

Dynamic grasp and trajectory planning for moving objects

Marturi, Naresh; Kopicki, Marek; Rastegarpanah, Alireza; Rajasekaran, Vijaykumar; Adjigble, Maxime; Stolkin, Rustam; Leonardis, Ales; Bekiroglu, Yasemin

DOI:

[10.1007/s10514-018-9799-1](https://doi.org/10.1007/s10514-018-9799-1)

License:

Creative Commons: Attribution (CC BY)

Document Version

Publisher's PDF, also known as Version of record

Citation for published version (Harvard):

Marturi, N, Kopicki, M, Rastegarpanah, A, Rajasekaran, V, Adjigble, M, Stolkin, R, Leonardis, A & Bekiroglu, Y 2018, 'Dynamic grasp and trajectory planning for moving objects', *Autonomous Robots*.
<https://doi.org/10.1007/s10514-018-9799-1>

[Link to publication on Research at Birmingham portal](#)

General rights

Unless a licence is specified above, all rights (including copyright and moral rights) in this document are retained by the authors and/or the copyright holders. The express permission of the copyright holder must be obtained for any use of this material other than for purposes permitted by law.

- Users may freely distribute the URL that is used to identify this publication.
- Users may download and/or print one copy of the publication from the University of Birmingham research portal for the purpose of private study or non-commercial research.
- User may use extracts from the document in line with the concept of 'fair dealing' under the Copyright, Designs and Patents Act 1988 (?)
- Users may not further distribute the material nor use it for the purposes of commercial gain.

Where a licence is displayed above, please note the terms and conditions of the licence govern your use of this document.

When citing, please reference the published version.

Take down policy

While the University of Birmingham exercises care and attention in making items available there are rare occasions when an item has been uploaded in error or has been deemed to be commercially or otherwise sensitive.

If you believe that this is the case for this document, please contact UBIRA@lists.bham.ac.uk providing details and we will remove access to the work immediately and investigate.



Dynamic grasp and trajectory planning for moving objects

Naresh Marturi^{1,2}  · Marek Kopicki³ · Alireza Rastegarpanah¹ · Vijaykumar Rajasekaran¹ · Maxime Adjigble¹ · Rustam Stolkin¹ · Aleš Leonardis³ · Yasemin Bekiroglu¹

Received: 1 January 2017 / Accepted: 9 August 2018
© The Author(s) 2018

Abstract

This paper shows how a robot arm can follow and grasp moving objects tracked by a vision system, as is needed when a human hands over an object to the robot during collaborative working. While the object is being arbitrarily moved by the human co-worker, a set of likely grasps, generated by a learned grasp planner, are evaluated online to generate a feasible grasp with respect to both: the current configuration of the robot respecting the target grasp; and the constraints of finding a collision-free trajectory to reach that configuration. A task-based cost function enables relaxation of motion-planning constraints, enabling the robot to continue following the object by maintaining its end-effector near to a likely pre-grasp position throughout the object's motion. We propose a method of dynamic switching between: a local planner, where the hand smoothly tracks the object, maintaining a steady relative pre-grasp pose; and a global planner, which rapidly moves the hand to a new grasp on a completely different part of the object, if the previous graspable part becomes unreachable. Various experiments are conducted using a real collaborative robot and the obtained results are discussed.

Keywords Human–robot collaboration · Grasp planning · Motion planning · Grasping · Pose tracking

1 Introduction

Autonomous grasping and manipulation of stationary objects has been studied extensively for several decades. In contrast, far less work has been done on the more challenging problems of grasping moving objects. This paper presents a

novel approach to dynamically tracking and grasping moving objects. We further demonstrate how such methods can be used to enable fundamental human–robot collaboration activities, such as a human handing over an object to the robot (Fig. 1). Additional applications might include grasping objects using a robot arm mounted on an erratically moving underwater vehicle, collaborative assembly of automotive parts etc.

This work was supported by: the Innovate UK KTP partnership 9573 between KUKA UK and University of Birmingham; EU H2020 RoMaNS, 645582; EPSRC Grant EP/M026477/1. We also acknowledge MoD/Dstl and EPSRC for supporting Aleš Leonardis' involvement in this work, via an ONR MURI project.

Marturi and Kopicki are identified as joint lead authors of this work.

Electronic supplementary material The online version of this article (<https://doi.org/10.1007/s10514-018-9799-1>) contains supplementary material, which is available to authorized users.

✉ Naresh Marturi
nareshmarturi@kuka-robotics.co.uk

¹ Extreme robotics Lab, University of Birmingham, Birmingham B15 2SE, UK

² KUKA Robotics UK Ltd., Great Western Street, Wednesbury WS10 7LL, UK

³ School of Computer Science, University of Birmingham, Birmingham B15 2TT, UK

The mainstream grasping literature has previously addressed a variety of grasping sub-problems, including: grasp synthesis based on analytical or data-driven approaches (Kopicki et al. 2016; Lenz et al. 2015; Levine et al. 2016; Roa and Suárez 2015); grasp control with tactile sensing (Romano et al. 2011); reactive grasping dealing with uncertainty in object location and shape (Hsiao et al. 2010); integrated grasp and arm motion planning (Fontanals et al. 2014a). Additionally, adaptation strategies have been proposed to increase the robustness of grasps, including: learning to maintain grasp stability based on tactile sensing Li et al. (2014); learning to adapt hand postures through human corrections (Sausser et al. 2012); using probabilistic inference to detect grasps which are likely to be unsuccessful, and then correct them (Bekiroglu et al. 2016); and human guided shared control-

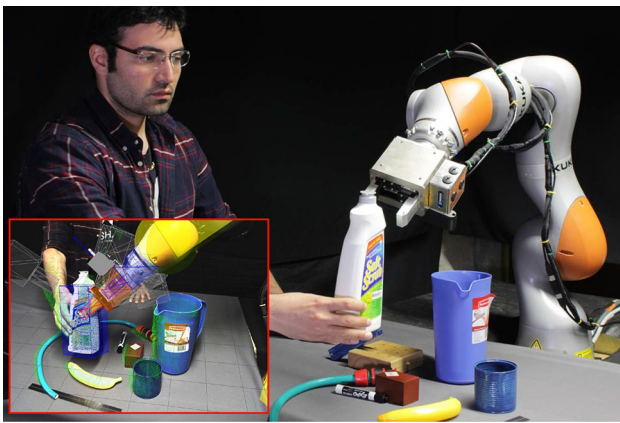


Fig. 1 A human operator handing over a test object to the robot. (Inset) feasible grasp found during the online planning task

based grasping using dynamic virtual-fixtures (Vogel et al. 2016).

In contrast to the above work, grasping of moving objects engenders a variety of difficult new challenges. An important example of this problem is that of human–robot collaboration, where a human hands an object to a robot, and the robot must track the motions of that object and continuously adapt its grasp plan in order to successfully receive the object from the human.

Note that this problem cannot be solved by simply combining a conventional vision system (for object pose tracking) with a conventional grasp planner (designed for stationary objects). While the object is moving in unpredictable ways, new grasps and new reach-to-grasp trajectories must be continually replanned on-the-fly. For an arbitrarily moving object, there will typically be periods of time during which *no feasible grasp is possible at all*. During such periods, it is desirable that the robot continues to “track” the object by keeping its hand as close as possible to a feasible pre-grasp location, even while the actual pre-grasp location is unattainable.

Additionally, even if the robot can keep its hand near to a likely grasp location on the moving object (which we refer to as “grasp tracking”), a small rotation of the object can make this grasp location suddenly disappear completely from the robot’s reachable space and/or from the camera’s field of view. This necessitates immediate replanning of a completely new grasp, for a completely new part of the object, leading to a gross discontinuity as the robot’s trajectory is replanned to reposition the hand near a new object part. We address this problem, by showing how the robot can dynamically switch between:

- (i) a local planner, which causes the hand to smoothly track the object and maintain a steady pre-grasp pose relative to the object;

- (ii) a global planner, which plans a new grasp, and new reach-to-grasp trajectory, to rapidly move the hand to a different part of the object, when the previous graspable part becomes unreachable.

1.1 Related work

A comparatively small amount of previous literature has explored the more difficult problems of grasping moving objects. Early work by Houshangi (1990) described a system for online visual servoing of a robot manipulator to grasp a moving target. This early work executed a top grasp on a cylinder, constrained to move on a flat surface while viewed from a ceiling-mounted camera, *i.e.*, the problem was reduced to 2D tracking of a circle on a plane, and execution of a pre-programmed grasp on the circle. Other early work, by Allen et al. (1993), used a PUMA robot with a parallel jaw gripper to grasp a moving toy train which was tracked by a pair of cameras. The main contribution of this work was the development of fast algorithms for computing optical flow at real-time frame rates, using the low-power computers of the time, and predictively extrapolating the motion of the toy train to overcome computational time delays. As with Houshangi (1990), the actual grasp planning in this work was relatively simplistic, and worked by simply grasping the toy train from above, while aligning the gripper’s jaws with the direction of the train’s motion. The work of Smith and Papanikolopoulos (1995) was similarly limited, with a rectangular block, moving in a straight line, being grasped from above. Other papers, *e.g.* Zhang and Buehler (1994), explore control strategies for dynamic grasping via simplified 2D simulation experiments, in which a successful grasp is defined as servoing a gripper until it has the same position, velocity and acceleration as an object to be grasped (represented as a circle).

In contrast, we use an advanced, learning-based grasp planner, which probabilistically generates a large number of likely grasps at many different locations on the surface of arbitrarily shaped objects with complex 3D geometries. In our experiments, the robot continuously replans its grasps and reach-to-grasp trajectories, while the objects are moving with complex, arbitrary 6-degrees of freedom (DoF) trajectories. As the object moves, the robot dynamically switches between several different reach-to-grasp configurations, *i.e.*, it replans trajectories to reach new parts of an object, when old parts become unreachable (due to both kinematic limitations of the robot and also the constraints of avoiding obstacles in the environment).

More recently, Kim et al. (2014) presented an impressive programming-by-demonstration approach to catching thrown objects. The system learns object flight dynamics by observing examples, and learns a distribution of feasible grasps from several human-demonstrated grasps on each

object. The system searches the space of reachable grasps to find an appropriate catching motion which intercepts the trajectory of the thrown object. Joint ranges were limited to improve the speed of trajectory planning, and grasp selection was simplified by using a heuristic of always orienting the palm of the robot's hand to be opposite to the thrown object's velocity vector. Collisions between the arm trajectory and surrounding objects were not explicitly considered. In contrast, our approach allows the robot to exploit potential grasps on any part of the object, regardless of motion direction, and reach-to-grasp trajectories are continuously re-planned (utilising the entire motion space of the robot) by avoiding self-collisions. We also exploit a learning-based grasp planner, which can generalise from a small number of examples to plan grasps on new objects with previously unencountered shapes.

Menon et al. (2014) presented a search-based kinodynamic motion planning algorithm that generates a time parametrised trajectory for both the arm and end-effector to grasp a moving object at the earliest feasible point based on graph search. The system relied on prior knowledge of the object's trajectory and pre-programmed grasps for each object. The proposed planner is unable to compute a real-time trajectory when the object is very close to the robot. In contrast, our approach utilizes a task space error to relax the goal constraints on the reach-to-grasp trajectory, which enables our robot to perform continuous "grasp tracking", always maintaining a pre-grasp pose, or a pose that is very close to the desired pre-grasp pose, throughout the object's motion. We also do not use pre-programmed grasps, and our system can autonomously generate novel grasps on new objects with previously unencountered shapes.

In most previous literature, grasp planning and arm motion planning have been seen as two separate steps. A smaller body of recent literature has started to address the problem of simultaneous grasp and motion planning. Vahrenkamp et al. (2012) proposed an RRT based planner, that combines building a feasible grasp, solving the inverse kinematics problem and determining a collision-free trajectory that brings the hand to the grasping pose, and presented evaluations in simulation. This work was limited to grasping of stationary objects, in contrast to our focus on moving objects in this paper. Additionally, the grasp planning of Vahrenkamp et al. (2012) is based on a classical force-closure and wrench-space approach, which requires accurate knowledge of physical parameters, e.g. friction coefficients, which are rarely available for real hands and real objects. In contrast, we use a probabilistic grasp planner based on learning, which is capable of generalising to new objects with unknown physical parameters.

Kitaev et al. (2015) presented a physics-based trajectory optimisation method that plans grasp approach trajectories for grasping in cluttered environments. An interesting con-

tribution of this work, which we do not consider, is the idea of planning reach-to-grasp trajectories which push aside clutter objects on the way to grasping a desired object. However, unlike our work, Kitaev et al. (2015) is aimed at grasping stationary objects, and grasping of complex shapes is also not addressed, with experiments limited to simple grasps on primitive blocks in a simulation environment.

Fontanals et al. (2014b) proposed an approach using the initial configuration of the robotic arm and the pose of the target object to simultaneously plan a hand pose and arm trajectory, where grasps are found based on independent contact regions. Again, this work does not address the problem of grasping moving objects, instead using simulation experiments with stationary objects which have simple shapes. Horowitz and Burdick (2012) formulated the combined problem of grasp contact selection, grasp force optimisation, and manipulator arm/hand trajectory planning as a problem in optimal control. This work also did not address the problem of grasping moving objects, although it does consider the finger contact forces that are needed for moving an object after it has been grasped. Simple fingertip point placements are considered, in a simplified 2D simulation environment. In contrast to the above methods, our work explicitly addresses the problem of grasping objects which are moving with arbitrarily complex 6DoF trajectories. Our method is tested in real experiments, using objects which have complex 3D shapes.

1.2 Overview of our approach

Our approach enables a robot to grasp arbitrarily moving objects (using no prior model information). It is based on learned generative grasp model, which generates offline a diverse set of possible grasp trajectories¹ for a query object, *i.e.*, the object, which robot has to follow and grasp. The object pose is estimated and updated by a particle filter-based optimised pose tracker, which is then used by a real-time iterative inverse kinematics solver (IK solver) based on differential evolution optimisation of joint angles to transform grasp trajectories into a new object frame. At any given instant, a single grasp trajectory is selected online that has the smallest task space error due to kinematic constraints and collisions. Finally, a real-time arm trajectory planner selects and executes a collision-free trajectory leading to the selected grasp trajectory.

Our approach is different from the approaches known from the literature in the following ways:

1. It relies on a generative grasp model which generates a diverse set of grasp trajectories (also called as grasp

¹ Each trajectory is represented as a sequence of waypoints from approach to grasp, where each waypoint consists of hand-wrist pose and hand-joint configuration.

hypotheses in some parts of this paper) for each new object. Typically for a moving object, only a sub-set of these trajectories can successfully be executed at any particular moment. Nevertheless, it is very likely that for most object poses this trajectory sub-set will not be empty.

- Grasp trajectories are selected using a soft measure of trajectory quality, a task space error, which can be incorporated into the robot's motion planning by solving inverse kinematics using a non-classical approach based on iterative numerical optimisation of the robot's joint angles in order to minimise the soft task space error. The robot continuously "tracks" the current grasp trajectory approach waypoint, by executing trajectories which yield the smallest feasible task space error at any instant. This technique enables handling of situations where no error-free trajectories exist, but where feasible grasps are likely to become available again in the near future.
- Our approach relies on dynamically switching between two types of planners, depending on the distance of the robot from the current destination pre-grasp position. This technique results in smooth, "local" robot movements (in terms of joint space) as long as the current task space error is not larger than a predefined threshold. However, when the current goal pre-grasp location becomes far from feasible (task space error becomes large), the robot switches to a "global" planner which triggers a gross discontinuity as the robot chooses and moves towards a new pre-grasp pose with respect to the moving object (needed because the object's motion has caused the previous desired pre-grasp pose to become unachievable). It is worth mentioning that for local trajectory planning, we use a fast anytime local planner which plans collision-free trajectories such that waypoints are guaranteed to stay close to each other in joint space. For global re-planning of new reach-to-grasp trajectories, a non real-time global planner is used to plan collision-free trajectories towards a new choice of grasp pose, that cannot be solved using the local planner.

The remainder of the paper is structured as follows. We describe the components of the system and the robotic hardware in Sect. 2 and the grasp planning algorithm in Sect. 3. Experimental results are presented in Sects. 4 and 5 provides concluding remarks.

2 System outline

Online moving object grasping method presented in the next section requires the object pose to be updated continuously throughout the task. To this extent, in this section, we present a particle filter-based method, which is capable of tracking

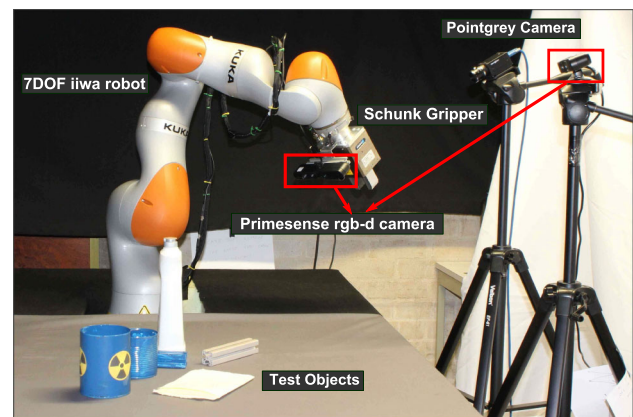


Fig. 2 Experimental setup used for this work

object's full 3D pose using partial point clouds of the objects acquired using a low-cost sensor. Initial object scanning and model building tasks are the primary initialisation blocks that are executed before initiating the tracking process. We first present our hardware set-up and then proceed with the object shape modeling and 3D pose tracking.

2.1 Robotic platform

The experimental set-up used in this work is shown in the Fig. 2. It consists of a 7DoF collaborative robot, KUKA iiwa 140 whose end-effector is equipped with a two finger parallel-jaw gripper, Schunk PG 70 with a stroke size of 68 mm. The vision system includes three different cameras: 2 PrimeSense Carmine RGB-D narrow angle sensors (task and track cameras) and 1 Point Grey Grasshopper USB-3.0 camera. The task camera mounted on top of the gripper is mainly responsible for acquiring scene point clouds, which are then processed to generate a target object cloud. Consecutively, the obtained cloud will be used to generate a set of possible grasps as well as to trigger visual object tracking. On the other hand, the track camera located in the workspace is responsible for tracking full pose of the moving objects. The Point Grey camera is used merely for recording videos of experiments, which can be augmented to aid visualisation by superimposing virtual views of possible grasps being considered by the planner onto the camera images. All the three cameras are calibrated with respect to the robot base frame and are connected to the work computer (4.00 GHz Intel Core i7 CPU with 32 Gb of RAM). The communication with the robot controller (running Kuka Sunrise.OS) to transfer motion commands has been realised as a cross-platform client (C++) - server (Java) communication over UDP, where the client program runs on the work PC. The objects used for robot-human handover experiments are visually tracked with respect to the robot task space and, as a matter of safety, the robot's joint torque limits are constrained.

2.2 Object shape modeling

The objects used for training and validating our approach are shown in Fig. 3a. In this work, all these objects are represented as (partial) depth point clouds obtained by registering multiple depth clouds. Overall model generation process is comprised of two steps: scanning and stitching. During the scanning phase, robot moves to four different poses around the object. At each individual pose a single depth cloud is acquired (see first two rows of Fig. 3b) using the robot wrist mounted task camera. The robot poses are selected such that all faces of the object are maximally covered. Later during the stitching phase, all these individual depth clouds are first processed to remove the dominant ground plane. The remaining points from the processed clouds are registered together to form a model of the object. Last row of Fig. 3b show multiple views of an obtained final registered point cloud of a test object. In this case, stitching the point clouds is trivial since the exact pose of the camera is known from the robot's forward kinematics. Alternatively, the stitching accuracy can also be improved using cloud registration algorithms like iterative closest point (ICP) (Rusinkiewicz and Levoy 2001), three dimensional normal distribution (Magnusson 2009) etc. Figure 3c shows registered partial point clouds of various test objects. Since our algorithm, presented in the next section, is capable of working robustly on partial point clouds, no efforts were made in this work to interpolate or smooth these partial clouds.

2.3 Object pose tracking

Visual tracking of the object's 6DoF pose plays an important role in planning and controlling the robot trajectories. In this work, we use a particle filter-based 3D tracker to track full 6DoF pose of a moving object in order to generate both grasp and motion plans (Aldoma et al. 2012). Even though this type of visual object tracking is not new, many researchers in the fields of computer vision and robotics have previously proved their robustness and adaptivity in case of solving challenging problems (Fukui et al. 2016; Marturi et al. 2015; Wu et al. 2013). Nevertheless, it is worth mentioning that due to the highly modular nature of our implementation, the current pose tracker can be easily replaced with another one e.g. a CAD model-based tracker (Ortenzi et al. 2016; Tamadazte et al. 2010) or a coupled layer pose tracker (Ma et al. 2016).

In this work, the tracker is initialised using the registered object's point cloud. Hereafter, for each new cloud, the tracker uses the tracked pose obtained from the previous frame to initialise, and propagates a uniform sample of N particles, where each particle represents a candidate 6DoF pose of the object. Likelihoods for each particle are computed according to the degree of fit between a reference point cloud (for first cloud in process, this will be the regis-

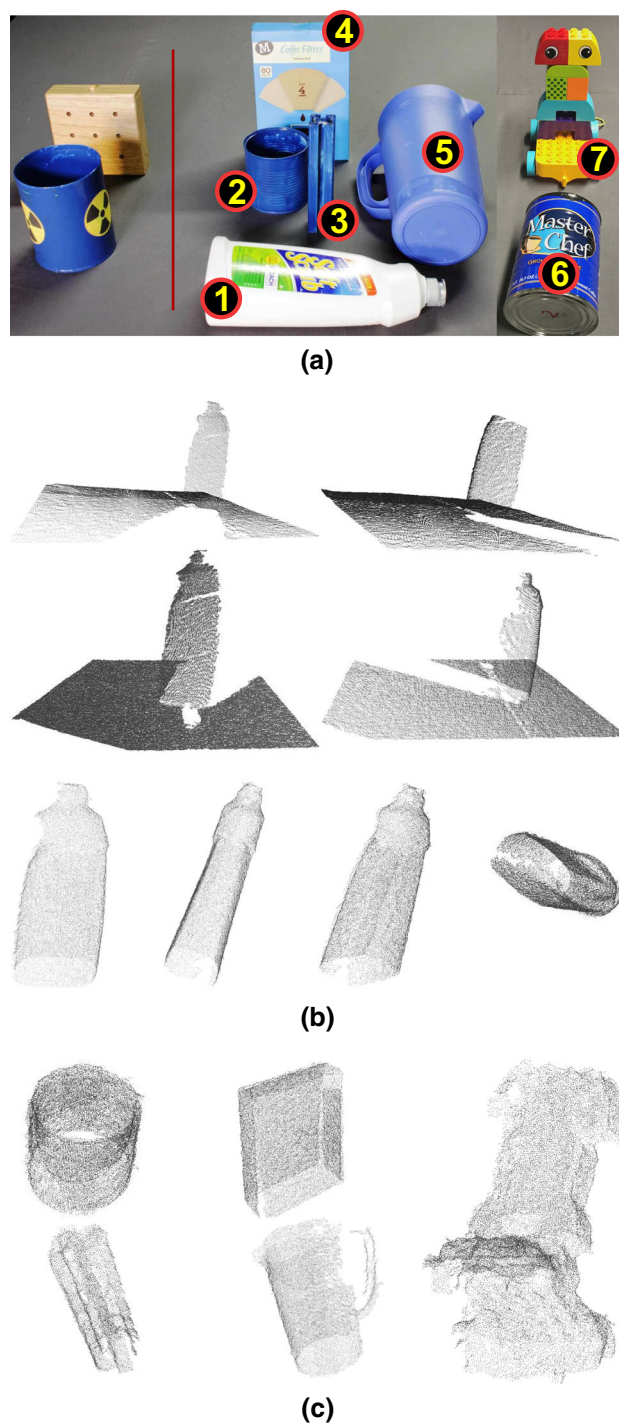


Fig. 3 **a** Objects used in this work. Red line differentiates train and test objects. The two objects on the left are used for training and on the right are used for validation, as labeled: (1) dishwasher, (2) can (3) bar (4) coffee box (5) jug (6) big can (7) lego toy. **b** Illustration of object modeling. Point clouds obtained from four scanning poses are shown in the first two rows. Multiple views of final segmented and registered partial cloud of a test object are shown in the third row. **c** Partial point clouds of remaining test objects (Color figure online)

tered cloud from object modeling step) and the input cloud obtained from the track camera, using the distance likelihood function $\mathcal{L}_{dist}(r_j, c_j)$ given by (1):

$$\mathcal{L}_{dist}(r_j, c_j) = \frac{1}{1 + |r_j - c_j|^2} \quad (1)$$

where, r_j and c_j are the j th points in the reference and current (input) point clouds, respectively. Weights with each particle $\omega_i |_{i=1 \dots N}$ are thus computed using (2)

$$\omega_i = \sum_j \mathcal{L}_{dist}(r_j, c_j) \quad (2)$$

The particle with the maximum weight will be retained as the reference cloud for the next iteration and its pose will be used as an input for the robot's motion planning module.

In order to speed-up the overall process, we parallelized the likelihood computation over multiple threads on a CPU. In other cases, such as high volume tracking, the process speed can be further improved by implementing these computation routines over a GPU as shown by Choi and Christensen (2013). Besides, the speed of the tracking also depends on the number of particles to track and the size of the target object, *i.e.*, the density of the points in the clouds. For high volume clouds, the tracking speed can be increased by downsampling the points and by reducing the number of particles. However, this can affect the overall visual tracking accuracy (in turn the grasp selection accuracy). Hence a good trade-off needs to be achieved beforehand. In this work, by trial and error based on failure rate, we selected a downsampling size of 0.5 cm and the number of particles to track as $(\frac{S}{75})$, where S is the total number of points in the cloud. Alternatively, this selection process can be automated using adaptive version of the particle filter proposed by Soto (2015).

3 Online grasping algorithm

The proposed online grasping algorithm is summarized in Algorithm 1. Here we consider tracking (arm following the object) and grasping of each object as a trial. Each trial consists of five different phases: (i) grasp generation, (ii) tracking, (iii) selection, (iv) planning and (v) grasp execution. Phases (ii)–(iv) are repeated until a human co-worker gives the robot permission to enter the grasp execution phase (v). Each of these phases is detailed below.

3.1 Grasp generation

At the beginning of each experimental trial, for a stationary object (registered depth point cloud), a number of grasp tra-

Algorithm 1 Moving object grasping algorithm

```

1:  $\mathcal{C}^{\mathcal{I}} = \text{capture}()$ 
2: Get object's initial pose  $v^{\mathcal{C}^{\mathcal{I}}} = \text{track}(\mathcal{C}^{\mathcal{I}})$ 
    $\triangleright$  Generate initial grasp trajectories and select first  $\mathcal{N}$  ones
3:  $\{\mathcal{T}_j^{v^w}\}_{j=1 \dots \mathcal{N}} = \text{generate}(\mathcal{C}^{\mathcal{I}})$ 
    $\triangleright$  Set current trajectory error  $\epsilon_r^*$  and index  $r$  to infinity
4:  $\epsilon_r^* \leftarrow \infty, r \leftarrow \infty$ 
5: while tracking() do
6:  $\mathcal{C} = \text{capture}()$ 
7: Get object's current pose  $v^{\mathcal{C}} = \text{track}(\mathcal{C})$ 
8: Compute object pose update  $s = v^{\mathcal{C}}(v^{\mathcal{C}^{\mathcal{I}}})^{-1}$ 
9: for  $j=1$  to  $\mathcal{N}$  step 1 do
    $\triangleright$  Transform grasp trajectories to cloud frame  $v^{\mathcal{C}}$ 
10:  $s \otimes \mathcal{T}_j^{v^w} = \mathcal{T}_j^{sv^w}, \mathcal{T}_j^{sv^w} \equiv \{(sv^w, c^h)_{ij}\}_{i=1 \dots m_j}$ 
    $\triangleright$  Read current arm and hand configuration
11:  $c^c \leftarrow (c^{ca}, c^{ch})$ 
    $\triangleright$  Find grasp trajectories with arm joint configurations  $c^a$ 
   and corresponding trajectory error  $\epsilon^*$ 
12:  $\{\mathcal{T}_j^{c^a}, \epsilon_j\} = \text{inverse}(c^c, \mathcal{T}_j^{sv^w})$ 
13: end for
    $\triangleright$  Select grasp trajectory with the smallest error  $\epsilon$ 
14:  $\{r, \epsilon_r^*\} = \begin{cases} k, \epsilon_k & \text{if } \epsilon_k + \delta_w < \epsilon_r^*, k = \text{argmin}_j \epsilon_j \\ r, \epsilon_r^* & \text{otherwise} \end{cases}$ 
15: Find trajectory  $\mathcal{T}_r^{c^c}$  from  $c^c$  to  $c_{1r} = (c^a, c^h)_{1r}$ 
    $\triangleright$  Use local or global planner
16:  $\mathcal{T}_r^{c^c} = \begin{cases} \text{global}(c^c, c_{1r}) & \text{if } \|c^c - c_{1r}\|^2 > \delta_c \\ \text{local}(c^c, c_{1r}) & \text{otherwise} \end{cases}$ 
    $\triangleright$  Execute trajectory  $\mathcal{T}_r^{c^c}$ 
17:  $\text{exec}(\mathcal{T}_r^{c^c})$ 
18: if permitted to grasp then
19:  $\text{exec}(\mathcal{T}_r)$ 
20: Set tracking To false
21: return
22: end if
23: end while

```

jectories $\mathcal{T}_j^{v^w}$ are generated using $\text{generate}()$ function. These grasp trajectories are sequences of waypoints (v^w, c^h) given by (3), each consisting of hand-wrist pose $v^w \in SE(3)$ and hand-joint configuration $c^h \in \mathbb{R}^{N^h}$, with N^h being the number of hand joints,

$$\mathcal{T}_j^{v^w} = \{(v^w, c^h)_i\} \quad \text{with } i = 1 \dots m_j, \quad (3)$$

where, m_j represent number of waypoints of j th trajectory.

The grasp trajectory beginning, called *approach* waypoint, has index-1 (pre-grasp). The last *grip* waypoint specifies the hand-object contact and has index m_j , which may be different for different grasp types (Kopicki et al. 2016). The grasp trajectory generator, represented by $\text{generate}()$, relies on a generative grasp learning approach introduced in our previous work, (Kopicki et al. 2014, 2016). It generates from a single or multiple-view point cloud, a number of grasp hypotheses, *i.e.*, grasp trajectories $\mathcal{T}_j^{v^w}$, where only

the first \mathcal{N} most likely ones are selected. In Sect. 4, we analyse the affect of \mathcal{N} selected hypotheses on the performance of our algorithm. The grasp hypotheses are generated by considering previously known (from training) hand configuration model and object contact model, which is the relation between a rigid link of the hand and the local object shape near its point of contact. Combinely, these two models are represented as probability distributions (kernel density estimation models). For a query object, candidate grasps are explored to find a candidate grasp that maximises the product of these distributions.

This approach does not assume any models of objects, and is robust enough even if the training and validation objects are substantially different (see Fig. 3). The grasp planning algorithm (Kopicki et al. 2016) provides a diverse set of grasp trajectories of which, only a subset is expected to be feasible at any particular moment, given moving object and static obstacles.

3.2 Tracking

The tracking phase starts after grasp generation. During this phase, a human operator starts moving the object (arbitrarily) for which a set of feasible grasps were generated during the grasp generation phase. The robotic arm “tracks” the grasp approach waypoint of a *current grasp trajectory* $\mathcal{T}_r^{v^w}$, however without executing the entire grasp trajectory, *i.e.*, the final *grasp* waypoint will not be reached during this phase.

Tracking, $\text{track}()$, relies on the previously described visual object pose tracker, which continuously estimates pose changes $s \in SE(3)$ of the moving object’s point cloud \mathcal{C} . These tracked or updated poses are then used to transform wrist poses of all generated grasp trajectory waypoints back to the object frame, as shown in (4).

$$s \otimes \mathcal{T}_j^{v^w} = \mathcal{T}_j^{sv^w} \quad (4)$$

where, \otimes denotes rigid body transformation applied to the entire set of waypoints of the grasp trajectories (see (3)), and s is the object pose update (given by (5)) that is computed from the current point cloud pose v^c with respect to the initial or reference pose v^{c^x} ,

$$s = v^c (v^{c^x})^{-1}. \quad (5)$$

In (4), $\mathcal{T}_j^{sv^w}$ is a set of grasp trajectories in the updated object frame and is represented as

$$\mathcal{T}_j^{sv^w} \equiv \{(sv^w, c^h)_{ij}\} \text{ with } i = 1 \dots m_j \quad (6)$$

where, sv^w are the transformed wrist poses.

3.3 Selection

In this phase, the grasp trajectory to be executed with the current object’s pose will be selected. The current grasp trajectory $\mathcal{T}_r^{c^a}$ is the r th trajectory computed and selected from the previously obtained set of grasp trajectories $\{\mathcal{T}_j^{sv^w}\}_{j=1 \dots N}$ (with N being number of hypotheses) in a two stage process as explained below.

3.3.1 Anytime iterative inverse kinematic solver

In the first stage, grasp trajectory waypoints $(sv^w, c^h)_{ij}$ with wrist poses $sv^w \in SE(3)$ are transformed by $\text{inverse}()$ into trajectories in a complete joint space of the robot,

$$\mathcal{T}_j^{c^a} = \{(c^a, c^h)_{ij}\} \text{ with } i = 1 \dots m_j \quad (7)$$

with hand-joint configuration $c^h \in \mathbb{R}^{N^h}$ and arm-joint configuration $c^a \in \mathbb{R}^{N^a}$, where N^a is the number of arm joints. Trajectories given by (7) are computed using our anytime, iterative, inverse kinematic solver (IK solver), which is based on differential evolution optimisation of joint angles, but not on, e.g. inverting or pseudo-inverting the Jacobian as in most classical methods. The complete theoretical description of the used inverse kinematics algorithm can be found in Sect. 4.2.3.4 of (Kopicki 2010).

The IK solver provides a collision-free solution $\mathcal{T}_j^{c^a}$ for any input target $\mathcal{T}_j^{sv^w}$ given various task-space constraints such as collision bounds or joint limits. This is possible because solutions are allowed to be inaccurate with accuracy measured by task space errors (see Eq. (8)). The algorithm favours solutions which waypoints c_{ij} are close to the current robot configuration c^c . Furthermore, the algorithm can be interrupted after a specific amount of time to preserve fixed duration of the robot control cycle in phases (ii) to (iv).

3.3.2 Error-based selection

The quality of inverse kinematic solutions are measured by a *task space error*, which compares the desired wrist pose sv^w to the pose computed by forward kinematics map $\mathcal{K}^{for} : \mathbb{R}^{N^a} \rightarrow SE(3)$ from the approximate solution \tilde{c}^a ,

$$\epsilon_{ij} = (1 - a) \|\text{lin}(sv_{ij}^w) - \text{lin}(\mathcal{K}^{for}(\tilde{c}_{ij}^a))\|^2 + a (1 - |\text{ang}(sv_{ij}^w) \cdot (\text{ang}(\mathcal{K}^{for}(\tilde{c}_{ij}^a)))^{-1}|) \quad (8)$$

where $\text{lin}(\cdot)$ extracts the argument \mathbb{R}^3 translation vector, $\text{ang}(\cdot)$ extracts the argument $SO(3)$ quaternion, $\|\cdot\|$ is the Euclidean norm, \cdot is the quaternion dot product, and $a \in \langle 0 \dots 1 \rangle$ is a coefficient weighting linear and angular distances.

The overall j -th trajectory error ϵ_j is a sum of task space errors for of all waypoints,

$$\epsilon_j = \frac{1}{m_j} \sum_{i=1}^{m_j} \epsilon_{ij} \quad (9)$$

After computing all grasp trajectories together with corresponding errors $\{\mathcal{T}_j^{c^a}, \epsilon_j\}_{j=1 \dots \mathcal{N}}$, the k -th trajectory with the least trajectory error ϵ_k is found,

$$\{r, \epsilon_r^*\} = \begin{cases} k, \epsilon_k & \text{if } \epsilon_k + \delta_w < \epsilon_r^*, \text{ where } k = \arg \min \epsilon_j \\ r, \epsilon_r^* & \text{otherwise} \end{cases} \quad (10)$$

The k -th trajectory becomes current grasp trajectory r , if its error ϵ_k is smaller than previous least error ϵ_r^* with additional offset $\delta_w \in \mathbb{R}^+$. The offset δ_w prevents too frequent trajectory selection and is selected by trial and error. For the experiments reported in this paper, we used a value of $\delta_w = 2.0$.

3.4 Planning

This fourth phase of our algorithm involves finding a *tracking trajectory* $\mathcal{T}_r^{c^c}$ from the current robot configuration c^c to the approach waypoint c_{1r} of the current grasp trajectory \mathcal{T}_r , selected from the previous phase. This is then executed immediately.

3.4.1 Tracking trajectory planning

Tracking trajectory $\mathcal{T}_r^{c^c}$ is found using one of the two trajectory planners (see details in Sect. 4.4.4 of Kopicki (2010)):

- local iterative optimisation-based local planner, represented as `local()`; (or)
- probabilistic roadmap (PRM) based global planner, represented as `global()`.

Both the planners find a collision-free trajectory $\mathcal{T}_r^{c^c}$ from the current robot configuration c^c to the target approach waypoint c_{1r} . Local planner guarantees that all trajectory waypoints $\{c_i\}$ lie in some neighbourhood of c^c and c_{1r} . Global planner does not guarantee this property, albeit it tries to find the shortest path from c^c to c_{1r} . The decision whether to use global or local planner is made based on the Euclidean distance between c^c and c_{1r} ,

$$\mathcal{T}_r^{c^c} = \begin{cases} \text{global}(c^c, c_{1r}) & \text{if } \|c^c - c_{1r}\|^2 > \delta_c \\ \text{local}(c^c, c_{1r}) & \text{otherwise} \end{cases} \quad (11)$$

where $\delta_c \in \mathbb{R}^+$ is a threshold which controls switching of the global and local planning. Typical values of δ_c can be in between 0 and 1, where smaller values allow faster switching.

3.4.2 Tracking trajectory execution

Robot performs tracking trajectory $\mathcal{T}_r^{c^c}$ at the end of each control cycle. In this way robot tries to stay at the approach waypoint of the current grasp trajectory until tracking is interrupted.

3.5 Grasp execution

In the current implementation, also for demonstration purposes, a human operator decides when to interrupt tracking, *i.e.*, to stop arm following the object and execute the final stage of the current grasp trajectory \mathcal{T}_r , which means to reach the final grasp waypoint. The online grasping algorithm guarantees that \mathcal{T}_r is a trajectory with the task space error ϵ_r^* no larger than $\epsilon_k + \delta_w$, where ϵ_k is the smallest trajectory error among all grasp trajectories (see Eq. (9)).

4 Experimental evaluation

Various experiments are conducted using the system presented in Sect. 2 to evaluate the performance of our proposed moving objects adaptive grasping approach. Validation has been performed considering different aspects of the algorithm. Firstly, we evaluate the performance of our approach qualitatively in the scenario where a human handovers different test objects to the robot, which is also the central objective of this work. Main motivation behind these tests is two-fold; to find out the suitability of objects that can be used with our current implementation (especially visual tracking) and to analyse the system performance in terms of repeatability. Next, we evaluate the effect of various parameters in a semi-controlled manner, where again a human demonstrator moves the objects, but in a pre-defined systematic way. It is worth noting that the human operator is well-trained in using the industrial robots and possess extensive knowledge of the developed system. Before proceeding with the experimental analysis, we begin this section detailing the training process as well as with an analysis of the used visual tracker.

4.1 Grasp training

The grasp training has been performed as follows. Initially we obtain the stitched point clouds of the two training objects (wooden block and a can) depicted in Fig. 3 by scanning them from four different views. Using our implemented interface, a human operator then demonstrates each possible grasp kinesthetically on the obtained point cloud. During this process,

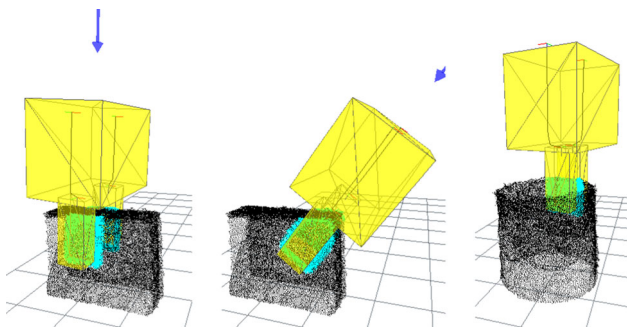


Fig. 4 Sample screenshots obtained during the training process, where an operator demonstrates the possible grasps kinesthetically using our software interface. Final grip configuration of the robotic parallel gripper with contact regions (in cyan) can be seen in the images (Color figure online)

different hand poses are recorded along the trajectory from approach configuration to the final grip (contact) as shown in Fig. 4. Due to simple two finger (parallel jaw) configuration of the used gripper, only *pinch* and *power* grasp configurations are used in this work. The hand-joint configuration and contact models are learned from a final configuration of this trajectory, which are used for generating grasp hypotheses as explained in Sect. 3.1. An important point to note is that the training is solely performed using the kinematic information and no tactile information is used.

4.2 Analysing visual object pose tracker

From the description of our method, it is obvious that visual object pose tracking plays a vital role in the success of our approach, *i.e.*, to dynamically adapt grasp trajectories for the moving objects. In this context, we assess the performance of the implemented particle filter-based tracker irrespective of object's size, shape and colour. As mentioned before, the tracker is initialised using the supplied depth point cloud (no rgb information is used, except for visualisation) of the target object and its various parameters are set as explained in Sect. 2.3. Once initialised, the object has been moved arbitrarily by a human, where its poses are tracked and updated simultaneously. Top two rows of Fig. 5 show the results of tracking two test objects at different instances of time and the bottom row of Fig. 5 shows the trajectories followed by the objects in task space. After evaluating all the test objects over ten different tracking trials spanning 2 min, it has been determined that by using our choice of parameters, on an average we are able to process 7.62 clouds per second. Since the tracker performed reliably well in all cases, we believe its applicability for the online task planning. Nevertheless, an important point to be highlighted is target object occlusion. Even though particle filter-based trackers have been previously proved to be robust enough handling large occlusions, in this work we avoid such cases; the human is constrained

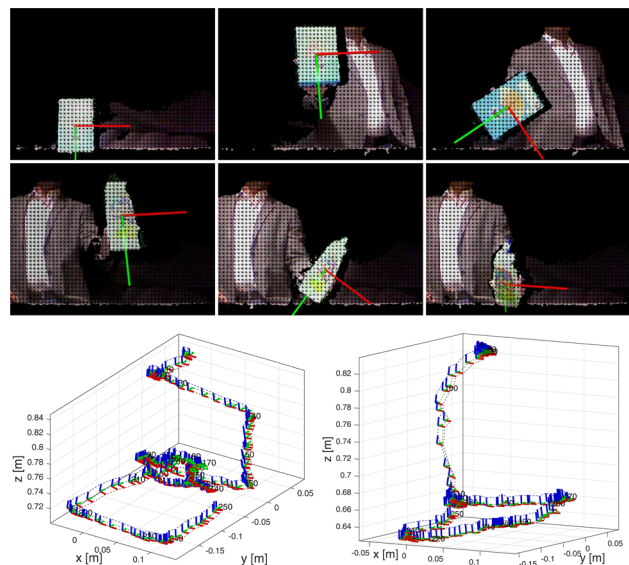


Fig. 5 The first two rows show some of the screen-shots obtained during pose tracking of (top) coffee filter box and (middle) dishwasher soap. The objects are moved randomly in the task space. Bottom row shows the trajectories followed by both these objects respectively

to hold the objects only at bottom while working with real robot. Moreover, since no object detection has been used (e.g. model matching), the tracker re-initialisation is not considered in this work. Therefore any trial matching this case is treated as a failure.

4.3 Demonstration of the online grasp planning

We conducted experiments to evaluate our grasp adaptation approach, where a human moves the test objects while they are being tracked. Figure 6 illustrates the steps during these experiments. Provided supplementary video illustrates the reported results in more detail. Weighting coefficient a (from Eq. (8)) and switching threshold δ_c (from Eq. (11)) are chosen to be 0.4 and 0.2, respectively for these tests. In the next section, we also analyse the performance with different values for δ_c .

Firstly the model point cloud of the object is obtained as explained in Sect. 2.2 and is used for building the set of initial grasp hypotheses. It is worth noting that in case of multiple objects presented in the scene, the query object, *i.e.*, the object to be followed and grasped has been selected manually by a human operator using our software interface. Whereas in case of single object, the human operator directly allows the algorithm to proceed with next steps. Out of the generated hypothesis, the first 20 are selected for these tests. In the next section, we analyse the effect of the amount of these hypothesis. Once the visual tracking starts, the selected grasps are adapted based on changed object poses so that they are still applicable according to the current object pose. The grasp

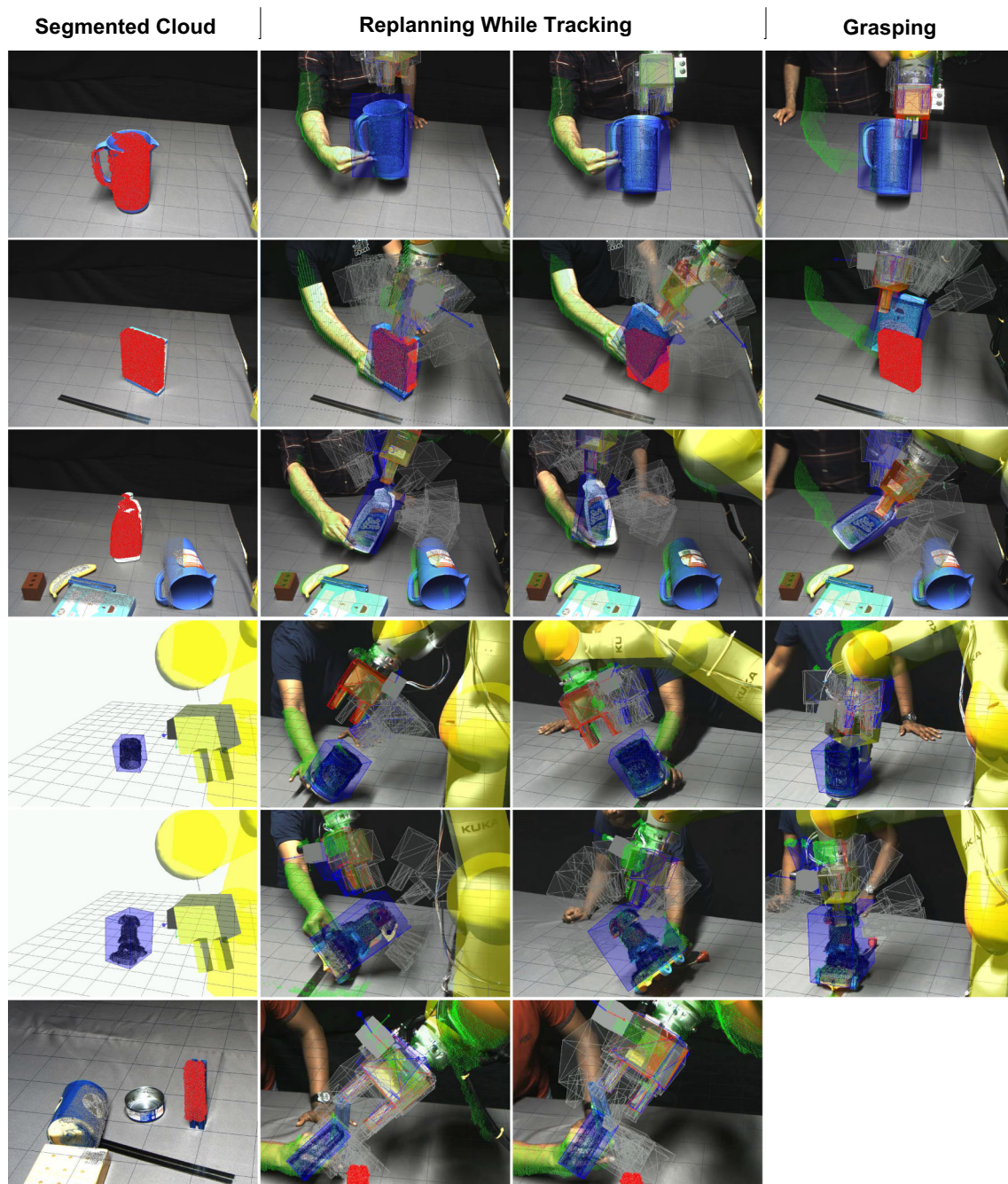


Fig. 6 Grasp adaptation experiments: the test objects are segmented out from the background and the initial grasp set is chosen based on that. Virtual models of the robot, hand and object are shown superimposed over the observed camera images of the scene. Red points show the segmented selected object depth cloud (first column). As the objects are moved, the grasp set is updated accordingly to include applicable candidates (middle three columns). Gray frames indicate all the possible

grasp candidates, red frame represents the current grasp followed by the robot and blue frame points to the next best possible candidate. The point cloud inside the blue bounding box illustrates the tracked object. Successful grasps are shown in the last column. Provided supplementary video illustrates the results in more detail. The trial with *bar* failed during rotation, where the visual tracker failed to track the object (can be seen in middle two columns of last row) (Color figure online)

set includes the grasps that correspond to the lowest error from step 14 in Algorithm 1 and the set is shown with gray wire-frames in Fig. 6. The transparent blue bounding box corresponds to the region where the grasps should be applicable. The red wire-frame corresponds to the best feasible grasp based on the error (given by Eq. (9)) and the blue one shows the next best feasible configuration. The last column in the figure shows example grasps that were executed during the trials. From our experiments, we found out that one of the used test objects: bar (labeled 3 in Fig. 3) is not suitable for the experiments. Results with bar are shown in the last row

of Fig. 6, where it can be seen that the visual tacking failed during rotation due to lack of distinguishable depth textures and symmetry of the object, leading to wrong grasp poses. Since no tracker re-initialisation is used, we consider this case as a failure.

The switching process while executing the current grasp trajectory, *i.e.*, while the arm follows the approach way-point, has been visually illustrated for two different objects in Figs. 7 and 8. It can be observed that once large errors are present, the global planner is triggered to reduce the error immediately in each trial, which clearly validates our

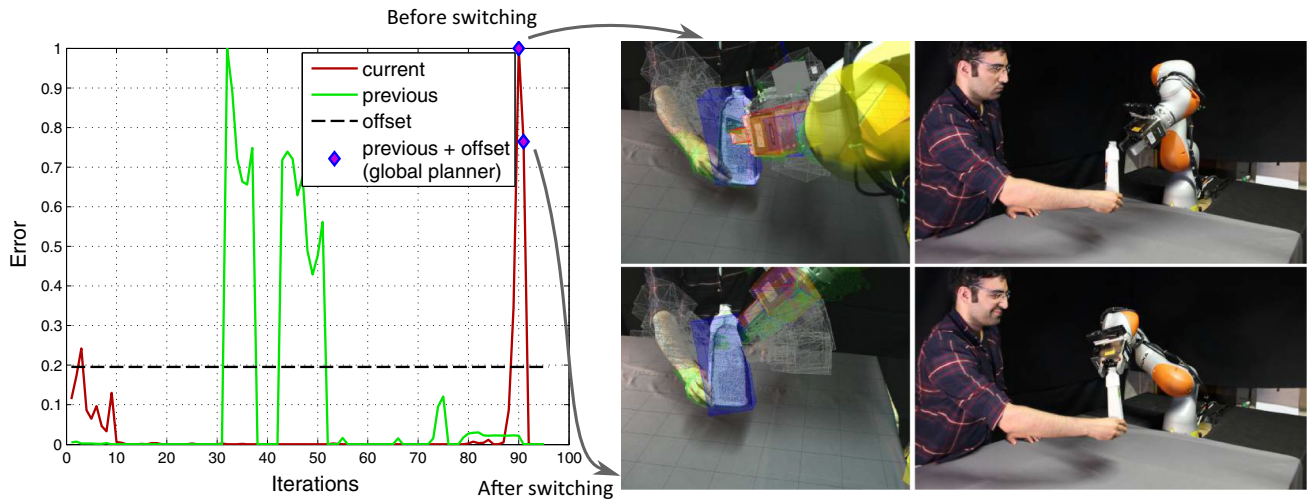


Fig. 7 Analysis of planner switching while handling dishwasher. Red curve represents currently executed grasp trajectory error and green curve shows previous error. Selected offset is shown with dotted line. As explained with Eq. (10), a trajectory becomes current grasp trajec-

tory, when its error is less than previous plus offset. The points where this switching occurred are pointed with diamonds and corresponding robot locations are shown in the images beside (Color figure online)

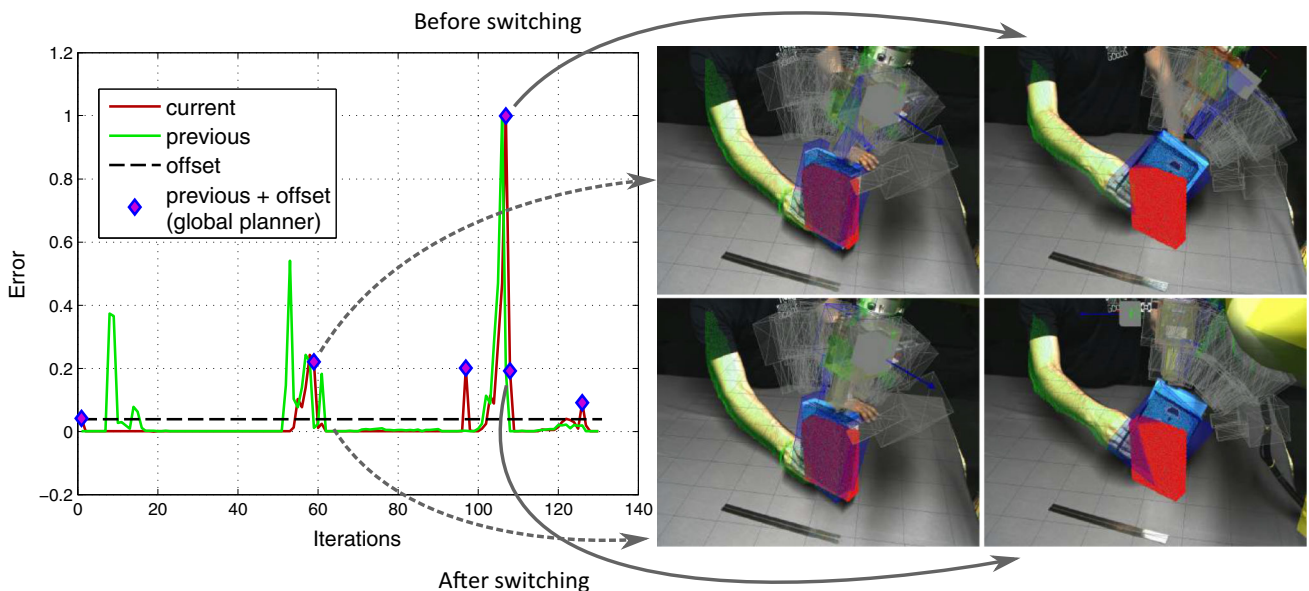


Fig. 8 Analysis of planner switching while handling coffee box

Table 1 Statistical results for repeated trials

Object	Trial	Switches	Success rate (%)
Dishwash	T-1	2	100
	T-2	3	
	T-3	3	
Can	T-1	5	90
	T-2	1	
	T-3	3	
Coffee box	T-1	6	100
	T-2	4	
	T-3	1	
Blue Jug	T-1	1	80
	T-2	6	
	T-3	1	
Big can	T-1	2	100
	T-2	1	
	T-3	1	
Lego toy	T-1	7	100
	T-2	4	
	T-3	8	

approach. This way the grasp set is updated with applicable grasps. It is worth mentioning that the used IK solver from Sect. 3.3.1 is already integrated with a collision expert due to which no self-collisions are noticed during the trials.

We repeated the experiments ten times for each test object, where a human arbitrarily moved the object in the robot workspace for about half a minute on average. Table 1 summarizes the results for the first three trials from these experiments, listing the number of switches between local and grasp planners and the success rates (for all ten trials). The resulting number of switches between the planners demonstrate the re-planning capability of the proposed approach. One trial with *can* failed due to visual tracking failure caused by self-occlusion and two experiments with *blue jug* failed as the rotation of the object caused the handle to appear again after it became hidden from the camera which led to wrong grasp poses as the object pose was inverted. It can be also noted that the number of switchings for lego toy are comparatively high due to its complexity in surface model. Overall, these results show that once the initial grasps are chosen, the proposed approach can find feasible configurations in the work-space given that the objects can be visually tracked continuously. Detailed results can also be found at <https://youtu.be/kzG-TxT4wd8>.

4.4 Analysing the effect of selected grasp hypotheses

In this section, we report the experiments performed to analyse the effect of the number of grasp hypotheses used for following and grasping moving objects. The tests are conducted in a semi-controlled environment, where a human moves the objects in a systematic way along the two tracks shown in Fig. 9. The objects are rotated along the black line and are translated along the metallic scale (30 cm both forward and backward). For the following tests, three different amounts of hypotheses: first 10, first 20 and first 30 (out of the total generated hypotheses), as shown in Fig. 9b–d are evaluated. Three different test objects are selected for these experiments based on the experimental performance achieved in the previous tests and also on the shape complexity: *big can* (labelled 6 in Fig. 3-symmetric shape), *dishwash* (labelled 1 in Fig. 3-moderately complex shape) and *lego toy* (labelled 7 in Fig. 3-highly complex shape). On an average, each trial took 3 min and 10 s. The values of a and δ_c remain same as the previous experiments. Table 2 summarises the obtained results. From the results, an important conclusion can be drawn, which is with an increase in the number of grasp hypotheses the number of switches increases, i.e., the system is more flexible in tracking (following) a moving object. Also, it can be seen that the average task space error reduces with increase in the grasp hypotheses, i.e., the system tries to track the object as closely and smoothly as possible by continuous replanning. Although the system performance is convincing, we believe the requirement of automatic best hypotheses selection from the available ones, which we plan to integrate in future.

4.5 Analysing the effect of switching threshold δ_c

In this section, various experiments are performed to analyse the effect of switching threshold on the task performance. Similar to the previous, these tests are performed in a semi-controlled manner and same test objects are used. The number of grasp hypotheses are fixed to be first 20 and a value of 0.4 is used. For analysis, we conduct the experiments using four different thresholds: 0.2, 0.4, 0.6 and 0.8. The obtained results are summarised in Table 3. From the results, it can be clearly seen that the system is more dynamic with a smaller threshold. However, an interesting fact is with the task space error, which does not follow any trend. With this results, two conclusions can be derived. On one hand, the task space error is more dependent on the number of grasp hypotheses provided, which is also true from the previous results. On the other hand, due to the availability of ample amount of grasp hypotheses (20 in this case), the system tries to follow the object locally, keeping the error low.

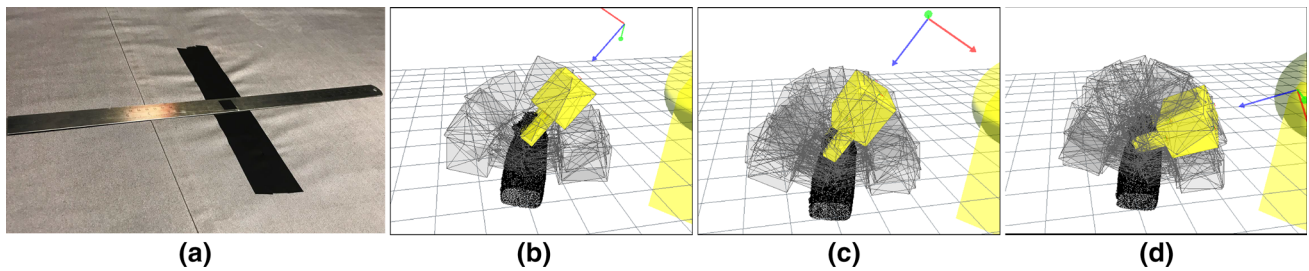


Fig. 9 **a** Tracks used by the human operator to move the object. The objects are rotated along the black line and are translated along the metallic scale (30 cm both forward and backward). **b–d** Illustration of the number of selected hypotheses on *dishwash* object. **b** First 10, **c** first 20 and **d** first 30

Table 2 Analysis of the selected grasp hypotheses

Object	Hypotheses	Switches	Avg. task space error
Big can	10	18	0.839
	20	26	0.77
	30	29	0.513
Dishwash	10	27	0.631
	20	43	0.392
	30	45	0.344
Lego toy	10	23	0.98
	20	48	0.51
	30	45	0.383

Table 3 Analysis of the switching thresholds

Object	Threshold	Switches	Avg. task space error
Big can	0.2	27	0.8
	0.4	23	0.59
	0.6	13	1.43
Dishwash	0.8	8	0.91
	0.2	39	0.35
	0.4	31	0.43
Lego toy	0.6	17	0.61
	0.8	14	0.58
	0.2	47	0.31
	0.4	39	0.4
	0.6	22	0.29
	0.8	18	0.76

5 Conclusions

This paper presented a novel approach for adaptive grasping and trajectory planning, which enables a robot arm with hand to dynamically track and grasp objects moving in arbitrary 6DoF trajectories. For a moving object whose pose is continuously updated by a particle filter-based 3D visual object tracker, the proposed method evaluates online a set of pre-generated grasp hypotheses (provided by a learned generative grasp planner) enabling the robot to maintain a suitable pre-grasp pose with minimum task space error while following the object (throughout its motion). The task space error is incorporated into the robot's motion planning through an iterative numerical optimization-based IK solver. The motion planning associated with the proposed method is comprised of dynamic switching between local and global planners, which enables the robot to track the object as closely and smoothly as possible by continuous re-planning.

The proposed method has been demonstrated in an application, in which a human move an object in arbitrary ways before handing it over to the robot, as might be necessary for collaborative human–robot co-working. The first set of experiments are conducted with a number of different objects of diverse shapes, moved along arbitrary 6DoF trajectories to analyse the performance in terms of repeatability. Obtained results suggest that the method is highly repeatable in track-

ing and grasping the objects' given that the objects pose can be visually tracked. They also demonstrate how the grasps are adapted by dynamically switching between local and global trajectory planners for objects with variety of shapes. The second set of experiments are conducted at a semi-controlled environment to analyse the effect of various parameters on the proposed approach. The results obtained from these tests suggest that the system demonstrates smooth performance with an increase in the initially selected grasp hypotheses or by using small switching threshold. Altogether, these results clearly indicate the robustness of our proposed online grasp planning method.

Our ongoing work is directed towards increasing the speed of the visual tracking by using a GPU implementation. Additionally, there are several possible directions for future research, which include integrating a gaze controller (Marturi et al. 2015) with tracking for hand-eye coordination, improving the switching approach between local and global trajectory planners, and improving the detection and selection of candidate grasp poses. We will also consider alternative applications, such as grasping stationary objects from robot arms mounted on moving vehicles.

Open Access This article is distributed under the terms of the Creative Commons Attribution 4.0 International License (<http://creativecommons.org/licenses/by/4.0/>), which permits unrestricted use, distribution, and reproduction in any medium, provided you give appropriate credit to the original author(s) and the source, provide a link to the Creative Commons license, and indicate if changes were made.

References

- Aldoma, A., Marton, Z. C., Tombari, F., Wohlkinger, W., Potthast, C., Zeisl, B., et al. (2012). Point cloud library. *IEEE Robotics & Automation Magazine*, 19(3), 80–91.
- Allen, P. K., Timcenko, A., Yoshimi, B., & Michelman, P. (1993). Automated tracking and grasping of a moving object with a robotic hand-eye system. *IEEE Transactions on Robotics and Automation*, 9(2), 152–165. <https://doi.org/10.1109/70.238279>.
- Bekiroglu, Y., Damianou, A., Detry, R., Stork, J. A., Kragic, D., & Ek, C. H. (2016). Probabilistic consolidation of grasp experience. In *Proceedings of the IEEE International Conference on Robotics and Automation*, pp. 193–200. <https://doi.org/10.1109/ICRA.2016.7487133>.
- Choi, C., & Christensen, H. I. (2013). Rgb-d object tracking: A particle filter approach on gpu. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 1084–1091.
- Fontanals, J., Dang-Vu, B. A., Porges, O., Rosell, J., & Roa, M. A. (2014a). Integrated grasp and motion planning using independent contact regions. In *Proceedings of the International Conference on Humanoid Robots*, pp. 887–893. <https://doi.org/10.1109/HUMANOIDS.2014.7041469>.
- Fontanals, J., Dang-Vu, B. A., Porges, O., Rosell, J., & Roa, M. A. (2014b). Integrated grasp and motion planning using independent contact regions. In *Proceedings of the IEEE-RAS International Conference on Humanoid Robots*, pp. 887–893. <https://doi.org/10.1109/HUMANOIDS.2014.7041469>.
- Fukui, S., Hayakawa, S., Iwahori, Y., Nakamura, T., & Bhuyan, M. (2016). Particle filter based tracking with image-based localization. *Procedia Computer Science*, 96, 977–986.
- Horowitz, M. B., & Burdick, J. W. (2012). Combined grasp and manipulation planning as a trajectory optimization problem. In *Proceedings of the IEEE International Conference on Robotics and Automation*, pp. 584–591. <https://doi.org/10.1109/ICRA.2012.6225104>.
- Houshang, N. (1990). Control of a robotic manipulator to grasp a moving target using vision. In *Proceedings of the IEEE International Conference on Robotics and Automation*, pp. 604–609 vol. 1. <https://doi.org/10.1109/ROBOT.1990.126048>.
- Hsiao, K., Chitta, S., Ciocarlie, M., & Jones, E. G. (2010). Contact-reactive grasping of objects with partial shape information. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 1228–1235. <https://doi.org/10.1109/IROS.2010.5649494>.
- Kim, S., Shukla, A., & Billard, A. (2014). Catching objects in flight. *IEEE Transactions on Robotics*, 30(5), 1049–1065. <https://doi.org/10.1109/TRO.2014.2316022>.
- Kitaev, N., Mordatch, I., Patil, S., & Abbeel, P. (2015). Physics-based trajectory optimization for grasping in cluttered environments. In *Proceedings of the IEEE International Conference on Robotics and Automation*, pp. 3102–3109. <https://doi.org/10.1109/ICRA.2015.7139625>.
- Kopicki, M. (2010). Prediction learning in robotic manipulation. Ph.D. thesis, University of Birmingham.
- Kopicki, M., Detry, R., Schmidt, F., Borst, C., Stolkin, R., & Wyatt, J. L. (2014). Learning dextrous grasps that generalise to novel objects by combining hand and contact models. In *Proceedings of the IEEE International Conference on Robotics and Automation*, IEEE, pp. 5358–5365.
- Kopicki, M., Detry, R., Adjigble, M., Stolkin, R., Leonardis, A., & Wyatt, J. L. (2016). One-shot learning and generation of dextrous grasps for novel objects. *International Journal of Robotics Research*, 35(8), 959–976.
- Lenz, I., Lee, H., & Saxena, A. (2015). Deep learning for detecting robotic grasps. *International Journal of Robotics Research*, 34(4–5), 705–724.
- Levine, S., Pastor, P., Krizhevsky, A., & Quillen, D. (2016). Learning hand-eye coordination for robotic grasping with large-scale data collection. In *International Symposium on Experimental Robotics*, Springer, pp. 173–184.
- Li, M., Bekiroglu, Y., Kragic, D., & Billard, A. (2014). Learning of grasp adaptation through experience and tactile sensing. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 3339–3346.
- Ma, M., Marturi, N., Li, Y., Stolkin, R., & Leonardis, A. (2016). A local-global coupled-layer puppet model for robust online human pose tracking. *Computer Vision and Image Understanding*, 153, 163–178.
- Magnusson, M. (2009). The three-dimensional normal-distributions transform: An efficient representation for registration, surface analysis, and loop detection. Ph.D. thesis, Örebro universitet.
- Marturi, N., Ortenzi, V., Xiao, J., Adjigble, M., Stolkin, R., & Leonardis, A. (2015). A real-time tracking and optimised gaze control for a redundant humanoid robot head. In *Proceedings of the IEEE-RAS International Conference on Humanoid Robots*, pp. 467–474.
- Menon, A., Cohen, B., & Likhachev, M. (2014). Motion planning for smooth pickup of moving objects. In *Proceedings of the IEEE International Conference on Robotics and Automation*, pp. 453–460. <https://doi.org/10.1109/ICRA.2014.6906895>.
- Ortenzi, V., Marturi, N., Stolkin, R., Kuo, J. A., & Mistry, M. (2016). A vision-guided approach to estimate the configuration of an under-sensored manipulator. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 3567–3574. <https://doi.org/10.1109/IROS.2016.7759525>.
- Roa, M. A., & Suárez, R. (2015). Grasp quality measures: Review and performance. *Autonomous Robots*, 38(1), 65–88.
- Romano, J. M., Hsiao, K., Niemeyer, G., Chitta, S., & Kuchenbecker, K. J. (2011). Human-inspired robotic grasp control with tactile sensing. *IEEE Transactions on Robotics*, 27(6), 1067–1079. <https://doi.org/10.1109/TRO.2011.2162271>.
- Rusinkiewicz, S., & Levoy, M. (2001). Efficient variants of the icp algorithm. In *International Conference on 3-D Digital Imaging and Modeling*, IEEE, pp. 145–152.
- Sausser, E. L., Argall, B., Metta, G., & Billard, A. (2012). Iterative learning of grasp adaptation through human corrections. *Robotics and Autonomous Systems*, 60(1), 55–71.
- Smith, C. E., & Papanikolopoulos, N. P. (1995). Grasping of static and moving objects using a vision-based control approach. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, vol 1, pp. 329–334 vol. 1. <https://doi.org/10.1109/IROS.1995.525816>.
- Soto, A. (2005). Self adaptive particle filter. In *Proceedings of the 19th International Joint Conference on Artificial Intelligence*, Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, IJCAI'05. pp. 1398–1403. <http://dl.acm.org/citation.cfm?id=1642293.1642515>.
- Tamadazte, B., Marchand, E., Dembélé, S., & Fort-Piat, N. L. (2010). Cad model-based tracking and 3d visual-based control for mems microassembly. *International Journal of Robotics Research*, 29(11), 1416–1434. <https://doi.org/10.1177/0278364910376033>.
- Vahrenkamp, N., Asfour, T., & Dillmann, R. (2012). Simultaneous grasp and motion planning: Humanoid robot armar-iii. *IEEE Robotics*

& *Automation Magazine*, 19(2), 43–57. <https://doi.org/10.1109/MRA.2012.2192171>.

- Vogel, J., Hertkorn, K., Menon, R. U., & Roa, M. A. (2016). Flexible, semi-autonomous grasping for assistive robotics. In *Proceedings of the IEEE International Conference on Robotics and Automation*, IEEE, pp. 4872–4879.
- Wu, Y., Lim, J., & Yang, M. H. (2013). Online object tracking: A benchmark. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 2411–2418.
- Zhang, M., & Buehler, M. (1994). Sensor-based online trajectory generation for smoothly grasping moving objects. In *IEEE International Symposium on Intelligent Control*, pp. 141–146. <https://doi.org/10.1109/ISIC.1994.367827>.

Publisher's Note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.



Naresh Marturi is a KTP Robot Vision Scientist with KUKA Robotics UK Ltd. and the University of Birmingham, since 2015. He obtained his Ph.D. degree in automatic control from the University of Franche-Comté, Besançon, France, in 2013 and M.S. degree in robotics and intelligent systems from Örebro University, Örebro, Sweden, in 2010. After Ph.D. he spent one year at FEMTO-ST Institute, France as a post-doctoral researcher. His primary research interests are in the

fields of machine vision, industrial robotics, human–robot interaction, vision-based robotic control and deep learning. At Kuka, he is mainly responsible for developing industrially robust vision-guided applications by transferring academic knowledge, tools and techniques to industry. He also possesses a solid knowledge and research experience in developing vision-guided techniques at Micro and Nano scales.



Marek Kopicki is a post-doctoral researcher at the University of Birmingham. He has a MSc in physics at Adam Mickiewicz University (Poznan), a MSc in Advance Computer Science and Ph.D. in robotics at University of Birmingham in 2010. He has more than twelve years' experience of developing state-of-the-art algorithms and software for robot control and vision which underpinned much of FP7 CogX, FP7 GeRT, and now is central to FP7 PaCMan and H2020 RoMaNs.

He invented an adaptive grasping algorithm, for which he holds an international patent pending. He has published a number refereed papers in robotics in conferences and top journals.



Alireza Rastegarpanah is an interdisciplinary engineer with diverse research interests, although his research interest broadly centres on rehabilitation robotics, physical human–robot interaction and cooperation, telerobotics and robotic manipulation. He has received MEng in Biomedical Engineering from the University of Birmingham (UoB) in 2011. Following his MEng, he started his PhD with the focus on developing a new methodology for functional lower limb robot-assisted rehabilitation based on a human-machine closed-loop system. He was graduated with a PhD in robotics from UoB in July 2016. He started his first postdoc at the Extreme robotic laboratory, the University of Birmingham, in January 2016. He worked on a project called “Machine-learning of vision-guided bi-manual grasps, for adaptable autonomous manipulation in manufacturing environments”, sponsored by Cranfield-Loughborough EPSRC.



Vijaykumar Rajasekaran is a Post-Doctoral Research Fellow in the Extreme Robotics Lab at the University of Birmingham. He completed his Masters in Robotics and control from the Ecole Centrale de Nantes, France and Warsaw University of Technology, Poland in 2010. He obtained his PhD in 2015 from the Universitat Politècnica de Catalunya, Spain. His research interests are in the scope of Human–Robot interaction, Dynamic Control, Exoskeletons and Walking Robots.



Maxime Adjigble is a Mechatronics Engineer currently working in the Extreme Robotics Laboratory at the University of Birmingham, UK. He has been involved with robotic applications on a professional level for more than 7 years and was also one of the principal Software Engineer on various projects involving a combination of Perception, Control and Machine Learning. The most popular ones being the “Boris Humanoid Robot” featured on BBC and the French “Hercule Exoskeleton” also featured on several TV shows and websites.



Rostam Stolkin is a Royal Society Industry Fellow for Nuclear Robotics and is the Director of A.R.M Robotics Ltd Birmingham UK. He is also Chair of Robotics at University of Birmingham, where he is director of the Extreme Robotics Lab, dedicated to robotics in extreme environments, with a particular focus on nuclear decommissioning. He has been with the University of Birmingham since 2008. His training included undergraduate and master's degrees in Engineering from

Oxford University, and a PhD in Robot Vision undertaken between University College London and UK imaging industry. He also worked as an Assistant Professor (Research) at Stevens Institute of Technology, USA, 2004–2008. Stolkin is highly cross-disciplinary, with patents, publications and projects spanning computer vision, sensor systems, robotic manipulation, robot vehicles, and human–robot interfaces. He is an international leader in the development of robots for nuclear applications, currently leading projects spanning six countries on three continents. He is an active member of the IEEE Technical Committee on Robotics and Automation in Nuclear Facilities.



Aleš Leonardis is a Professor at the School of Computer Science, University of Birmingham and co-Director of the Centre for Computational Neuroscience and Cognitive Robotics. He is also a Professor at the FCIS, University of Ljubljana and adjunct professor at the FCS, TU-Graz. His research interests include robust and adaptive methods for computer/cognitive vision, object and scene recognition and categorization, statistical visual learning, object tracking, and biologically

motivated vision. He has extensive experience in participating and managing multi-site international projects including eight European Commission (FP5, FP6, FP7 and H2020) projects, one DARPA project (Neovision2) and one ONR MURI project. He is also actively involved in applied robotics and computer vision projects. He has been an Associate Editor of the IEEE PAMI and IEEE Robotics and Automation Letters, an editorial board member of Pattern Recognition and Image and Vision Computing, and an editor of the Springer book series Computational Imaging and Vision.



Yasemin Bekiroglu completed her Ph.D. at the Royal Institute of Technology (KTH), in Stockholm, Sweden, in 2012. Then, she worked as a postdoctoral researcher at KTH, and at University of Birmingham, UK, and as a research scientist at ABB, Corporate Research, Sweden, coordinating the EU project SARA-Fun. She is currently a roboticist at Vicarious AI, CA, leading a team of researchers. Her research interests include robotics, computer vision and machine learning.

More specifically, she works on learning-based approaches for robot grasping and manipulation using multisensory data.