Learning better generative models for dexterous, single-view grasping of novel objects

Marek S. Kopicki1, Dominik Belter2 and Jeremy L. Wyatt1

Abstract
This paper concerns the problem of how to learn to grasp dexterously, so as to be able to then grasp novel objects seen only from a single viewpoint. Recently, progress has been made in data-efficient learning of generative grasp models which transfer well to novel objects. These generative grasp models are learned from demonstration (LfD). One weakness is that, as this paper shall show, grasp transfer under challenging single view conditions is unreliable. Second, the number of generative model elements rises linearly in the number of training examples. This, in turn, limits the potential of these generative models for generalisation and continual improvement. In this paper, it is shown how to address these problems. Several technical contributions are made: (i) a view-based model of a grasp; (ii) a method for combining and compressing multiple grasp models; (iii) a new way of evaluating contacts that is used both to generate and to score grasps. These, together, improve both grasp performance and reduce the number of models learned for grasp transfer. These advances, in turn, also allow the introduction of autonomous training, in which the robot learns from self-generated grasps. Evaluation on a challenging test set shows that, with innovations (i)-(iii) deployed, grasp transfer success rises from 55.1% to 81.6%. By adding autonomous training this rises to 87.8%. These differences are statistically significant. In total, across all experiments, 539 test grasps were executed on real objects.

Keywords
learning, generative models, dexterous grasping

1 Introduction
Dexterous grasping of novel objects is an active research area. The scenario considered here is that in which a novel object must be dexterously grasped, after being seen from just a single viewpoint. We refer to this scenario as dexterous, single-view grasping of novel objects. This is essentially the grasping problem solved by humans and establishes a high bar for robots.

The combination of scenario features means that it is hard to apply planning methods based on analytic mechanics. This is because they require knowledge of frictional coefficients, object mass, and complete object shape to evaluate a proposed grasp. None of these are either known a priori nor easily recovered from a single view.

Alternatively, there are learning methods. Broadly speaking, these divide into those that learn generative models and those that learn evaluative models. Generative models take sensor data as input and produce one or more candidate grasps. Evaluative models take sensor data and a grasp candidate as input and produce an estimate of the quality of that grasp on the target object. In this paper, we consider how to improve generative model learning.

This paper builds on recent work on learning grasps from demonstration (LfD). There are now LfD methods for learning probabilistic generative models of grasps. Specifically, the baseline method that this paper builds on learns such a model from a small number of examples, and can then use it to generate dexterous grasps for novel objects. One drawback is that, while this baseline method can work for many single-view grasps, it is not reliable on challenging cases. Such cases include objects placed so that the surface recovery is limited, and where grasping requires contact to be made on a hidden back surface, as shown in Figure 1. In addition, there are limits to its ability to take advantage of an increasing quantity of training data, since the approach is a purely memory-based learner and has no ability to combine models from different training examples.

This paper shows how to make single-view grasping more reliable. This involves several innovations. First, (Innovation 1) we show how to learn multiple, view-specific grasp models from a single example grasp. These view-specific models enable grasps to be generated that compensate for missing back surfaces of deep objects, a typical occurrence in single-view grasping (Figure 1). Second, (Innovation 2) we move beyond memory-based models by showing how to combine information from multiple training grasps into a smaller number of generative models. This compression leads to an improvement in model generalisation and inferential efficiency on test objects. Third, (Innovation 3) we present a novel way to calculate the likelihood of finger-object contacts in a candidate grasp. This new

1 School of Computer Science, University of Birmingham, Edgbaston, Birmingham, B15 2TT.
2 Institute of Control and Information Engineering, Poznan University of Technology, Poland.

Corresponding author:
Jeremy L. Wyatt, School of Computer Science, University of Birmingham, Edgbaston, Birmingham, B15 2TT.
Email: jeremy.l.wyatt@gmail.com
likelihood function is used both to generate and evaluate candidate grasps. Together, these three innovations improve the performance of dexterous, single-view grasping from 55.1% to 81.6% on a test set of novel objects placed in difficult poses. Finally, we show how learning can be scaled by using self-generated grasps on test objects as further training data. This raises the grasp success rate to 87.8%.

Given these innovations, the paper tests the following hypotheses.

**H1** Even without an enlarged set of training grasps, the combined innovations 1-3 improve the grasp success rate.

**H2** View-based grasp modelling enables better generation of grasps for thick objects.

**H3** The grasp success rate improves progressively as innovations are added.

**H4** With all innovations the grasp success rate improves as the training data increases.

**H5** With all innovations, learning is better able than the baseline algorithm to exploit an increased amount of training data.

**H6** With all innovations the algorithm dominates the baseline algorithm without any innovations.

The paper is structured as follows. First, related work is described. Second, the approach is described in detail. This begins with a description of the basic framework for probabilistic generative grasp learning. It then proceeds to a description of the new learning algorithm, notably the view-based model representation and the technique for generative model compression. Following this, the new contact likelihood function and its uses are described. Finally, the results of an empirical study are presented.

## 2 Related work

### 2.1 Overview

We identify four broad approaches to grasp planning. First, there are those that use analytic mechanics to evaluate grasp quality. Second, there are methods that engineer a mapping from sensor data to a candidate grasp or grasps. Third, there are methods that learn this mapping. Finally, there are methods that instead learn a mapping from sensor data and a candidate grasp to a prediction of grasp quality or grasp success probability. To place our work in context, we review the properties of each of these, plus relevant methods in grasp execution. Recent surveys of grasping include those by Bohg et al. (2014) on data-driven grasping and Sahbani et al. (2012) on analytic methods.

### 2.2 Analytic approaches

Analytic approaches to grasping use the laws of mechanics to predict the outcome of a grasp (Bicchi and Kumar 2000; Liu 2000; Pollard 2004; Miller and Allen 2004). These analyses require a model of the target object’s shape, mass, mass distribution, and surface friction. They also need a model of the gripper kinematics and the exertable contact forces in different configurations. Obtaining these permits computation of the resistable external wrenches for a grasp. Based on this, a number of so-called grasp quality metrics can be defined (Ferrari and Canny 1992; Roa and Suarez 2015; Shimoga 1996).

Analytic approaches have been successfully applied to find grasps for multi-fingered hands (Boutsalis et al. 2014; Gori et al. 2014; Hang et al. 2014; Rosales et al. 2012; Saut and Sidobre 2012; Ciocarlie and Allen 2009). All of these essentially pose grasp generation as optimisation against a grasp-quality metric. The appeal is that analytic methods are interpretable and scrupulous, but there are drawbacks. Estimation of the object’s properties is challenging. One solution is to build a library of objects that might be grasped, matching partially reconstructed novel objects to similar ones in the library (Goldfeder and Allen 2011). Alternatively, parameters of the target object such as mass (Zheng and Qian 2005; Shapiro et al. 2004) or friction (Rosales et al. 2014) must be recovered on the fly using vision and touch. One approach is for a complete object model to be recovered from a partial view (e.g. a single depth image) by assuming shape symmetries (Bohg et al. 2011), by using a 3D CNN (Varley et al. 2017), or by a hierarchical shape approximation (Huebner et al. 2008). The complete object can then be fed as input to an engine such as GraspIt. Search for a grasp can be improved by employing low-dimensional hand pose representations (Ciocarlie and Allen 2009).
Allen 2009). There are, however, several assumptions underpinning many analytic methods, such as hard contacts with a fixed contact area (Bicchi and Kumar 2000) and static friction (Shimoga 1996). There are methods that extend modelling to, for example, soft contacts (Ciocarlie et al. 2005). A more fundamental problem is that even a small error in the estimated shape, mass or friction can render an apparently good grasp unstable (Zheng and Qian 2005). This can be mitigated to an extent by using independent contact regions (ICRs) (Ponce and Favrejon 1995; Rusu et al. 2009b). Despite this, there is some evidence grasp quality metrics based on mechanics are not strongly indicative of real-world grasp success (Bekiroglu et al. 2011a; Kim et al. 2013; Goins et al. 2014). In addition, such metrics can be costly to compute many times during a grasp search procedure. Analytic models are quite general, however, and so can be used for tasks such as planning grasps in clutter (Dogar et al. 2012).

2.3 Engineered mappings from sensor to grasp

Difficulties with analytic methods led to investigation of vision-based grasp planning. These methods use RGB or depth images, or representations like point clouds or meshes. Grasp generation becomes a search for object shapes that fit the robot’s gripper (Popovic et al. 2010; Trobina and Leonardis 1995; Klingbeil et al. 2011; Richtsfeld and Zillich 2008; Kootstra et al. 2012; ten Pas and Platt 2014). This includes finding parallel object edges in intensity images (Popovic et al. 2010) or planar sections in range images (Klingbeil et al. 2011). Potential grasps can be also found by matching curved patches in a point cloud that can support contacts (Kanoulas et al. 2017). These candidate grasps are then refined using their shape and pose properties. The rule based approach works well for pinch gripping, but does not scale well to dexterous grasping because of the increased size of the search space. Visual servoing can be used to improve grasp reliability (Kragic and Christensen 2003). Partially known shapes can be grasped using heuristics both for grasp generation and for the reactive finger closing strategy used to execute the grasp under tactile sensing (Hsiao et al. 2010). Such reactive strategies can also be derived for pose and or shape uncertainty automatically in a decision theoretic framework (Hsiao et al. 2011; Arruda et al. 2016). These reactive strategies can include push manipulation to make a good grasp more likely (Dogar and Srinivasa 2010). Finally, the grasp itself may also be formulated by taking uncertainty into account as a constraint in the planning process (Li et al. 2016).

2.4 Learning a mapping from sensor to grasp

The next wave of grasp generation methods learned this mapping from data to grasp instead. Most of these methods learn relations between features extracted from the object representation, such as SIFT or other features (Saxena et al. 2008; Fischinger and Vincze 2012), shape primitives (Platt et al. 2006), box-decompositions (Huebner and Kragic 2008) or object parts (Kroemer et al. 2012; Detry et al. 2013). The grasp itself can be parametrised as a grasp position (Saxena et al. 2008), gripper pose (Herzog et al. 2014) or a set of contact points (Ben Amor et al. 2012; Bohg and Kragic 2010). Some methods learn grasps from demonstration (Ekvall and Kragic 2004; Hillenbrand and Roa 2012; Kopicki et al. 2014, 2015; Hsiao and Lozano-Perez 2006), and in the case of Kopicki et al. (2015) create a generative model able to generate many grasp candidates for the target object. Others learn a distribution of possible grasps indexed by features from semi-autonomous grasping experiments (Detry et al. 2011). Recently, deep learning has been applied to learn such mappings (Redmon and Angelova 2015; Kumra and Kanan 2017).

2.5 Learning a grasp evaluation function

Learning approaches have also been applied to the problem of acquiring a grasp evaluation function from data. For example grasp stability for an executed grasp can be learned (Bekiroglu et al. 2011b). An evaluation of a proposed grasp can also be learned. This problem has been tackled recently using data intensive learning methods. Most of these methods predict the grasp quality for a parallel-jaw gripper (Pinto and Gupta 2016; Lens et al. 2015; Johns et al. 2016; Mahler et al. 2016, 2017; Redmon and Angelova 2015; Seita et al. 2016; Wang et al. 2016; Gualtieri et al. 2016; Levine et al. 2017). Pinto and Gupta (2016), for example, learn a function that predicts the probability of grasp success for an image patch and gripper angle. To reduce the quantity of real grasps a rigid-body simulation (Johns et al. 2016; Bousmalis et al. 2018) or synthetic dataset (Mahler et al. 2016, 2017) may be used. A synthetic data set requires that analytic grasp metrics be computed offline. Mahler et al. (2016) use a synthetic data set to predict the probability of force closure under uncertainty in object pose, gripper pose, and friction coefficient. Seita et al. (2016) performed supervised learning of grasp quality using deep learning and random forests. Gualtieri et al. (2016) predict whether a grasp will be good or bad using a CNN trained on depth images, using instance or category knowledge of the object to help. Hyttinen et al. (2015) used tactile signatures fed to a trained classifier to predict object grasp stability.

2.6 Deep learning of dexterous grasping

A small number of papers have explored deep learning as a method for dexterous grasping. (Lu et al. 2017; Varley et al. 2015; Veres et al. 2017; Zhou and Hauser 2017; Kappler et al. 2015). All of these methods use simulation to generate the set of training examples for learning. Kappler et al. (2015) showed the ability of a CNN to predict grasp quality for multi-fingered grasps, but uses complete point clouds as object models and only varies the wrist pose for the pre-grasp position, leaving the finger configurations the same. Varley et al. (2015) and later Zhou and Hauser (2017) went beyond this, each being able to vary the hand pre-shape, and predicting from a single image of the scene. Each of these posed search for the grasp as a pure optimisation problem (using simulated annealing or quasi-Newton methods) on the output of the CNN. They all, also, take the approach of learning an evaluative model, and generate candidates for evaluation uninfluenced by prior knowledge. Veres et al. (2017), in contrast, learn a deep generative model. Finally, Lu et al. (2017) learn an evaluative model, and then, given
Figure 3. The structure of grasp training and testing in four stages. Stage 1: an example grasp is shown kinesthetically. Multiple contact models (one for each hand-link) and a hand configuration model are learned. Stage 2: when a new object is presented a partial point cloud model is constructed and combined with each contact model to form a set of query densities. Stage 3: many grasps are generated, each by selecting a hand-link, sampling a link pose on the new object from the query density and sampling a hand configuration. Stage 4: grasp optimisation maximises grasp likelihood. This stage is repeated until convergence.

2.7 Relation of this work to the literature
The main similarities and differences between this work and previous methods reported in the literature may be summarised as follows. First, our method falls within the category of learning a mapping from sensory input to grasp. Thus, it differs from methods that learn an evaluation function for a proposed grasp. Second, like (Kopicki et al. 2015) it learns a generative model, and is thus able to generate many candidate grasps for a new object, rather than just one. One particular property of the method built upon (Kopicki et al. 2015) is that it can learn from a very small number of demonstrations. There are two main drawbacks of that previous work. First, it is not sufficiently robust when grasping a novel object from a single view. Second, it is purely memory based, so it cannot merge learned models, and so doesn’t extract the best models from an increasing amount data. In this paper, these drawbacks are addressed.

3 Generative Grasp Modelling Basics
The general approach is one of Learning from Demonstration (LfD). We first sketch the general structure of learning a generative grasp model from demonstration (Kopicki et al. 2015). This structure is the same across both the algorithm described in Kopicki et al. (2015) and the algorithm described here. For simplicity, we refer to the algorithm from Kopicki et al. (2015) as the vanilla algorithm. Throughout we assume that the robot’s hand comprises $N_L$ rigid links: a palm, and several phalanges per finger. We denote the set of hand-links $L = \{L_i\}$. *

First, an example grasp is presented, and then a model of that grasp is learned. This grasp model comprises two different types of sub-model (Figure 3: Stage 1). Both of these types of model are probability densities. The first type is a contact-model of the spatial relation between a hand-link and the local object shape near its point of contact. A contact-model is learned for every hand-link in contact with the object. The second type is a hand-configuration model, that captures the overall hand-shape.

Given this learned model of a grasp, new grasps can be generated and evaluated for novel objects. This grasp transfer occurs in three stages (Figure 3: Stages 2-4). First, each contact model is combined with the available point cloud model of the new object, to construct a third type of density called a query density (Stage 2). We build one query density for every contact model created during learning. Each query density is a probability density over where the hand-link will be on the new object.

Next, we generate initial candidate grasps on the new object (Stage 3) from the generative model of a grasp. Each candidate grasp is produced in three steps. First a hand-link is chosen at random. Then a pose for this hand-link on the new object is sampled. Finally, a configuration of the hand is sampled. The sampling of the hand-link pose uses the query density for that hand-link. The sampling of the hand-configuration uses the hand-configuration model. By forward kinematics these three samples (hand-link, hand-link pose and hand configuration) together determine a grasp. Many such initial grasps are generated in this way.

In the final stage each initial grasp is refined by simulated annealing (Stage 4). The goal is to improve each initially generated grasp so as to maximise its likelihood according to the generative model. The optimisation criterion is a product of experts. There is one expert for each hand-link.

* For clarity we refer throughout to hand-links.
and one expert for the hand-configuration. These experts are the query-densities and the hand-configuration model.

Generative grasp learning of this type has several desirable properties. First, it is known to be somewhat robust to partial point-cloud data for both training and test objects. Second, it displays generalisation to globally different test shapes. Third, the speed of inference is quite good if the learner is only given a small number of training grasps (2000 candidate grasps are generated and refined in < 1 sec on a modern 16-core PC). Fourth, there is evidence of robustness to variation in the orientation of the novel object.

However, as mentioned previously, there are also weaknesses in this type of generative grasp model. These are: (i) a need to further improve robustness when grasping from a single-view of the test object; (ii) a need to extract the best possible generative models as the number of training grasps grows.

Having sketched the basic structure of generative grasp learning we now detail model learning and grasp inference for our new algorithm. We start by describing the basic representations that underpin the work. As we proceed we will highlight the innovations made.

4 Representations

The method requires that we define several models: an object model (partial and acquired from sensing); a model of the contact between a hand-link and the object; and a model of the whole hand configuration.

Since all of these models are probability densities, underpinning all of them is a density representation. We first describe the kernel density representation we employ. Then we describe the various local surface descriptors we may use as the basis for the contact and query density models. We follow this with a description of each model type.

4.1 Kernel Density Estimation

$SO(3)$ denotes the group of rotations in three dimensions. A feature belongs to the space $SE(3) \times \mathbb{R}^{N_r}$, where $SE(3) = \mathbb{R}^3 \times SO(3)$ is the group of 3D poses, and surface descriptors $r$ are composed of $N_r$ real numbers. We extensively use probability density functions (PDFs) defined on $SE(3) \times \mathbb{R}^{N_r}$. We represent these PDFs non-parametrically with a set of $N$ features (or particles) $x_j$

$$S = \{x_j : x_j \in \mathbb{R}^3 \times SO(3) \times \mathbb{R}^{N_r}, j \in [1,N]\}.$$

(1)

The probability density in a region 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 $K$ to each particle supporting the density, as

$$pdf(x) \approx \sum_{j=1}^{N} w_j K(x|x_j, \sigma),$$

(2)

where $\sigma$ is the kernel bandwidth and $w_j \in \mathbb{R}^+$ is a weight associated with $x_j$ such that $\sum_j w_j = 1$. We use a kernel that factorises into three functions defined by the separation of $x$ into $p \in \mathbb{R}^3$ for position, a quaternion $q \in SO(3)$ for orientation, and $r \in \mathbb{R}^{N_r}$ for the surface descriptor.

Furthermore, let us define $\mu$ and $\sigma$:

$$x = (p,q,r),$$

$$\mu = (\mu_p, \mu_q, \mu_r),$$

$$\sigma = (\sigma_p, \sigma_q, \sigma_r).$$

(3a,b,c)

We define our kernel as

$$K(x|\mu, \sigma) = N_3(p|\mu_p, \sigma_p)\Theta(q|\mu_q, \sigma_q)N_{N_r}(r|\mu_r, \sigma_r)$$

(4)

where $\mu$ is the kernel mean point, $\sigma$ is the kernel bandwidth, $N_{N_r}$ is a $N_r$-variate isotropic Gaussian kernel, and $\Theta$ corresponds to a pair of antipodal von Mises-Fisher distributions which form a Gaussian-like distribution on $SO(3)$ (Fisher 1953); Sudderth (2006). It is the natural equivalent of the Gaussian for circular variables such as orientation. The value of $\Theta$ is given by

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

(5)

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

Using this representation, conditional and marginal probabilities can easily be computed from Eq. (2). The marginal density $pdf(r)$ is computed as

$$pdf(r) = \int \sum_{j=1}^{N} w_j N_3(p|p_j, \sigma_p)\Theta(q|q_j, \sigma_q)N_{N_r}(r|r_j, \sigma_r)dpdq$$

(6a)

$$= \sum_{j=1}^{N} w_j N_{N_r}(r|r_j, \sigma_r),$$

(6b)

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

$$pdf(p, q|r) = \frac{pdf(p, q, r)}{pdf(r)}$$

(7)

4.2 Surface Descriptors

To condition the contact models it is necessary to have some descriptor of the local surface properties in the region of the contact. In principle these descriptors could capture any property, including local curvatures, surface smoothness, etcetera, that may influence the finger pose. In this paper we consider two different surface descriptors based solely on point-cloud data: local curvatures and fast point feature histograms (FPFH). We briefly describe the former. The latter is described in Rusu et al. (2009a).

The principal curvatures are the surface descriptor used in the vanilla algorithm. To create the descriptor, all the points in the point cloud are augmented with a frame of reference and a local curvature descriptor. For compactness, we also denote the pose of a feature as $v$. As a result,

$$x = (v, r), \quad v = (p, q).$$

(8)

The surface normal at $p$ is computed from the nearest neighbours of $p$ using a PCA-based method (e.g. Kanatani (2005)). The surface descriptors are the local principal curvatures, as described in Spivak (1999). Their directions
are denoted \( k_1, k_2 \in \mathbb{R}^3 \), and the curvatures along \( k_1 \) and \( k_2 \) form a 2-dimensional feature vector \( r = (r_1, r_2) \in \mathbb{R}^2 \).

The surface normal and the principal directions define the orientation \( q \) of a frame that is associated with the point \( p \).

### 4.3 Object View Model

The new method proposed in this article requires a set of view-based models of each training object. Let there be several object-grasp training examples \( g = 1 \ldots N_G \). Let each of these examples be observed from \( m \) views \( m = 1 \ldots N_V \). A model of the grasped object from this view, denoted \( V_{mg}(v, r) \), is computed from a single point cloud \( m \) captured by a depth sensor as a set of \( N_{V_m} \) features \( \{(v_{jmg}, r_{jmg})\} \). This set of features defines, in turn, a joint probability distribution, which we call the object-view model:

\[
V_{mg}(v, r) \equiv \text{pdf}_{mg}(v, r) \simeq \sum_{j=1}^{N_{V_m}} w_{jmg} \mathcal{K}(v, r|x_{jmg}, \sigma_x)
\]

where \( V \) is short for \( \text{pdf}^V \), \( x_{jmg} = (v_{jmg}, r_{jmg}) \), \( \mathcal{K} \) is defined in Eq. (4) with bandwidth \( \sigma_x = (\sigma_v, \sigma_r) \), and where all weights are equal, \( w_{jmg} = 1/N_{V_m} \). In summary, this object-view model \( V_{mg} \) represents the object surface as a pdf over surface points and descriptors.

### 5 Contact Learning

Having set up the basic representations, we now describe the learning procedure. This includes the new view-based grasp model, and the procedure to merge contact-models learned from different grasp examples. This section corresponds to the left branch of Stage 1 of Figure 3.

#### 5.1 Views

In Kopiicki et al. (2015) the representation required that all views of the training object were registered into a single point cloud (Figure 4 left). A model of the grasp was learned from this registered point cloud. Instead, in this paper, a separate grasp-model is learned for each view (Figure 4 centre and right). Each view-based grasp model contains both a model of the hand shape—thereby modelling all the hand-link positions relative to one another—and a model of each of the hand-object contacts that can be seen in that view. Thus each view-based grasp model excludes contact information for contacts it cannot see.

This means that the grasp models are organised by view. (Figure 5). The purpose is that the learned models more closely reflect the partial information available to the robot when grasping an object from a single available view. At inference time it means that the grasp optimisation procedure will not try to force all hand-links which were in contact in the training grasp into contacts with visible surfaces, instead relying on the hand shape model to implicitly guide hand-links to hidden back surfaces.

#### 5.2 Contact Receptive Field

Having defined the structure of the view-based grasp model, we now define the contact models that form part of it. This involves defining a receptive field around a contact, which determines how important different points on the object surface are in the contact model.

The contact receptive field \( F_i \) is a region of space relative to the associated hand-link \( L_i \) (see Fig. 6) which specifies the neighbourhood of that link. The contact receptive field \( F_i \) is realised as a function of surface feature pose \( v \):

\[
F_i : SE(3) \rightarrow [0, 1]
\]

the value of which determines the relevance of a particular surface feature \( x = (v, r) \) to a given hand-link \( L_i \) in terms of the likelihood of the physical contact. We use contact receptive fields which are family of parameterised functions for which the value falls off quickly with the distance to the link:

\[
F_i(v|\lambda_i, \delta_i) = \begin{cases} \exp(-\lambda_i ||p - a||^2) & \text{if } ||p - a|| < \delta_i \\ 0 & \text{otherwise} \end{cases}
\]

where \( \lambda_i > 0 \) and \( a \) is the point on the surface of \( L_i \) that is closest to \( v = (p, q) \). This means that the contact receptive field will only take account of the local shape, while
falling off smoothly. A variety of monotonic, fast declining functions could be used instead.

5.3 Contact Model Density

Now we have defined the receptive field we can define the contact model itself. We denote by \( u_{ij} = (p_{ij}, q_{ij}) \) the pose of link \( L_i \) relative to the pose \( v_j \) of the \( j \)-th object 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 \( v_j^{-1} = (-q_j, p_j, q_j^{-1}) \) (see Fig. 6).

Contact model \( M_{ijmg} \) encodes the joint probability distribution of object surface features for the \( i \)-th hand-link, \( m \)-th view and \( g \)-th object-grasp example:

\[
M_{ijmg}(U, R) = \mathbf{pdf}_{ijmg}^M(U, R)\tag{13}
\]

where \( M_{ijmg} \) is short for \( \mathbf{pdf}_{ijmg}^M \), \( U \) is the random variable modelling object surface features of grasp-view example \( g \), and \( U \) models the pose of \( L_i \) relative to the frame of reference defined for the feature. In other words, denoting realisations of \( U \) and \( R \) by \( u \) and \( r \), \( M_{ijmg}(u, r) \) gives us the probability of finding \( L_i \) at pose \( u \) relative to the frame of a nearby object surface patch exhibiting surface descriptor \( r \).

The contact model for link \( i \), view \( m \) and object \( g \) is estimated as:

\[
M_{ijmg}(u, r) \approx \frac{1}{Z} \sum_{j=1}^{N_{mg}} F_i(v_{jmg})K(u, r|u_{ijmg}, r_{jmg}, \sigma)\tag{14}
\]

where \( Z > 0 \) is a normalising constant, \( u_{ijmg} = v_{jmg}^{-1} \circ s_i \), \( \sigma = (\sigma_u, \sigma_r) \), \( N_{mg} \) is the number of features in the point cloud, and \( K \) is kernel function (4) defined at poses from Eq. (12).

We now also introduce, for the first time, the idea of a contact model norm, which estimates the extent of the likely area of a physical contact of hand-link \( i \) with surface features visible from view \( m \) of grasp-object pair \( g \):

\[
\|M_{ijmg}\| = \sum_{j=1}^{N_{mg}} F_i(v_{jmg})\tag{15}
\]

We use this norm to help estimate which links are reliably involved in a grasp.

5.4 Contact model selection

A view-based learning framework has the consequence that not all learned models are useful. In any grasp-view pair some hand-links may make poor contacts with the observed parts of the object. This can also simply be caused by the relevant contact not being visible from a particular viewpoint. In both cases we must determine which contact-models should be created and which ignored. This is the purpose of lines 1-13 of Algorithm 1.

The contact model selection procedure determines, for a given grasp example \( g \), view \( m \) and hand-link \( i \), whether the contact model \( M_{ijmg} \) should be created. It proceeds in two phases. The first phase uses the contact model norm (15) to prune out unreliable grasps (Algorithm 1, lines 1-3). The decisions are recorded in a set of binary variables termed contact hypotheses, \( b_{ijmg} \):

\[
b_{ijmg} = \begin{cases} 1 & \text{if } \frac{N_L N_v N_G}{\sum_{j=1}^{N_{mg}} \|M_{ijmg}\|} > \eta_i \\ 0 & \text{otherwise} \end{cases}\tag{16}
\]

where the contact-model is retained if \( b_{ijmg} = 1 \), \( \eta_i \in \mathbb{R}^+ \) is the threshold, \( N_L \) is the number of hand links, \( N_v \) is the number of views of grasp example \( g \), and \( N_G \) is the number of grasp examples.

Having pruned out unreliable contact models from each grasp-view pair the second phase prunes out unreliable grasp-view examples (Algorithm 1, lines 4-6). A grasp-view example is retained if the total number of non-empty contact models for a particular view \( m \) and grasp \( g \), determined by \( b_{ijmg} \), is higher than some minimum number \( \zeta \). This is encoded in a set of binary variables termed view hypotheses, \( c_{mg} \):

\[
c_{mg} = \begin{cases} 1 & \text{if } \sum_{i=1}^{N_L} b_{ijmg} > \zeta \\ 0 & \text{otherwise} \end{cases}\tag{17}
\]

After a number of example grasps we will thus obtain a set of non-empty contact models (Algorithm 1, lines 7-13). We may index these using a triplet of indices \((i, m, g)\) corresponding to the hand-link, view and grasp example. Because of contact-model pruning not all \((i, m, g)\) will have a contact-model, i.e. for some views, links or grasps the contact model will be empty. We denote the set of indices for the non-empty contact models \( M \). The size of this set is \( N_M \).

\[
(i, m, g) \in M\tag{18}
\]

The parameters \( \lambda, \eta, \zeta, \sigma_u, \sigma_r \) were chosen empirically. The time complexity for learning each contact model \( M_{ijmg}(u, r) \) is \( O(T_i N_{Vmg}) \) where \( T_i \) is the number of triangles in the mesh describing hand link \( i \), and \( N_{Vmg} \) is the number of features of view \( m \) and example grasp \( g \).
5.5 Clustering Contact Models

So far, we have defined how a contact model is learned. Using this memory based scheme, the number of contact models $N_M$ will grow linearly with the number of training examples. In a memory based learner, every contact model must be transferred to the target object. This may also limit the generalisation power of the contact models. This paper presents an alternative to memory based learning. We may exploit a growing number of training examples by merging contact models. This is the purpose of lines 14-22 of Algorithm 1. We hypothesize that this merging process will result in higher grasp success rates at transfer time. To merge models we first cluster the contact models according to similarity. Since each contact model is a kernel density estimator, the key step is to define an appropriate similarity measure between any pair of such estimators. Our principal aim is to produce a distance that is fast to compute and which is robust to the natural variations in the underlying data in the grasping domain. We define, first, an asymmetric divergence and then, on top of it, a symmetric distance. Since we are using kernel density estimators we can most simply define a distance between the sets of kernel centres.

First we define a distance between two kernels lying in $SE(3)$, $x = (p_x, q_x, r_x)$ and $y = (p_y, q_y, r_y)$ (see (3a)), as a weighted linear combination of sub-distance measures:

$$d_k(x, y) = w_{lin}d_{lin}(p_x, p_y) + w_{ang}d_{ang}(q_x, q_y)$$

where $w_{lin}, w_{ang} \in \mathbb{R}^+$ are weights, and

$$d_{lin}(p_x, p_y) = |p_x - p_y|^2,$$

$$d_{ang}(q_x, q_y) = 1 - \|q_x\| \cdot \|q_y\|$$

The $\|.\|$ operator extracts the surface normal, and $\cdot$ denotes a dot product. Using this, we define the distance of kernel $x$ from kernel density $M$ as

$$d_{dd}(x, M) = \min_{y \in M} \{d_k(x, y)\}$$

This distance considers only one kernel $y \in M$ that is the nearest to $x$. This has two major advantages. First, it allows the use of fast nearest neighbour search techniques with time complexity $O(\log(N))$ rather than $O(N)$, where $N$ is the number of kernels in $M$. Second, the distance given by (21) is independent of the remaining kernels in the density $M$. Additionally, for further efficiency, the distance measure (21) ignores kernel weights. This approach is valid since all weights are computed using (11). Thus, each weight depends only on the kernel position relative to the local frame of the relevant hand-link, which is already accounted for in the linear distance (20a).

Next, we use this distance to define the divergence of kernel density $M_j$ from kernel density $M_i$

$$d_{dd}(M_i, M_j) = \frac{1}{N_i} \sum_{x \in M_i} d_{dd}(x, M_j)$$

where $N_i$ is the number of kernels of $M_i$. Note that this divergence (22) is asymmetric. For example $d_{dd}(M_i, M_j)$ may be large, while $d_{dd}(M_j, M_i) = 0$, if $M_i$ is constructed from $M_j$ by removing some large “surface patch”.

Algorithm 1 Contact model selection and clustering

1: for all grasps $g$, views $m$, links $i$ do
2: compute $b_{img}$ using Eq. (16)
3: end for
4: for all grasps $g$, views $m$ do
5: compute $c_{img}$ using Eq. (17)
6: end for
7: Set $M = \{\}$
8: for all grasps $g$, views $m$, links $i$ do
9: if $b_{img} > 0$ and $c_{img} > 0$ then
10: add index triplet $(i, m, g) \rightarrow M$
11: $N_M = |M|$
12: end if
13: end for
14: Set $D$ to be an $N_M \times N_M$ matrix
15: for all pairs of triplets $(k, j) \in M$ such that $k \neq j$ do
16: compute distance $D_{kj} = d_{dd}(M_k, M_j)$ using Eq. (22)
17: end for
18: $C \leftarrow$ affinity-propagation($M, D$)
19: for all clusters $C_l$ do
20: compute $M_l^C$ using Eq. (24)
21: end for
22: return $\{M_l^C\}_{l=1}^{N_C}$

To cluster the contact models, however, we require a symmetric distance, which we define as:

$$d(M_i, M_j) = \max\{d_{dd}(M_i, M_j), d_{dd}(M_j, M_i)\}$$

It is worth noting some other benefits of this distance definition within our domain. First, (21) ignores surface descriptors. This is both because they can be high dimensional and because the shape properties they encode are already encoded in the remaining kernels of the contact models, and so are captured in (22). Thus, measuring the distance with respect to the surface descriptor adds no benefit. For the same reason, we do not compute the distances between pairs of local frames. Instead, we simply compare pairs of surface normals.

This distance is calculated for every pair of contact models (Algorithm 1, lines 14-17). The next stage is to cluster contact models using the metric defined by (23). Many clustering procedures could be used. We used affinity propagation (Frey and Dueck 2007), which requires computation of all pair-wise distances (Algorithm 1, line 18). Affinity propagation finds clusters together with cluster exemplars—the most representative cluster members. Clustering creates a partition $C = [C_1 \ldots C_l \ldots C_{N_C}]$ of the set of contact models $M$. Where $N_C$ is the number of clusters and $C_l$ denotes the $l$-th cluster, which is a set with $N_{C_l}$ members. There is a one-to-one map from each contact model index $(i, m, g)$ onto its corresponding cluster and index within that cluster, $(l, k)$. Thus, we can write out cluster $C_l$ as $C_l = \{M_{i_1}^C \ldots M_{i_k}^C \ldots M_{i_{N_{C_l}}}^C\}$. Note that,

1This is useful in our domain since $M$ is constructed from a depth image taken from an RGBD camera. The density of points underpinning $M$ varies with the object-camera distance. Using a distance between $x$ and $M$ that depends only on the closest kernel in $M$ renders the distance much less sensitive to variations in the density of $M$. This improves generalisation.
Figure 7. The prototypes produced for four of the clusters after affinity propagation. Each picture shows the density over rigid body transformations for the surface relative to the frame (shown) attached to the finger link. Only the positions of the kernel centres (black dots) are shown. Surface descriptors and orientations have been marginalised out.

when referring to a contact model as a member of a cluster, we use the superscript \( C \) for clarity as to the meaning of the index in the subscript.

In order to boost the generalisation capability of the clustered contact models (in particular for small clusters), we create \( \text{cluster prototypes} \), denoted \( M^C_l \), to replace cluster exemplars. We first define a multinomial distribution, for each cluster \( l \), over the members \( k \) of that cluster, \( P(k|C_l) \). This lets us define, in turn, a cluster prototype contact model \( M^C_l \) as a mixture model:

\[
M^C_l(u, r) = \sum_{M^C_{lk}\in C_l} P(k|C_l)M^C_{lk}(u, r) \quad (24)
\]

We evaluate this mixture with a simple Monte Carlo sampler. The probabilities \( P(k|C_l) \) are obtained using the density distance (23) between a cluster member with index \( k \) and the cluster exemplar:

\[
w^C_{lk} = \exp \left( -\xi d(M^C_{lk}, M^C_l) \right) \quad (25)
\]

where \( \xi > 0 \) controls the spread of the probability density around the cluster exemplar. The \( P(k|C_l) \) are the normalised versions of the weights \( w^C_{lk} \). A cluster prototype is calculated for each cluster (Algorithm 1, lines 19-21).

We can visualise the resulting prototypes (Figure 7) and the cluster members (Figure 8). It can be seen that the clusters are coherent and well separated. This corresponds to the fact that, in terms of the link-to-surface relations, there are a distinct number of contact types.

The parameters \( w_{\text{lin}}, w_{\text{ang}} \) and \( \xi \) were chosen empirically. The time complexity for computing all contact model pairwise distances is \( O(N_M(N_M - 1)N \log(N)) \) where \( N_M \) is the number of contact models after the selection procedure has been applied, and where \( N \) is the average number of kernels in a contact model. The time complexity of the clustering algorithm is \( O((N_M)^2N_{\text{steps}}) \) where \( N_{\text{steps}} \) is the number of iterations of the affinity propagation algorithm Frey and Dueck (2007).

5.6 Hand Configuration Model

The hand configuration model \( C^g \), for a grasp \( g \), was originally introduced in Kopicki et al. (2015) and remains the same here. It is thus described for completeness. It 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 \( g \). The purpose of this model is to allow us to restrict the grasp search space, during grasp transfer, to be close to hand configurations in the training grasp. Learning this model is the right hand branch of Stage 1 in Figure 3.

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. We denote by \( h^t_c \) the joint angles at some small distance before the hand reached the training object, and by \( h^g_c \) the hand joint angles at the time when the hand made contact with the training object. We consider a set of configurations interpolated between \( h^t_c \) and \( h^g_c \), and
extrapolated beyond $h^g$, as
\[ h_c(\gamma) = (1 - \gamma)h^g + \gamma h_c^T \]  
(26)
where $\gamma \in \Gamma$ and $\Gamma$ is a regularly spaced set of values from the real-line. For all $\gamma < 0$, configurations $h_c(\gamma)$ are beyond $h^g$. A hand configuration model $C$ is constructed by applying kernel density estimation
\[ C(h_c) \equiv \sum_{\gamma \in [-\beta,\beta]} w(h_c(\gamma))N_D(h_c|h_c(\gamma),\sigma_{h_c}) \]  
(27)
where $w(h_c(\gamma)) = \exp(-\alpha\|h_c(\gamma) - h^g\|^2)$ and $\alpha \in \mathbb{R}^+$. $\alpha$ and $\beta$ were hand tuned and kept fixed in all the experiments. One hand-configuration model $C^g$ is learned for each example grasp. The complexity of learning a hand-configuration model is $O(N_g N_g |\Gamma|)$, where $N_g$ is the number of example grasps.

Having completed the description of the learning procedure (Stage 1 in Figure 3) we turn to describing our novel grasp inference procedures (Stages 2-4).

6 Grasp Inference

The inference of grasps on a new object relies on three procedures: (i) a procedure of transferring contact models to the new object; (ii) a grasp generation procedure and (iii) a grasp optimisation procedure. These correspond to Stages 2, 3 and 4 of Figure 3 respectively.

In this paper the novel contribution to grasp inference is that we modify the procedure for transferring contact models so as to improve the quality of the proposed grasps. This is achieved by incorporating the density-divergence measure introduced earlier in the paper.

6.1 Contact Query Density Computation

A query density is, for a particular hand-link and an object model, a density over the pose of that hand-link relative to the object. Query densities are used both to generate and to evaluate the likelihood of candidate grasps. Intuitively, the query density encourages a finger link to make contact with the object at locations with similar local surface properties to those in the training example. A query-density is simply the result of convolving two densities: a contact model density and the object model density. This section describes the formation of a query-density (Figure 3 Stage 2). The main innovation is that we present a new likelihood function for generating and evaluating finger contacts with the object. If we were to directly adopt the previous approach of Kopicki et al. (2015) the query density $Q^F$ would be defined as:
\[ Q^F(s) = \iiint T(s|u,v)V(v,r)M^{L}(u|v)M^{F}(r|v)dvdrdr \]  
(28)
where $V(v,r)$ is the test object-view model (9). As described previously, this is a joint density over frames in global workspace coordinates $v \in SE(3)$ and over surface descriptors $r \in \mathbb{R}^N$. The term $T(s|u,v)$ is the Dirac delta function, since $s$ is determined by $u,v$. The relevant contact model is factored into the product $M^{F}(r|v) = M^{F}(u|v)M^{F}(r|u,v)$, Algorithmically, this density is approximated using importance sampling.

Algorithm 2 Query density formation

1: for all samples $k = 1$ to $NQ^F_c$ do
2: sample ($\hat{v}_k$, $\hat{r}_k$) $\sim V(v,r)$
3: sample from conditional density $\hat{u}_{lk} \sim M^{L}_{k}(u|\hat{r}_k)$
4: set $\hat{v}_{lk} = v_k \circ \hat{u}_{lk}$
5: set $w_{lk} = \exp \left(-\phi d_{d}(\hat{v}_{lk} \circ M^{F}_{k},V)\right)$
6: separate $\hat{v}_{lk}$ into position $\hat{p}_{lk}$ and quaternion $\hat{q}_{lk}$
7: end for
8: normalise weights $w_{lk}$ such that $\sum_k w_{lk} = 1$
9: return $\{\hat{p}_{lk}, \hat{q}_{lk}, w_{lk}\}_{lk=1..NQ^F_c}$

In this paper, the query density is re-defined. Specifically, the term $M^{L}_{k}(r)$ is replaced. This term defines a density over the test-object’s surface shape $r$ around $v$, according to the contact model. But $r$ is only a low-dimensional summary of the ideal surface shape. To avoid the resulting loss of information we may, instead of $M^{L}_{k}(r)$, use a conditional density over the precise shape surface in the neighbourhood of $v$.
\[ P(N^{(v)}|s, M^{F}_{k}) \]  
(29)
where $N^{(v)}$ is the surface patch, on the test object, in the neighbourhood of $v$.

Substituting this for $M^{L}_{k}(r)$ gives us a new query density definition:
\[ Q^F(s) = \iiint T(s|u,v)V(v,r)M^{L}(u|v)P(N^{(v)}|s, M^{F}_{k})dvdrdr \]  
(30)

We desire that the more alike the test surface is to the training surface the higher $P(N^{(v)}|s, M^{F}_{k})$ should be. The density divergence defined earlier is ideal for this purpose:
\[ P(N^{(v)}|s, M^{F}_{k}) \propto \exp \left(-\phi d_{d}(s \circ M^{F}_{k}, V)\right) \]  
(31)
where $\phi$ is a constant and we define $s \circ M^{F}_{k}$ as a composition of a transform and a set. First, recall that $M^{C}_{k} = \{(u_{i,j}, r_{i,j})\}_{j=1..N_{c_k}}$ and that $s = v \circ u$. Then, for every pair $(u_{i,j}, r_{i,j})$ in $M^{L}_{k}$, we simply compose $s$ and $u_{i,j}$
\[ s \circ (u_{i,j}, r_{i,j}) = (s \circ u_{i,j}^{-1}, r_{i,j}) \]  
(32)
So that, when we extend this to $M^{F}_{k}$, we obtain
\[ s \circ M^{F}_{k} = \{(s \circ u_{i,j}^{-1}, r_{i,j})\}_{j=1..N_{c_k}} \]  
(33)
which performs a rigid body transform on the surface density over surface shape, defined by the contact model, so as to map it onto the test object’s actual surface around $v$. To obtain $P(N^{(v)}|s, M^{F}_{k})$ the divergence of the relevant surface patch density $V$ from the transformed contact model density $\hat{v}_{lk}$ should be defined by Eq.(31) and thus by Eq. (22).

As mentioned above, the query density is approximated using importance sampling. When a test object-view model, $V(v,r)$, is presented a set of query densities $Q^F$ is calculated, one for each contact model prototype $M^{F}_{i}$, $l = 1 \ldots N_{c}$, according to (34). The algorithm proceeds as follows. Each $Q^F_{i}$ consists of $NQ^F_{i}$ kernels centred on weighted hand-link poses:
\[ Q^F_{i}(s) \simeq \sum_{k=1}^{NQ^F_{i}} w_{ik} N_{3}(p|\hat{p}_{ik}, \sigma_{p}) \Theta(q|\hat{q}_{ik}, \sigma_{q}) \]  
(34)
with $k$-th kernel centre $(\hat{p}_{ik}, \hat{q}_{ik}) = s_{ik}$, and $P(N^{(i)} | u, M^c_k)$ gives the weight

$$w_{ik} \propto \exp\left(-\phi d_{dd}(s_{ik} \circ M^c, V)\right), \quad s.t. \sum_{k} w_{ik} = 1$$

(35)

The sampling procedure is detailed in Algorithm 2. First, a joint sample $(\hat{r}_k, \hat{r}_k)$ is taken from $V$ (line 2), then $u_{ik}$ is sampled from $M^c_k(u | \hat{r}_k)$ (line 3). This completely specifies a possible hand-link pose and curvature combination (line 4). Then the importance weight is calculated (line 5). The weights are normalised before the set of kernels is returned (lines 8-9).

The parameter $\phi$ was chosen empirically. The time complexity for computing contact query density $Q^c = O(N_Q N_M (1 + \log(N_V)))$, where $N_Q$ is the number of contact query kernels, $N_M$ is the number of contact model kernels and $N_V$ is the number of test object view kernels.

6.2 Grasp Generation

Once query densities have been created for the new object for each contact model prototype, an initial set of grasps is generated (Figure 3, Stage 3). Generation is by a series of random samples. We randomly sample, a grasp-view combination $(g, m)$ and then a hand-link $k$. This triple points to a contact-model cluster $C_l$ and hence to a query density $Q^c_l$. A link pose $s_i$ is then sampled from that query density. Then a hand configuration $h_i$ is sampled from $C^g_m$. Together, the seed pose $s_i$ and the hand configuration $h_i$ define a complete grasp $g_i$, via forward kinematics, including the wrist pose $w$. This is an initial ‘seed’ grasp, which will subsequently be refined. A large set $H^1$ of such initial solutions is generated, where $h(j) = (g_i(j), h_i(j))$ means the $j$th initial solution.

Having generated an initial solution set $H^1$, stages of optimisation and selection are then interleaved to create a sequence of $K$ solution sets $H^k$ for $k = 1 \ldots K$.

6.3 Grasp Optimisation

The final stage of the schema is optimisation of the candidate grasps (Figure 3, Stage 4). The objective of grasp optimisation is, given a candidate equilibrium grasp and a reach to grasp model, to find a grasp that maximises the product of the likelihoods of the query densities and the hand configuration density

$$h^* = \arg\max_{h} \mathcal{L}^{gm}(h)$$

(36a)

$$= \arg\max_{h} \mathcal{L}^{c}_{C}(h) \mathcal{L}^{gm}_{Q}(h)$$

(36b)

$$= \arg\max_{(g_i, h_i) \in \mathcal{Q}^{gm}} C^g_{m}(h_i) \prod_{(i) \in \mathcal{Q}^{gm}} Q^c_{l(i)} \left(\text{kin}^\text{for}_{l(i)}(h_i, h_i)\right)$$

(36c)

where $\mathcal{L}^{gm}(h)$ is the overall grasp likelihood and $C^g_{m}(h_i)$ is the hand configuration model (27). The query density $Q^c_{l(i)}$ is the query density for the cluster prototype $l(i)$ to which hand-link $i$ is mapped. The pose for hand-link $i$ is given by the forward kinematics of the hand, $\text{kin}^\text{for}_{l(i)}(h_i, h_i)$. Finally, $\mathcal{Q}^{gm}$ is the set of instantiated query-densities for grasp-view pair $gm$.

Thus, whereas each initial grasp is generated using only a single query density, grasp optimisation requires evaluation of the grasp against all query densities. It is only in this improvement phase that all query densities must be used. 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 $k$, one step of simulated annealing is applied to every grasp $h(j)$ in $H^k$.

6.4 Grasp Selection

At predetermined selection steps (here steps 1 and 50 of annealing), grasps are ranked and only the most likely 10% retained for further optimisation. During these selection steps the criterion in (36c) is augmented with an additional expert $W(h, h_i)$ penalising collisions for the entire reach to grasp trajectory in a soft manner. 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. 36c:

$$\mathcal{L}^{gm}(h) = \mathcal{L}^{W}(h) \mathcal{L}^{C}_{C}(h) \mathcal{L}^{gm}_{Q}(h)$$

(37a)

$$= W(h, h_i) C^g_{m}(h_i) \prod_{Q^c_{l(i)} \in Q^{gm}} Q^c_{l(i)} \left(\text{kin}^\text{for}_{l(i)}(h_i, h_i)\right)$$

(37b)

where $\mathcal{L}^{gm}(h)$ 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 that the number of hand-links in contact during a grasp varies across grasps and views. Thus the number of query densities $N^{\text{max}}_Q \cdot N^{\text{max}}_m$ also varies, and so the values of $\mathcal{L}^{gm}$ and $\mathcal{L}^{g_m}$ cannot be compared directly. Given the grasp with the maximum number of involved links $N^{\text{max}}_Q$, we therefore normalise the likelihood value (37a) with

$$\|\mathcal{L}^{gm}(h)\| = \mathcal{L}^{W}(h) \mathcal{L}^{C}_{C}(h) \left(\mathcal{L}^{gm}_{Q}(h)\right)^{\frac{N^{\text{max}}_Q}{N^{\text{max}}_Q}}.$$  

(38)

It is this normalised likelihood $\|\mathcal{L}^{gm}\|$ that is used to rank all the generated grasps across all the grasp-view pairs during selection steps. After simulated annealing 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.

6.5 Grasp Execution

The remaining best scoring hand pose $h^*$, evaluated with respect to (38), is then used to generate a reach to grasp trajectory. This is the command sequence that is executed on the robot.

7 Experimental Study

This section is structured as follows. First, the creation of challenging data set is described. Second, the algorithmic variants tested are enumerated. Then, three experiments are presented in turn. Each of these varies in the size of the training set. Experiment 1 trains with five grasps and Experiment 2 with ten. Experiment 3 introduces training from robot generated grasps. A discussion examines each of the hypotheses in the light of the results.
7.1 Test set creation

In preparation for the experimental evaluation, a challenging test set was created. This used 40 novel test objects (Figure 2). The test cases were object-pose pairs relative to a single, fixed viewpoint. The object-pose combinations were chosen to be particularly challenging by using a pose that reduced the typical surface recovery from the fixed view. Some objects were employed in several poses, yielding a total of 49 object-pose pairs. Because both object and pose are controlled this means that we can test algorithms using a paired comparisons statistical methodology. This can yield statistically significant results for small numbers of test grasps.

The question arises as to whether it can be validated that this new data-set is indeed challenging. A previous single-view data set was also generated in Kopicki et al. (2015), using many of the same objects, but without deliberately challenging poses. The performance of the algorithm presented in Kopicki et al. (2015) on this original single view test set was 77.8% (35/45). Therefore, to verify the challenging nature of the new test data-set, that algorithm was tested on the new single-view dataset. To make the two test-sets comparable the same set of five training grasps, presented in Kopicki et al. (2015), was used. Testing on the new dataset the performance of the algorithm reduced to 59.2% (29/49). Since we hypothesized that the harder data set should produce a lower success rate we applied a one-tailed $t^2$ test, using Fisher’s exact test, which gave a $p$-value of 0.043. Thus the difference between the success rates is statistically significant and is unlikely to have been caused by chance. We therefore accept the hypothesis that the new data set of object-pose pairs is more challenging than the previous data set.

7.2 Algorithmic variations

The paper has presented three main innovations. These are: (i) a view based representation; (ii) a method for merging contact models; (iii) a new evaluation method for calculating the likelihood of a generated grasp. In addition, the surface descriptor may be either principal curvatures or fast point feature histograms. It is clearly desirable to evaluate which of these innovations is most effective, and to study how well they work in combination. The six 'algorithmic variations' are listed in Table 1. The algorithm reported in Kopicki et al. (2015) is variation A1. Note that variants A1 and A2 are not to be confused with Algorithm 1 (contact model clustering) and Algorithm 2 (query density computation), which are components of all variants.

The algorithms presented depend on a number of parameters, which have been presented earlier in the text. It would not be possible to systematically tune these parameters using grid search, but a small number of informal
The same parameter settings were used for all algorithmic experiments. The values used are presented in Table 2.

### 7.3 Experiment 1

This experiment tests the hypothesis that, even without an enlarged set of training grasps, the combined innovations, present in A4 and A6, will improve the grasp success rate. Thus, variations A1, A4 and A6 were trained with the five grasps from Kopicki et al. (2015). A paired comparisons experiment with all 49 test cases was executed. This led to a grasp success rate of 59.2% (as reported above) for A1, a success rate of 75.5% (37/49) for A6, and 77.6% (38/49) for A4. Although the success rates for both A4 and A6 are higher than that for A1, using the two-tailed McNemar test for the difference between two proportions on paired data these differences are not statistically significant (p=0.1175 for A6:A1, p=0.0665 for A4:A1).

### 7.4 Experiment 2

This experiment tests hypotheses H2 and H3. H2 is the hypothesis that view-based grasp modelling enables better generation of grasps for thick objects. H3 is the hypothesis that the success rate progressively increases as innovations 1 to 3 are added. Experiment 2 also provides evidence to test hypotheses H4 and H5, that the grasp success rate will improve as training data is added, and do so faster if all innovations are deployed. The training set was therefore increased to 10 grasps. Figure 2 shows the objects used for training. To test H2 and H3, six algorithmic variations A1:A6 were tested, as detailed in Table 1. These progressively add the three innovations. Algorithm A1 is the version described in Kopicki et al. (2015). A2 is A1 plus view based organisation of grasp models. Algorithm A3 is A2 plus improved evaluation of grasp likelihood by density comparison. Algorithm A4 is A3 plus contact model merging. All of algorithms A1-A4 use principal curvatures as the features. As a final step, we also test the robustness of the method to changes in the surface descriptors used. Variants A5 and A6 are the equivalents of A3 and A4 respectively, but use FPFH as the surface descriptor instead of curvatures.

The success rates for each algorithmic variation are shown in Table 3. As the innovations are added the success rate rises. Applying McNemar’s test for the difference in proportions for paired data, A4 and A6 dominate A1 and A2, and these differences are highly statistically significant. A full table of p-values is shown in Table 4.

### 7.5 Experiment 3

This experiment provides further evidence to test hypotheses H4 and H5. Specifically, it tests what happens to the grasp success rate when the training data continues to grow. Because human demonstration is time consuming, growing the training data can be achieved by using grasps generated autonomously by the robot as additional training data. This was implemented here using a leave-one-out training regime. The robot was trained with all successful grasps excluding the test object. Thus, the algorithm is trained autonomously by the robot as additional training data. This improves the grasp success rate when the training data continues to grow.

This autonomous training (AT) allows the algorithm to scale. The robot was trained with all successful grasps excluding the test object. This autonomous training regime was tested using the baseline variant A1 and variant A4. We refer to these

---

<table>
<thead>
<tr>
<th>Alg pair</th>
<th>p-value</th>
<th>Alg pair</th>
<th>p-value</th>
</tr>
</thead>
<tbody>
<tr>
<td>A4(5):A1(10)</td>
<td>0.0153</td>
<td>A4(10):A1(5)</td>
<td>0.0218</td>
</tr>
<tr>
<td>A6(5):A1(10)</td>
<td>0.0162</td>
<td>A6(10):A1(5)</td>
<td>0.0153</td>
</tr>
<tr>
<td>A4(10):A1(10)</td>
<td>0.0009</td>
<td>A4+AT:A1(10)</td>
<td>0.0002</td>
</tr>
<tr>
<td>A6(10):A1(10)</td>
<td>0.0036</td>
<td>A4+AT:A2(10)</td>
<td>0.0013</td>
</tr>
<tr>
<td>A4(10):A2(10)</td>
<td>0.0033</td>
<td>A4+AT:A3(10)</td>
<td>0.0265</td>
</tr>
<tr>
<td>A6(10):A2(10)</td>
<td>0.0033</td>
<td>A4+AT:A5(10)</td>
<td>0.0433</td>
</tr>
<tr>
<td>A4+AT:A1(5)</td>
<td>0.0056</td>
<td>A4+AT:A1+AT</td>
<td>0.0095</td>
</tr>
</tbody>
</table>

---

1. The training grasps varied from 49 and 50. We trained with 40 successful grasps from A4 in Experiment 2 and 10 demonstrated grasps. If the test grasp-object-pose had been successful in Experiment 2 it was removed from the training set hence there were 49 training examples. We used the same training set for A1+AT and A4+AT.
variants with autonomous training as A1+AT and A4+AT respectively. Grasps are shown in the multimedia extension. Since there was no appreciable difference between A4 and A6 in Experiment 2 we did not create an additional variant for A6. For A1+AT the success rate rose to 63.3% (31) and for A4+AT the success rate was 87.8% (43/49). A two-tailed McNemar test for paired comparisons data shows that the differences between A4+AT and several other variants (A1, A2, A3, A5) are statistically significant (Table 4, Figure 14).
7.6 Discussion

The hypotheses can be considered in order. Hypothesis H1 (even without an enlarged set of training grasps the combined innovations 1-3 will improve the grasp success rate.) was tested in Experiment 1. There is support for this in that the grasp success rate for A1(5) is 59.2%, whereas for A4(5) and A6(5) it is 77.6% and 75.5% respectively. Although this difference is nearly 20% it is not, however, statistically significant. This result thus provides moderate support for hypothesis H1.

Hypothesis H2 (view-based grasp modelling enables better generation of grasps for thick objects) can be tested by identifying which object-pose pairs possess a deep back-surface. Such situations force the robot to generate a grasp with a wide hand-aperture, so as to place a finger on that hidden surface. These so-called thick object-pose pairs in the dataset are listed in Table 5. To test the hypothesis the pairs of grasp outcomes for algorithmic variants A1(10) and A2(10) can be compared for this subset of 16 object-pose pairs. For this subset A1(10) has a success rate of 18.75% (3/16), whereas A2(10) has a success rate of 68.75% (11/16). Using a two-tailed McNemar test the difference is statistically significant (p-value=0.0133). This provides strong evidence in support of hypothesis H2. In addition, although it presents grasps from A4+AT and A1+AT, Figure 12 shows specific instances of grasps where this ability to grasp hidden back-surfaces of thick objects can be seen, such as the coke bottle, guttering, spray can and stapler.

Hypothesis H3 (the grasp success rate improves as innovations 1-3 are added) is again supported by the monotonic increase in grasp success rate as innovations 1, 2 and 3 are added in order. The success rate rises from 55.1% (A1(10)), through 57.1% (A2(10)), 69.4% (A3(10)), to 81.6% (A4(10)). Some of these differences are statistically significant. Notably, both variations A4(10) and A6(10) are better than A1(10) and A2(10) and the differences are either highly or extremely significant. This provides good support for hypothesis H3.

Hypothesis H4 (with all innovations the grasp success rate improves as the training data increases) can be tested by examining the change in success rate as data is added. Algorithm A4 was tested with the full range of training set sizes of 5, 10 and 49-50 grasps. The evidence supports H4, since the success rate rises from 77.6% (A4(5)) through 81.6% (A4(10)) to 87.8% (A4+AT).

Hypothesis H5 (with all three innovations, learning is better able than the baseline algorithm to exploit an increased amount of training data) can be tested by comparing the figures for A4 to those for A1, across all three training regimes. For A1 the corresponding success rates are 59.2%, 55.1% and 63.3%. Thus, when moving from five training examples to fifty, whereas algorithm A1 improved by 4.1%, A4 improved by 10.2%. This supports hypothesis H5. We suggest that this is because of the use of contact model merging. Figure 15 shows that the compression ratio (initial models:clusters) increases as the training set grows. This shows the effect on the number of models. The effect of this compression on the grasp success rate is shown by comparing the success rate of A4 with A3 (81.6%:69.4%), A4+AT with A5 (81.6%:71.4%). Thus, when adding contact model merging the improvement is of the order of 10%. This provides support for the idea that the advantage extracted from additional data is due in part to contact model merging.

Finally, hypothesis H6 (with all innovations the algorithm dominates the baseline algorithm without any innovations) is tested by examining the results of all three experiments. Variants A4 and A6 outperform the corresponding version of A1, regardless of the training regime. Figure 14 shows that all of these differences, bar A4(5) and A6(5) versus A1(5), are statistically significant. In addition, Figure 14 shows that A4+AT dominates A1 regardless of the training regime used, including the best version (A1+AT), and these differences are either highly (p < 0.01) or extremely (p <
This paper has presented a number of technical innovations that, when combined, improve grasp performance. These were: view-based grasp modelling, contact-model merging, and new method for generation and evaluation of contacts. These innovations enable an increase in the number of training examples. This, in turn, enables an change to the training methodology in which the grasps generated and executed for novel objects are fed back as further training examples. An empirical evaluation of the algorithms, on a data-set of challenging grasp-view combinations, showed that there are substantial differences in grasp success rate between some variations, from 55.1\% for the algorithm reported in Kopicki et al. (2015) to 87.8\% for a variant that employs all three innovations, plus autonomous training. Furthermore, these differences are statistically significant.

Acknowledgements

The authors gratefully acknowledge funding from the PaC-Man project FP7-IST-600918. The sourcecode for the algorithms reported here is publically available as part of the Golem robot manipulation stack: https://github.com/RbotsLab/Golem.

References


Table 6. Computation time, per test object, for algorithms A1 and A4.

<table>
<thead>
<tr>
<th>Experiment Number</th>
<th>Query density computation (secs)</th>
<th>Generation &amp; Optimisation (secs)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>A1</td>
<td>A4</td>
</tr>
<tr>
<td></td>
<td>A1</td>
<td>A4</td>
</tr>
<tr>
<td>1</td>
<td>0.41</td>
<td>7.95</td>
</tr>
<tr>
<td>2</td>
<td>1.04</td>
<td>14.47</td>
</tr>
<tr>
<td>3</td>
<td>1.70</td>
<td>17.77</td>
</tr>
</tbody>
</table>

Table 7. Most challenging object-pose pairs.

<table>
<thead>
<tr>
<th>Object (pose)</th>
<th># succ's from 11</th>
</tr>
</thead>
<tbody>
<tr>
<td>Kettle</td>
<td>6</td>
</tr>
<tr>
<td>Large spray-can</td>
<td>6</td>
</tr>
<tr>
<td>Large funnel (sideways)</td>
<td>5</td>
</tr>
<tr>
<td>Large saucepan</td>
<td>5</td>
</tr>
<tr>
<td>Mug 1 (upside-down)</td>
<td>4</td>
</tr>
<tr>
<td>Mug 3</td>
<td>4</td>
</tr>
<tr>
<td>Mug 5 (upside-down)</td>
<td>4</td>
</tr>
<tr>
<td>Mug 4 (upside-down)</td>
<td>3</td>
</tr>
<tr>
<td>Small saucepan</td>
<td>3</td>
</tr>
<tr>
<td>Tennis ball</td>
<td>2</td>
</tr>
<tr>
<td>Danish ham (sideways)</td>
<td>0</td>
</tr>
</tbody>
</table>

0.001) statistically significant. These results provide very strong support for H6.

We also note that there is no evident difference in performance caused by the choice of surface descriptor.

Next, we show the computation times for variants A1 and A4 in Table 6. These comparisons were made on a PC with two Xeon E5-2650V2 CPU processors. This comparison shows that the new algorithms are slower in terms of query density computation. This is because of the use of the new evaluation function, which is roughly 26 times more expensive than previously. This factor, however, is constant, so that as the number of training grasps rises A1 will eventually exceed A4 in terms of computation time. The absolute time for A4 is higher than would be suitable for real-world use. The algorithm, is, however, well suited for GPU implementation, which would significantly reduce absolute computation time.

Finally, we also highlight the most challenging object-pose pairs. There were eleven grasps executed per object-pose pair across all variants and experiments. Table 7 shows the objects for which the number of successes was six or lower. It is worth noting that these objects were difficult mostly because they needed to be grasped around their body, which was very close to the maximum aperture of the dexterous hand that was used.

8 Conclusions

Dexterous grasping of novel objects given a single view is an important problem that needs to be solved if dexterous grasping is to be deployed in real-world settings. While good progress has been made on simple pinch grasping from a single view, dexterous grasping is significantly more challenging, due to the increased dimensionality of the hand, and thus the search space.


