0$) proceed to Step 5. Otherwise, update $k \leftarrow k+1$ and go to Step 2.
5) Set $\mu_{j+1} = \mu_j/\rho$ (for some $\rho > 1$), update $j \leftarrow j+1$, $k \leftarrow k+1$ and go to Step 2.

for some vectors $\lambda \in \mathbb{R}^m$, $\underline{z} \in \mathbb{R}^n$, and $\overline{z} \in \mathbb{R}^n$ (for completeness: $\underline{z}^{(i)} = 0 \quad \forall i \notin I_{\min}$, $\overline{z}^{(i)} = 0 \quad \forall i \notin I_{\max}$). Notice that, if $\mu = 0$, then (16) together with $\underline{z} \geq 0$ and $\overline{z} \geq 0$ represent the KKT conditions for NLP (14). The KKT conditions (16) form a nonlinear algebraic system $F(\xi) = 0$ in the unknown $\xi = (x, \lambda, \underline{z}, \overline{z})$, which is solved in interior-point algorithms via Newton-like methods. If we denote by $E_\mu(\xi)$ the maximum absolute error of the KKT equations (16) (appropriately scaled), the basic algorithm implemented in IPOPT is summarized in Table I (in which j is the index of the outer loop, k is the index of the inner loop, and $\varepsilon > 0$ is a user-defined convergence tolerance).

C. Main computational aspects: calculating derivatives and solving (sparse) linear systems

The most expensive computation step in the basic interior-point algorithm is the computation of the Newton step p_k for the KKT system $F(\xi) = 0$, i.e. Step 2. We first note that evaluation of $F(\xi)$, at each iteration, involves the computation of the cost function gradient $\nabla f(x) \in \mathbb{R}^n$ and of the constraint Jacobian $\nabla c(x) \in \mathbb{R}^{n \times m}$. Then, the Newton step is found from the solution of the following linear system:

$$\begin{bmatrix} W_k & A_k & -I & I \\ A_k^T & 0 & 0 & 0 \\ \underline{Z}_k & 0 & \underline{X}_k & 0 \\ -\overline{Z}_k & 0 & 0 & \overline{X}_k \end{bmatrix} \begin{bmatrix} p_k^x \\ p_k^\lambda \\ p_k^{\underline{z}} \\ p_k^{\overline{z}} \end{bmatrix} = - \begin{bmatrix} \nabla \varphi_{\mu_j}(x_k) + A_k \lambda_k \\ c(x_k) \\ \underline{X}_k \underline{Z}_k \mathbf{1} - \mu_j \mathbf{1} \\ \overline{X}_k \overline{Z}_k \mathbf{1} - \mu_j \mathbf{1} \end{bmatrix} \tag{17}$$

in which: $W_k = \nabla^2_{xx} \mathscr{L}(x_k, \lambda_k, \underline{z}_k, \overline{z}_k)$, with $\mathscr{L}(x, \lambda, \underline{z}, \overline{z}) = f(x) + c(x)^T \lambda - (x - x_{\min})^T \underline{z} - (x_{\max} - x)^T \overline{z}$ the Lagrangian function associated with NLP (14); $A_k = \nabla c(x_k)$ the constraint Jacobian; $\underline{Z}_k = \mathrm{diag}(\underline{z}_k)$, $\overline{Z}_k = \mathrm{diag}(\overline{z}_k)$, $\underline{X}_k = \mathrm{diag}(x_k - x_{\min})$, and $\overline{X}_k = \mathrm{diag}(x_{\max} - x_k)$ diagonal matrices; $\nabla \varphi_{\mu_j}(x_k) = \nabla f(x_k) - \underline{z}_k + \overline{z}_k$. In order to generate the entries of system (17), it is necessary to evaluate the cost function gradient, the constraint Jacobian, as well as the Hessian of the Lagrangian (or a suitable approximation to it). Partial derivatives can be computed numerically by finite differentiation or analytically (for simple functions). A third approach is by means of so-called *Automatic Differentiation* (or Algorithmic Differentiation) techniques, which generate a numerical representation of partial derivatives by exploiting the chain rule in a numerical environment. Different approaches exist for AD, which are tailored to the computation of first-order and second-order derivatives. The interested reader is referred to [45]. A final computation observation is reserved to the numerical solution of system (17). First, it is transformed into a symmetric (indefinite) linear system via block elimination. Then, symmetry can be exploited by symmetric LDL factorizations. Furthemore, it should be noted that in trajectory planning problems considered here (and in general in optimal control problems) the Hessian W_k and the constraint Jacobian A_k are significantly sparse and structured. Exploiting these features can reduce the solution time significantly. To this effect, the MA57 multifrontal solver [46] from the Harwell Software Library [47] is used.

D. The CasADi framework

The transcribed optimal control problem is coded in a scripting environment using the Python [48] interface to the open-source CasADi framework [5], which provides building blocks to efficiently formulate and solve large-scale optimization problems.

In the CasADi framework, symbolic expressions for objective and constraints are formed by applying overloaded mathematical operators to symbolic primitives. These expressions are represented in memory as computational graphs, in contrast to tree representations common to computer algebra systems. The graph is sorted into an in-memory algorithm which can be evaluated numerically or symbolically with an efficient stack-based virtual machine or be exported to C code. Forward and backward source-code transforming AD can be performed on such algorithm at will, such that derivatives of arbitrary order can be computed. The sparsity pattern of the constraint Jacobian is computed using hierarchical seeding [49] and its unidirectional graph coloring is used to obtain the Jacobian with a reduced number of AD sweeps [50]. Regarding expressions and algorithm inputs and outputs, everything is a sparse matrix in CasADi. Yet the underlying computational graphs may be of either a type with scalar-valued (SX) nodes or a type with matrix-valued (MX) nodes. The combined usage of these two types amounts to a checkpointing scheme [45]: low-level functions are constructed with the SX type algorithm, which is optimized for speed. These algorithms are in turn embedded into a graph of the MX type, which is optimized for memory usage, to form the expression of objective and constraints.

In the context of optimal control problems, the CasADi framework offers several advantages over other AD tools: it comes bundled with common algorithms that can be embedded into an infinitely differentiable computional graph (e.g. numerical integrators, root-finding and linear solvers), and takes care of constructing and passing sensitivity information to various NLP solvers backends. Since CasADi does not impose an OCP solution strategy and allows fine-grained speed-memory trade-offs, it is suited more than black-box OCP solvers to explore non-standard optimal control problem formulations or efficient solution strategies.

VI. APPLICATION EXAMPLES

A. Environment-aware manipulation

1) Penalty-based approach with disk and two independent fingers: Figure 3 shows a first example of EC-exploiting manipulation. In a vertical plane, the two independent fingers H_0 and H_1, initially away from the circular object, must interact with the object and have it interact with the environment (edges e_0 and e_1) so that it will be in the shown final

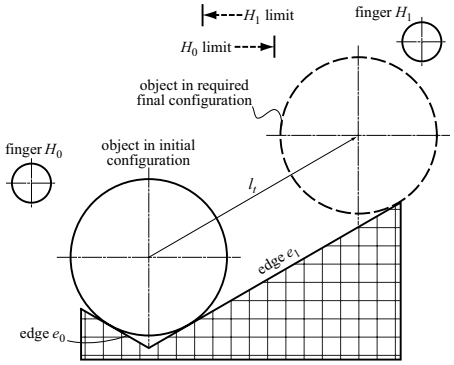


Fig. 3. EC manipulation scenario: the object must reach its final configuration at the prescribed final time $T = 8$ s.

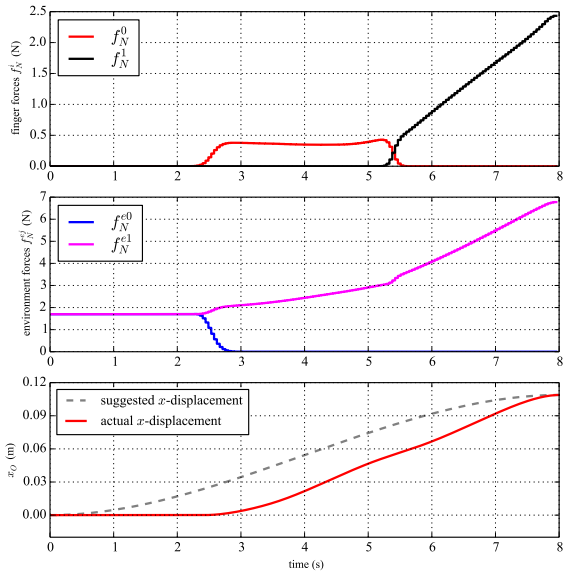


Fig. 4. Trajectories for a circular object manipulated by two independent fingers: normal contact forces f_N^i applied by the fingers; normal contact forces f_N^{ej} applied by the environment (segments e_0, e_1); suggested and actual x-displacement x of the object.

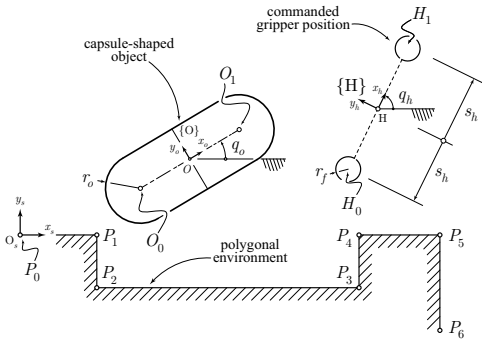


Fig. 5. EC manipulation scenario. Starting at $q_O(0) = \pi/2$, in contact with segment P_2P_3, the capsule must be placed, at time $T = 5$ s, in the same position but with $q_O(T) = -\pi/2$.

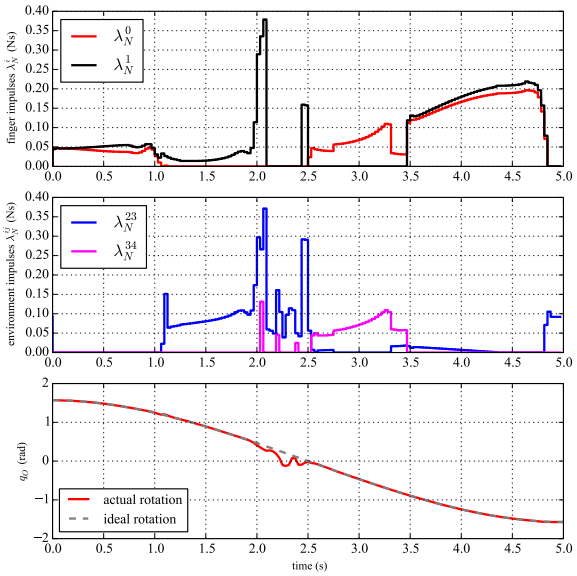


Fig. 6. Motion trajectories of a capsule-shaped object manipulated by an underactuated gripper: contact normal impulses λ_N^0 and λ_N^1 applied by the fingers; contact normal impulses λ_N^{ij} applied by the environment through segment P_iP_j; ideal and actual rotation q_O of the object-fixed frame.

position, with any orientation but zero velocity, at the end of a prescribed time horizon T. All contact interactions must occur without slippage (static friction). The object's initial state corresponds to a configuration of static equilibrium. The fingers have limitations on their horizontal workspace: as a result, grasping and lifting of the object is inhibited, and an environment-exploiting policy needs to be discovered in order to accomplish the task. Also, object-passing between fingers needs to emerge. This planning problem has been formulated and solved using the penalty-based approach. The resulting trajectories in terms of normal contact forces are shown in the first two plots of Fig. 4: finger H_0 approaches the object first (whose weight is symmetrically supported by e_0 and e_1), then rolls it on edge e_1 (without slipping) until it reaches its workspace limit and hands it over to finger H_1, which completes the task. Friction forces (not shown for brevity) satisfy constraint (3), where $\mu_s = 2$ was used. The third plot shows the actual x-component trajectory of the object versus a suggested trajectory, included as a hint in the objective function to facilitate convergence of the algorithm, but with a low weight (to avoid forcing such trajectory against dynamic constraints). With $N = 180$ discretization intervals (time step $h = 45$ ms) and considering the prescribed initial and terminal conditions, the problem size is $n = 8810$ decision variables. An animation of the obtained results can be found in part A.1 of the accompanying video, which also shows the grasping-and-lifting behavior that is discovered if finger workspace limitations are removed and the contact force exerted by edge e_1 is penalized.

2) Velocity-based time-stepping scheme with capsule and two-fingered underactuated gripper: With reference to Fig. 5, a capsule-shaped object, starting from an equilibrium configuration in contact with segment P_2P_3 (of a six-edged polygonal environment), has to find itself rotated by 180 deg at the end of the planning horizon T. Since the object is passive, a manipulation gait has to emerge for the gripper. Moreover, since we penalize high contact impulses and the gripper has a reduced mobility due to underactuation

— it has symmetrically closing jaws — it turns out that convenient EC-exploiting behaviors are indeed automatically synthesized by the optimizer. In fact, with reference to Fig. 6, beside finger impulses (first plot), which represent standard grasping/manipulation actions, contact interactions generated by collisions of the object with the environment (second plot) play a role of paramount importance in shaping the object motion. More in detail, with reference to part A.2 of the accompanying video, the object is initially grasped and lifted, then it is gently dropped so that it lays on segment P_2P_3 after hitting segment P_3P_4 (see the corresponding bumps in λ_N^{23} and λ_N^{34}). Then, with the circular part of the capsule pushed to corner P_3, the object is rotated with only one finger by sliding it on edges P_2P_3 and P_3P_4. Finally, both fingers grasp the object and, slightly lifting it up, they slide it on edge P_2P_3 to the initial position. It is worth noting that, with a wise exploitation of EC, the actual rotation of the object can closely follow the desired one (third plot). This condition can be violated in general, since the trajectory prescribed from the outset only constitutes a suggested behavior for some components of the system. Regarding the underlying numerical OCP, at each time step $k \in \{0,\dots,N-1\}$, the problem variables have the following dimensions: $q_k \in \mathbb{R}^{4+3}$, $v_{k+1} \in \mathbb{R}^{4+3}$, $\lambda_{N_{k+1}} \in \mathbb{R}^{6+2}$, $\lambda_{T_{k+1}} \in \mathbb{R}^{12+4}$, $\gamma_{k+1} \in \mathbb{R}^{6+2}$. With $N = 160$ discretization intervals ($h = 30$ ms) and considering the prescribed boundary conditions on the object/gripper, the problem size is $n = 7375$. An animation of the results can be found in part A.2 of the accompanying video.

B. Dexterous manipulation

With reference to Fig. 7, a circular object, starting from a configuration where it is held in equilibrium by three independent fingers in a force-closure grasp, has to find itself rotated by 360 deg at the end of the planning horizon T, with zero velocity. Since, again, the object itself is passive and each finger has workspace limitations (see Fig. 7), a relatively complex (dexterous) manipulation gait has to be discovered for the fingers. To obtain an in-place manipulation, a box constraint has also been assigned on the position of the object center. In order to obtain a relatively robust manipulation, the constraint described in section IV-A has been included to guarantee that any two fingers are always in contact. As we want no slipping between the fingers and the object during manipulation, the penalty-based approach has been used (with a coefficient of friction $\mu_s = 1.5$). The resulting optimal trajectories in terms of finger forces are shown in the first two plots of Fig. 8: intermittent contacts due to the discovered manipulation gait can be clearly seen. The third plot shows the object's actual rotation versus a smooth (third-order), suggested rotation trajectory. With $N = 200$ discretization intervals ($h = 30$ ms) and accounting for the fixed initial and terminal conditions, the problem size is $n = 12585$. An animation is provided in part B.1 of the accompanying video, while part B.2 shows the results obtained by solving the same problem with the velocity-based time-stepping scheme, where sliding between fingers and object is allowed and exploited.

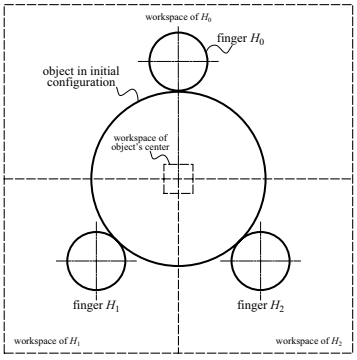


Fig. 7. Dexterous manipulation scenario: the object must find itself rotated by 360 deg at the final time $T = 6$ s.

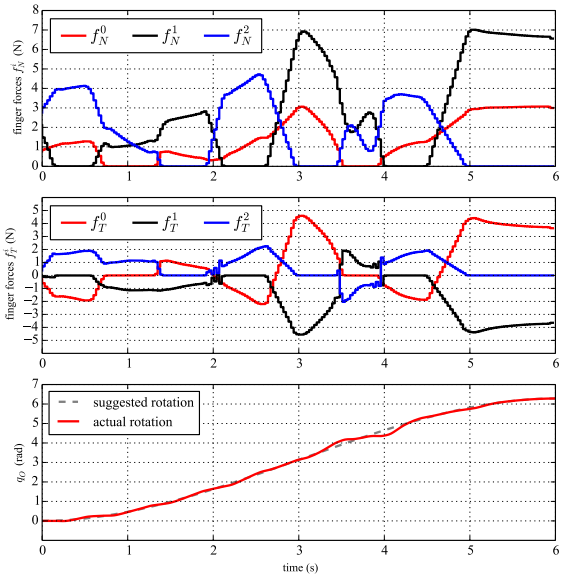


Fig. 8. Trajectories for a circular object manipulated by three independent fingers. Shown are: normal contact forces f_N^i and tangential contact forces f_T^i applied by the fingers; suggested and actual rotation q_O of the object.

VII. CONCLUSIONS AND FUTURE WORK

This paper proposed a computational framework to plan environment-aware manipulation behaviors that do not rely on an a-priori defined sequences of contacts. To this end, we framed the problem as a numerical optimal control one, including contact forces among the optimization variables as a key factor, and we sharpened the algorithmic pipeline by exploiting structural sparsity and leveraging Automatic Differentiation. Two contact models were proposed that best fit manipulation scenarios where sliding primitives need to be avoided or sought, respectively. These proved effective in solving manipulation planning problems where essential interactions with the environment had to be synthesized to accomplish a task (sub-section VI-A). The results presented in sub-section VI-B demonstrated that the very same method is able to perform successfully in discovering non-trivial gaits also in dexterous manipulation tasks. Current research is devoted to extending the method to 3D scenarios, the major thrust being the synthesis of EC-exploiting, whole-body manipulation strategies for humanoid platforms. Injec-

tion of motion primitives/synergies into the model are also being considered, and proper model scaling and tuning of IPOPT convergence parameters are under way to maximize computational efficiency.

REFERENCES

[1] J. T. Betts, *Practical Methods for Optimal Control Using Nonlinear Programming*. SIAM, 2001.

[2] D. E. Stewart and J. C. Trinkle, "An implicit time-stepping scheme for rigid body dynamics with inelastic collisions and coulomb friction," *Int. Journal for Numerical Methods in Engineering*, vol. 39, pp. 2673–2691, 1996.

[3] L. T. Biegler and V. M. Zavala, "Large-scale nonlinear programming using IPOPT: An integrating framework for enterprise-wide dynamic optimization," *Computers & Chemical Engineering*, vol. 33, no. 3, pp. 575 – 582, 2009.

[4] A. Griewank, *Evaluating Derivatives: Principles and Techniques of Algorithmic Differentiation*, ser. Frontiers in Appl. Math. Philadelphia, PA: SIAM, 2000, no. 19.

[5] J. Andersson, "A General-Purpose Software Framework for Dynamic Optimization," PhD thesis, Arenberg Doctoral School, KU Leuven, Department of Electrical Engineering (ESAT/SCD) and Optimization in Engineering Center, Kasteelpark Arenberg 10, 3001-Heverlee, Belgium, October 2013.

[6] T. Lozano-Pérez, M. T. Mason, and R. H. Taylor, "Automatic synthesis of fine-motion strategies for robots," *Int. Journal of Robotics Research (IJRR)*, vol. 3, no. 1, pp. 3–24, Mar. 1984.

[7] M. T. Mason, "The mechanics of manipulation," in *IEEE Int. Conf. on Robotics and Automation (ICRA)*, vol. 2, 1985, pp. 544–548.

[8] G. Deacon, "An attempt to raise the level of software abstraction in assembly robotics through an apposite choice of underlying mechatronics," *Journal of Intelligent and Robotic Systems*, vol. 28, no. 4, pp. 343–399, 2000.

[9] A. P. Ambler and R. J. Popplestone, "Inferring the positions of bodies from specified spatial relationships," *Artificial Intelligence*, vol. 6, pp. 157–174, 1975.

[10] C. Samson, M. L. Borgne, and B. Espiau., *Robot Control, the Task Function Approach*. Clarendon Press, Oxford, England, 1991.

[11] J. D. Schutter, T. D. Laet, J. Rutgeerts, W. Decré, R. Smits, E. Aertbeliën, K. Claes, and H. Bruyninckx, "Constraint-based task specification and estimation for sensor-based robot systems in the presence of geometric uncertainty," *International Journal of Robotics Research*, vol. 26, no. 5, pp. 433–455, 2007.

[12] I. Boyle, Y. Rong, and D. C. Brown, "A review and analysis of current computer-aided fixture design approaches," *Robotics and Computer-Integrated Manufacturing*, vol. 27, no. 1, pp. 1–12, Feb. 2011. [Online]. Available: http://linkinghub.elsevier.com/retrieve/pii/S0736584510000499

[13] G. Boothroyd, C. Poli, and L. Murch, "Handbook of feeding and orienting techniques for small parts," University of Massachusetts at Amherst, Dept. of Mechanical Engineering, Automation Project, 1981.

[14] R. Diankov, "Automated construction of robotic manipulation programs," PhD thesis, Carnegie Mellon University, Robotics Institute, August 2010.

[15] A. Miller and P. Allen, "Graspit! a versatile simulator for robotic grasping," *IEEE Robotics and Automation Magazine*, vol. 11, no. 4, pp. 110–122, 2004.

[16] M. Bonilla, E. Farnioli, C. Piazza, M. Catalano, G. Grioli, M. Garabini, M. Gabiccini, and A. Bicchi, "Grasping with soft hands," in *IEEE-RAS Int. Conf. on Humanoid Robots (Humanoids)*, 2014.

[17] M. R. Dogar and S. S. Srinivasa, "A planning framework for non-prehensile manipulation under clutter and uncertainty," *Autonomous Robots*, vol. 33, no. 3, pp. 217–236, June 2012.

[18] N. Dafle, A. Rodriguez, R. Paolini, B. Tang, S. Srinivasa, M. Erdmann, M. Mason, I. Lundberg, H. Staab, and T. Fuhlbrigge, "Regrasping objects using extrinsic dexterity," in *IEEE Int. Conf. on Robotics and Automation (ICRA)*, May 2014, pp. 2560–2560.

[19] M. Kazemi, J.-S. Valois, J. Bagnell, and N. Pollard, "Human-inspired force compliant grasping primitives," *Autonomous Robots*, pp. 1–17, 2014. [Online]. Available: http://dx.doi.org/10.1007/s10514-014-9389-9

[20] S. Karaman and E. Frazzoli, "Sampling-based algorithms for optimal motion planning," *International Journal of Robotics Research*, vol. 30, no. 7, pp. 846–894, June 2011.

[21] E. Glassman and R. Tedrake, "A quadratic regulator-based heuristic for rapidly exploring state space," in *IEEE Int. Conf. on Robotics and Automation (ICRA)*, Anchorage, Alaska, USA, 5 2010, pp. 5021–5028.

[22] A. Perez, R. Platt, G. Konidaris, L. Kaelbling, and T. Lozano-Perez, "LQR-RRT*: Optimal sampling-based motion planning with automatically derived extension heuristics," in *Robotics and Automation (ICRA), 2012 IEEE International Conference on*, 2012.

[23] Y. Koga and J.-C. Latombe, "On multi-arm manipulation planning," in *IEEE Int. Conf. on Robotics and Automation (ICRA)*, 1994.

[24] B. Cohen, M. Phillips, and M. Likhachev, "Planning single-arm manipulations with n-arm robots," in *Robotics: Science and Systems (RSS)*, 2014.

[25] J. Kober, J. A. Bagnell, and J. Peters, "Reinforcement learning in robotics: A survey," *Int. Journal of Robotics Research (IJRR)*, vol. 32, no. 11, pp. 1238–1274, 2013.

[26] N. Kohl and P. Stone, "Policy gradient reinforcement learning for fast quadrupedal locomotion," in *IEEE Int. Conf. on Intelligent Robots and Systems (IROS)*, 2004.

[27] R. Tedrake, T. Zhang, and H. Seung, "Stochastic policy gradient reinforcement learning on a simple 3d biped," in *IEEE Int. Conf. on Intelligent Robots and Systems (IROS)*, 2004.

[28] J. Peters and S. Schaal, "Reinforcement learning of motor skills with policy gradients," *Neural Networks*, vol. 21, no. 4, pp. 682–697, 2008.

[29] A. Ijspeert, J. Nakanishi, and S. Schaal, "Learning attractor landscapes for learning motor primitives," 2003.

[30] R. Jonschkowski and O. Brock, "State representation learning in robotics: Using prior knowledge about physical interaction," in *Proceedings of Robotics: Science and Systems*, Berkeley, USA, July 2014.

[31] S. Levine and P. Abbeel, "Learning neural network policies with guided policy search under unknown dynamics," in *Neural Information Processing Systems (NIPS)*, 2014.

[32] S. Levine, N. Wagener, and P. Abbeel, "Learning contact-rich manipulation skills with guided policy search," 2015, under review.

[33] G. Schultz and K. Mombaur, "Modeling and optimal control of human-like running," *IEEE/ASME Transactions on Mechatronics*, vol. 15, no. 5, pp. 783–792, 2010.

[34] W. Xi and C. Remy, "Optimal gaits and motions for legged robots," in *IEEE Int. Conf. on Intelligent Robots and Systems (IROS)*, 2014, pp. 3259–3265.

[35] I. Mordatch, E. Todorov, and Z. Popović, "Discovery of complex behaviors through contact-invariant optimization," in *ACM Transactions on Graphics*, 2012.

[36] I. Mordatch, Z. Popović, and E. Todorov, "Contact-invariant optimization for hand manipulation," in *Eurographics/ACM Symposium on Computer Animation*, 2012.

[37] I. Mordatch and E. Todorov, "Combining the benefits of function approximation and trajectory optimization," in *Proceedings of Robotics: Science and Systems*, Berkeley, USA, July 2014.

[38] M. Posa, C. Cantu, and R. Tedrake, "A direct method for trajectory optimization of rigid bodies through contact," *Int. Journal of Robotics Research (IJRR)*, vol. 33, no. 1, pp. 69–81, 2014.

[39] P. Wriggers, *Computational contact mechanics*. Berlin, New York: Springer, 2006. [Online]. Available: http://opac.inria.fr/record=b1120968

[40] M. P. Do Carmo, *Differential Geometry of Curves and Surfaces*. Englewood Cliffs: Prentice Hall, 1976.

[41] D. Stewart and J. C. Trinkle, "An implicit time-stepping scheme for rigid body dynamics with coulomb friction," in *IEEE Int. Conf. on Robotics and Automation (ICRA)*, vol. 1. IEEE, 2000, pp. 162–169.

[42] C. Glocker and C. Studer, "Formulation and preparation for numerical evaluation of linear complementarity systems in dynamics," *Multibody System Dynamics*, vol. 13, pp. 447–463, 2005.

[43] A. Wächter and L. T. Biegler, "On the implementation of a primal-dual interior point filter line search algorithm for large-scale nonlinear programming," *Mathematical Programming*, vol. 106, pp. 25–57, 2006.

[44] J. Nocedal and S. J. Wright, *Numerical Optimization*. New York: Springer Verlag, 2006.

[45] A. Griewank and A. Walther, *Evaluating Derivatives*, 2nd ed. SIAM, 2008.

[46] I. S. Duff, "Ma57 – a code for the solution of sparse symmetric definite and indefinite systems," *ACM Transactions on Mathematical Software (TOMS)*, vol. 30, no. 2, pp. 118–144, 2004.

[47] HSL, "A collection of fortran codes for large scale scientific computation." 2014. [Online]. Available: http://www.hsl.rl.ac.uk

[48] T. E. Oliphant, "Python for scientific computing," *Computing in Science & Engineering*, vol. 9, no. 3, pp. 10–20, 2007.

[49] J. Gillis and M. Diehl, "Hierarchical seeding for efficient sparsity pattern recovery in automatic differentiation," in *CSC14: The Sixth SIAM Workshop on Combinatorial Scientific Computing*, 2014.

[50] A. H. Gebremedhin, F. Manne, and A. Pothen, "What color is your Jacobian? Graph coloring for computing derivatives," *SIAM Review*, vol. 47, pp. 629–705, 2005.

APPENDIX

A. Transforming problem (13) into (14)

When all components of $g_{\min}$ and $g_{\max}$ are finite and $g_{\min} < g_{\max}$ we can write

$$g_{\min} \le g(\boldsymbol{v}) \Leftrightarrow \begin{cases} g_{\min} + s_{\min} - g(\boldsymbol{v}) = 0 \\ s_{\min} \ge 0 \end{cases} \tag{18a}$$

$$g(\boldsymbol{v}) \le g_{\max} \Leftrightarrow \begin{cases} g(\boldsymbol{v}) + s_{\max} - g_{\max} = 0 \\ s_{\max} \ge 0 \end{cases} \tag{18b}$$

Hence

$$x = \begin{bmatrix} \boldsymbol{v} \\ s_{\min} \\ s_{\max} \end{bmatrix}, \quad x_{\min} = \begin{bmatrix} v_{\min} \\ 0 \\ 0 \end{bmatrix}, \quad x_{\max} = \begin{bmatrix} v_{\max} \\ \infty \mathbf{1} \\ \infty \mathbf{1} \end{bmatrix} \tag{18c}$$

$$c(x) = \begin{bmatrix} g_{\min} + s_{\min} - g(\boldsymbol{v}) \\ g(\boldsymbol{v}) + s_{\max} - g_{\max} \end{bmatrix} \tag{18d}$$

When some components of $g_{\min}$ or $g_{\max}$ are unbounded, those constraints are discarded and no slack components are necessary. Similarly, when for some i, there holds $g^{(i)}_{\min} = g^{(i)}_{\max}$, the transformation of $g^{(i)}_{\min} \le g(\boldsymbol{v}) \le g^{(i)}_{\max}$ into a component of the equality constraint vector $c(x)$ is obvious, and again no slack component is necessary.

B. Further comments to the Interior-Point method discussed in V-B

We observe that the inner loop (Steps 2–4) is exited when the current iterate satisfies $E_{\mu_j}(\xi_{k+1}) \le \kappa\mu_j$. Then the value of μ_j is reduced in Step 5 performing an iteration of the outer loop (Steps 2–5). This implies that the solution of the inner loop becomes more and more accurate as the outer loop is executed. We also notice that $E_0(\cdot)$, i.e. $E_\mu(\cdot)$ with $\mu = 0$, represents the error of the KKT system for NLP (14). Hence, the overall algorithm is terminated when the KKT system is satisfied within a tolerance ε, specified by the user.

A crucial part of an effective interior-point algorithm is the line search that is performed in Step 2 using backtracking, i.e. starting from a large candidate value for α_k and reducing it when suitable conditions for acceptance are not satisfied. In general a step is acceptable if it makes some progress towards feasibility, i.e. reducing $\|c(x)\|$, and/or achieves an improvement in the objective function $\varphi_{\mu_j}(x)$. This is typically achieved in IPOPT by also including, in the line search, a so-called second-order correction Newton step which aims to solve solely $c(x) = 0$. In particular situations, the achieved value of α_k is too small, and this forces the algorithm to enter a *restoration* phase. For details and reasoning of this phase the interested reader is referred to [43].

C. Symmetric linear system solved in place of (17)

System (17) is non-symmetric; IPOPT performs elimination of the last two row blocks (and elimination of p^z_k and $p^{\bar{z}}_k$) obtaining the following symmetric system to be solved

$$\begin{bmatrix} W_k + \underline{\Sigma}_k + \overline{\Sigma}_k & A_k \\ A_k^T & 0 \end{bmatrix} \begin{bmatrix} p^x_k \\ p^\lambda_k \end{bmatrix} = - \begin{bmatrix} \nabla\varphi_{\mu_j}(x_k) + A_k\lambda_k \\ c(x_k) \end{bmatrix} \tag{19}$$

in which $\underline{\Sigma}_k$ and $\overline{\Sigma}_k$ are diagonal matrices with entries

$$\sigma^{(i)}_{\min} = \begin{cases} \underline{z}^{(i)} / (x^{(i)} - x^{(i)}_{\min}) & \text{if } i \in I_{\min} \\ 0 & \text{if } i \notin I_{\min} \end{cases} \tag{20a}$$

$$\sigma^{(i)}_{\max} = \begin{cases} \overline{z}^{(i)} / (x^{(i)}_{\max} - x^{(i)}) & \text{if } i \in I_{\max} \\ 0 & \text{if } i \notin I_{\max} \end{cases} \tag{20b}$$

APPENDIX

A. Transforming problem (13) into (14)

When all components of $g_{\min}$ and $g_{\max}$ are finite and $g_{\min} < g_{\max}$ we can write

$$g_{\min} \le g(\boldsymbol{v}) \Leftrightarrow \begin{cases} g_{\min} + s_{\min} - g(\boldsymbol{v}) = 0 \\ s_{\min} \ge 0 \end{cases} \tag{18a}$$

$$g(\boldsymbol{v}) \le g_{\max} \Leftrightarrow \begin{cases} g(\boldsymbol{v}) + s_{\max} - g_{\max} = 0 \\ s_{\max} \ge 0 \end{cases} \tag{18b}$$

Hence

$$x = \begin{bmatrix} \boldsymbol{v} \\ s_{\min} \\ s_{\max} \end{bmatrix}, \quad x_{\min} = \begin{bmatrix} v_{\min} \\ 0 \\ 0 \end{bmatrix}, \quad x_{\max} = \begin{bmatrix} v_{\max} \\ \infty \mathbf{1} \\ \infty \mathbf{1} \end{bmatrix} \tag{18c}$$

$$c(x) = \begin{bmatrix} g_{\min} + s_{\min} - g(\boldsymbol{v}) \\ g(\boldsymbol{v}) + s_{\max} - g_{\max} \end{bmatrix} \tag{18d}$$

When some components of $g_{\min}$ or $g_{\max}$ are unbounded, those constraints are discarded and no slack components are necessary. Similarly, when for some i, there holds $g^{(i)}_{\min} = g^{(i)}_{\max}$, the transformation of $g^{(i)}_{\min} \le g(\boldsymbol{v}) \le g^{(i)}_{\max}$ into a component of the equality constraint vector $c(x)$ is obvious, and again no slack component is necessary.

B. Further comments to the Interior-Point method discussed in V-B

We observe that the inner loop (Steps 2–4) is exited when the current iterate satisfies $E_{\mu_j}(\xi_{k+1}) \le \kappa\mu_j$. Then the value of μ_j is reduced in Step 5 performing an iteration of the outer loop (Steps 2–5). This implies that the solution of the inner loop becomes more and more accurate as the outer loop is executed. We also notice that $E_0(\cdot)$, i.e. $E_\mu(\cdot)$ with $\mu = 0$, represents the error of the KKT system for NLP (14). Hence, the overall algorithm is terminated when the KKT system is satisfied within a tolerance ε, specified by the user.

A crucial part of an effective interior-point algorithm is the line search that is performed in Step 2 using backtracking, i.e. starting from a large candidate value for α_k and reducing it when suitable conditions for acceptance are not satisfied. In general a step is acceptable if it makes some progress towards feasibility, i.e. reducing $\|c(x)\|$, and/or achieves an improvement in the objective function $\varphi_{\mu_j}(x)$. This is typically achieved in IPOPT by also including, in the line search, a so-called second-order correction Newton step which aims to solve solely $c(x) = 0$. In particular situations, the achieved value of α_k is too small, and this forces the algorithm to enter a *restoration* phase. For details and reasoning of this phase the interested reader is referred to [43].

C. Symmetric linear system solved in place of (17)

System (17) is non-symmetric; IPOPT performs elimination of the last two row blocks (and elimination of p^z_k and $p^{\bar{z}}_k$) obtaining the following symmetric system to be solved

$$\begin{bmatrix} W_k + \underline{\Sigma}_k + \overline{\Sigma}_k & A_k \\ A_k^T & 0 \end{bmatrix} \begin{bmatrix} p^x_k \\ p^\lambda_k \end{bmatrix} = - \begin{bmatrix} \nabla\varphi_{\mu_j}(x_k) + A_k\lambda_k \\ c(x_k) \end{bmatrix} \tag{19}$$

in which $\underline{\Sigma}_k$ and $\overline{\Sigma}_k$ are diagonal matrices with entries

$$\sigma^{(i)}_{\min} = \begin{cases} \underline{z}^{(i)} / (x^{(i)} - x^{(i)}_{\min}) & \text{if } i \in I_{\min} \\ 0 & \text{if } i \notin I_{\min} \end{cases} \tag{20a}$$

$$\sigma^{(i)}_{\max} = \begin{cases} \overline{z}^{(i)} / (x^{(i)}_{\max} - x^{(i)}) & \text{if } i \in I_{\max} \\ 0 & \text{if } i \notin I_{\max} \end{cases} \tag{20b}$$