VGQ-CNN: Moving Beyond Fixed Cameras and Top-Grasps for Grasp Quality Prediction

Anna Konrad
Hamilton Institute
Maynooth University
anna.konrad.2020@mumail.ie

John McDonald
Department of Computer Science
Maynooth University
john.mcdonald@mu.ie

Rudi Villing
Department of Electronic Engineering
Maynooth University
rudi.villing@mu.ie

***Abstract*—We present the Versatile Grasp Quality Convolutional Neural Network (VGQ-CNN), a grasp quality prediction network for 6-DOF grasps. VGQ-CNN can be used when evaluating grasps for objects seen from a wide range of camera poses or mobile robots without the need to retrain the network. By defining the grasp orientation explicitly as an input to the network, VGQ-CNN can evaluate 6-DOF grasp poses, moving beyond the 4-DOF grasps used in most image-based grasp evaluation methods like GQ-CNN. To train VGQ-CNN, we generate the new Versatile Grasp dataset (VG-dset) containing 6-DOF grasps observed from a wide range of camera poses. VGQ-CNN achieves a balanced accuracy of 82.1% on our test-split while generalising to a variety of camera poses. Meanwhile, it achieves competitive performance for overhead cameras and top-grasps with a balanced accuracy of 74.2% compared to GQ-CNN's 76.6%. We also propose a modified network architecture, Fast-VGQ-CNN, that speeds up inference using a shared encoder architecture and can make 128 grasp quality predictions in 12ms on a CPU. Code and data are available at https://aucoroboticsmu.github.io/vgq-cnn/.**

***Index Terms*—grasping, robotics, machine learning, flexible, mobile robot, grasp quality, CNN, 6-DOF grasps**

I. Introduction

Robotic grasping is a persistent challenge within robotics research due to its inherent complexity. While robotic grasping techniques are widely used in industrial setups where the influence of uncertainties can be reduced, the usage in less structured environments is still an active field of research. Multiple different approaches for grasping unknown objects have been developed, with the camera usually fixed to a pose above the object and the robot in a stationary pose close to the camera [1]–[6]. These fixed relationships between camera, object and robot limit the applicability of the trained networks in non-lab scenarios. When attempting to apply such networks to a robot with a different camera pose or when working with multiple cameras, the networks have to be retrained each time as they do not generalise to versatile viewpoints. Furthermore, usage with mobile manipulators (e.g. the PAL TIAGo, Toyota HSR, etc.), which constantly change their pose in relation to the object, is not practical if constrained to fixed camera poses.

Another related limitation is the use of 4-DOF grasp representations in current image-based approaches. By assuming alignment between the grasps and the table normal or the camera principal axis [1], [3]–[8], the grasps can be represented as a 2-D position and 1-D orientation within the image frame. In commonly reported configurations that mount the camera above the object [1], [5], [7], [9] these networks are limited to top-grasps, where the grasp approach is along the table normal. Consequently, the variety of grasp orientations is limited significantly, creating constraints for possible path planning pipelines downstream.

We base our work on GQ-CNN by Mahler et al. [1], a neural network for grasp quality evaluation of top-grasps from an overhead camera. While there have been follow-up networks for GQ-CNN [10]–[12], these extensions have focused on objects in clutter and alternative gripper mechanisms. As such, these networks maintain the 4-DOF grasp representation and therefore are also limited to top-grasps. While a fully convolutional version of GQ-CNN improves run-time [7], the necessary parameterisation to allow for 6-DOF grasps would lead to a significant increase in the size of the output tensor as currently formulated.

We move GQ-CNN beyond fixed cameras and top-grasps with the Versatile Grasp Quality Convolutional Neural Network (VGQ-CNN). The network can evaluate 6-DOF grasp proposals of objects on a planar surface viewed from a wide range of camera poses above the object. The position and viewing angle of the camera can be varied within a space as large as $2.1m^3$. We introduce the Versatile Grasp dataset (VG-dset) to include these variations in camera poses and grasp orientations, significantly exceeding the range available in commonly used datasets such as [1], [9], [13]. We further propose using a shared image encoder with our alternative network architecture, Fast-VGQ-CNN, to speed up the evaluation of multiple grasps during run-time. In summary, the contributions of this paper are:

- VG-dset: A versatile 6-DOF grasp quality dataset including 7.1 million grasps over an extended range of camera poses.
- VGQ-CNN: A network capable of predicting the quality of 6-DOF grasps under a wide range of camera poses.
- Fast-VGQ-CNN: An alternative network structure to VGQ-CNN to speed up inference and predict up to 128 grasps in $8ms$ on a NVIDIA RTX 2060 GPU and $12ms$ on an Intel i7-10750H CPU.

This publication has emanated from research supported in part by Grants from Science Foundation Ireland under Grant numbers 18/CRT/6049 and 16/RI/3399.

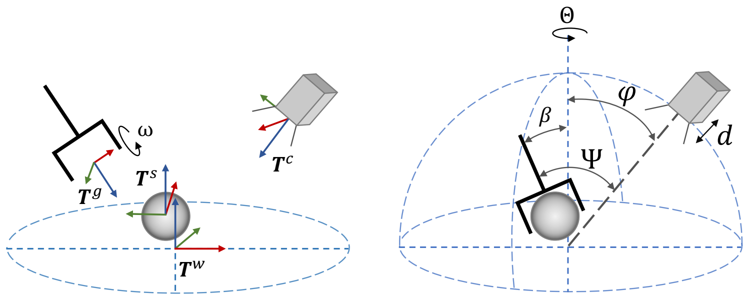

Fig. 1. Visualisation of coordinate frames, parameters and relative angles of camera-object-gripper configurations.

II. Related work

Identifying good grasping poses for unknown objects has been an active field of research for many years [14], [15]. Grasps can either be directly generated via generative methods [5], [7], [8], [16] or sampled and subsequently ranked by discriminative methods like GQ-CNN [1], [2], [17], [18]. Discriminative approaches are typically coupled with a higher runtime and more flexibility, while the parameterisation in generative approaches allows for low latency while limiting flexibility [15]. Independent of the approach, many of the presented methods have been developed for a camera pose fixed within a small tolerance in relation to the object [1], [2], [5], [7].

In addition to restricted camera poses, the methods usually only allow for 4-DOF grasps. This is especially apparent when taking into account the grasp representations utilised in those methods, often projecting the Tool Centre Point (TCP) and gripper orientation into the image plane and defining it as an oriented rectangle [5], [8], [9], [13] or pre-processing the image to align it with the grasp [1], [7]. Due to this representation the variety of grasps is reduced significantly, often only allowing for top-grasps [1], [5], [7], [9].

When moving away from those grasp representations and towards unrestricted 6-DOF grasps, the dimensions and complexity of the problem increase. Methods providing this functionality usually utilise point clouds and process them directly [18], in a voxelised grid [16], [17] or as object meshes through shape completion [19], [20]. One of the major disadvantages that working with unrestricted 6-DOF grasps often poses is increased run-time, typically taking several seconds [17]–[19]. While generative methods for 6-DOF grasps can reduce run-time down to $10ms$ with specialised hardware [16], [21], the parameterisation limits the number and flexibility of proposed grasps.

Another way 6-DOF grasps can be facilitated is by attaching cameras to the wrist of robotic manipulators and then iteratively approaching the object [22], [23]. These approaches might be difficult to incorporate into the overall path planning and collision checking of a robot, since they produce control outputs [22] or actions [23] rather than goal poses. Instead, they could be used for a closed-loop approach of the grasp pose after a pre-grasp position provided by an overall grasp planner has been reached.

Robotic grasping for real-world applications has to be fast and flexible in order to be used efficiently. Approaches need versatility in the configuration of grasp orientations and camera poses to cater for dynamic setups. Recent approaches have developed generative 6-DOF grasp proposal methods based on depth images [24]. While such approaches can compute grasp proposals within $0.5s$, they employ a camera positioned directly above the object and hence are not applicable to situations involving a wider variation of camera poses, e.g. with mobile manipulators. Extending such work to a discriminative grasp sampling method can enhance flexibility in the number of grasps that can be proposed. This enables not only iterative refinement of grasps [18], but could also be used to tailor grasp sampling on specific objects or object areas. Following this approach, we present VGQ-CNN, a 6-DOF grasp quality prediction network based on depth images that is robust to a wide range of camera poses. We furthermore provide our alternative network architecture Fast-VGQ-CNN, which speeds up grasp prediction significantly to enhance usability for real-world applications.

III. Problem formulation

We consider the problem of predicting grasp qualities for flexible grasp orientations with a parallel jaw gripper as observed from a wide range of camera perspectives. The environment is limited to a single object placed on a planar surface. The goal is to train a network which can predict grasp qualities based on depth images and grasp proposals from a wide range of camera poses without having to retrain the network for each new camera pose. In addition, the network should be able to evaluate a variety of 6-DOF grasp orientations where the gripper approach axis is not necessarily aligned with a pre-specified fixed axis, e.g. the camera principal ray. Such a network caters for scenarios that involve changing the pose of the object or camera between setups, whilst also including mobile manipulators, where the relationship between camera and object varies dynamically.

A visualisation of the setup can be seen in Fig. 1. The object meshes are placed on a planar surface in predefined, stable resting poses, with their coordinate frame T^s being in close proximity to the origin of world coordinate frame T^w. The camera position is defined in spherical coordinates in relation

TABLE I
PARAMETER OVERVIEW

Dataset	φ_{max}	d_{min}	d_{max}	β_{max}
VG-dset	$70°$	$0.4m$	$1.1m$	$90°$
DexNet2.0	$5.7°$	$0.65m$	$0.75m$	$5°$

to T^w, with the distance to the origin d, the elevation angle φ and the polar angle θ defining its position. The camera principal ray is pointing towards the origin of T^w and the x-axis is parallel to the table surface, orienting the camera frame, T^c, such that the camera views the table and object horizontal and upright, respectively. The gripper frame, T^g, is defined such that the x-axis lies between the contact points of the parallel jaw gripper and the z-axis denotes the linear approach direction.

We define the camera as placed above the planar surface with $d \in [0.4m, 1.1m]$, $\theta \in [0°, 360°]$ and $\varphi \in [0°, 70°]$, which corresponds to a volume of roughly $2.1m^3$. The limits were chosen in reference to the typical accessible range of mobile manipulators, e.g. PAL TIAGo robot [25], Toyota HSR [26], grasping objects placed on a table. Configurations where the camera is below or level with the planar surface, e.g. when grasping objects from a shelf, and those with objects placed close to the edge of the surface are excluded.

Since the quality and usability of a grasp is influenced not only by the gripper pose in relation to the table and the object, but also by the visibility from the camera, we define a set of relative angles to parameterise the space of unique camera-object-gripper configurations. The angle between the gripper z-axis and the table normal is denoted as β, while the angle between the gripper z-axis and the camera principal ray is denoted as Ψ. Rotating the grasp around the grasp x-axis is denoted by ω and does not alter the position of the contact points. For viable 6-DOF grasp poses, the angle between the gripper z-axis and the table normal is constrained to $\beta \in [0°, 90°]$. Since $\beta \geq 90°$ would likely cause collisions with the table, we exclude such grasps from our setup.

IV. Dataset

We create a new 6-DOF grasp quality dataset called the Versatile Grasp dataset (VG-dset) to satisfy the specifications in our problem formulation in section III. The complete overview of parameter ranges for VG-dset in comparison to DexNet2.0 [1] can be seen in Table I. We base VG-dset on the pre-sampled antipodal grasps $g \in \mathcal{G}(o)$ included in DexNet2.0 [1], where the object meshes $o \in \mathcal{O}$ are taken from the KIT [27] and 3DNet [28] mesh datasets.

Dataset preparation consists of two stages: dataset rendering and dataset sampling. Throughout the dataset rendering process, datapoints are created by rendering images and storing grasp poses and grasp quality values. A total of 131 million grasps are stored in the process, providing a baseline which can be sampled for different purposes. The dataset sampling process is necessary since certain minimal ratios, e.g. between ground-truth negative and ground-truth positive grasps, have to be satisfied to ensure successful training of VGQ-CNN.

```
Input: Object mesh $o$, Grasps $\mathcal{G}(o)$, Camera distance $\{d_{\min}, d_{\max}\}$,
       Camera polar angle $\{\theta_{\min}, \theta_{\max}\}$, Camera elevation angle
       $\{\varphi_{\min}, \varphi_{\max}\}$, Camera Intrinsics Matrix $K$, Number of images
       per stable pose $n$
Output: Depth image $\mathcal{I}$, Grasp image coordinates $(u(g), v(g))$, Grasp
        distance to camera $z(g)$, Grasp quaternion in camera frame
        $q(g^c)$, Grasp quality $\rho(g)$ and relative angles $\Psi(g), \beta(g)$
for $1 : n$ do
    // Sample camera poses
    Sample $d, \theta$ and $\varphi$ from uniform distributions;
    Let $\mathbf{T}_c^w$ be based on $p_c^w(d, \theta, \varphi)$;
    Render depth image $\mathcal{I}$ from $\mathbf{T}_c^w$; save $\mathcal{I}$, $d, \theta$ and $\varphi$;
    for $g \in \mathcal{G}(o)$ do
        // Apply random gripper rotation around grasp x-axis
        $\mathbf{R}_g = \mathbf{R}_g \cdot \mathbf{R}_X(\omega)$, $\omega \sim \mathcal{U}(0, 2\pi)$;
        if $g^w.z$ is pointing in positive $w.z$ then
            // Grasp coming from under the table
            $\mathbf{R}_g = \mathbf{R}_g \cdot \mathbf{R}_X(\omega)$, $\omega = \pi$;
        if linear approach of $g$ collides with object or table then
            Set quality of grasp $\rho(g) = 0$ ;          // Negative grasp
        $(u(g), v(g)) = \text{Project}(\mathbf{T}_g^c, K)$ ; // Project grasp into image
        Calculate $z(g) = \|p_g^c\|$, $\beta(g) = \arccos(\frac{\mathbf{R}_g^c.z[2]}{|\mathbf{R}_g^c.z|})$ and
            $\Psi(g) = \arccos(\frac{\mathbf{R}_g^w.z[2]}{|\mathbf{R}_g^w.z|})$;
        Save $(u(g), v(g))$, $z(g)$, $q(g^c)$, $\rho(g)$, $\beta(g)$ and $\Psi(g)$;
```

Fig. 2. Algorithm for rendering VG-dset.

The process can also be used to allow for different dataset compositions, e.g. excluding certain camera or grasp configurations. While the dataset generation process could be adjusted to generate a single, balanced dataset to be used for training, the time-consuming process of rendering the data and checking the grasps would have to be repeated each time a different dataset composition was desired.

A. Dataset rendering

We base our dataset rendering algorithm on the DexNet2.0 implementation by Mahler et. al. [1] available on github[1] with key changes in the camera poses, grasp alignment and grasp representation. Each object mesh $o \in \mathcal{O}$ has an average of 9 stable resting poses $s \in \mathcal{S}(o)$. When rendering the dataset, we repeat the algorithm detailed in Fig. 2 for each stable resting pose of each of the 1494 object meshes.

Moving away from top-grasps and allowing for a variety of grasp orientations represents a challenge to both the grasp preparation and representation. For stationary cameras and top-grasps, some sense of alignment between the grasp orientation and the camera position is usually assumed. In DexNet2.0 [1], this is realised by rotating the gripper around ω to minimise β and discarding grasps with $\beta \geq 5°$. By projecting the gripper Tool Centre Point (TCP) and gripper x-axis into the image plane and rotating/cropping the image accordingly, Mahler et al. [1] reduce the grasp representation to an aligned image and the distance between the grasp T^g and the camera T^c. This is similar to the reduced grasp representations in [3], [4], [8], all of which are defined as positions and rotations in the 2-D image plane.

[1]https://github.com/BerkeleyAutomation/dex-net

Since VG-dset should include 6-DOF grasps, we aim to include grasps with the same grasp x-axis and varying approach angles. For example, a top-grasp rotated by 90° around ω would end up grasping the object parallel to the table. Ideally, both grasps should be included and proposed to the path planning system of a robot to choose the best grasp. We augment the data with a variety of grasp orientations in VG-dset by applying a random rotation around ω to the grasps with each new camera pose. We flip grasps approaching from under the table with $\beta > 90°$ by rotating them once more for 180° around ω. We use this approach to provide flexibility to various sampling schemes (see section IV-B) and to simplify constraining the final grasp orientation. Note that rotating the gripper around ω does not change the robust Ferrari-Canny metric [29] of a given grasp, and therefore does not influence the grasp quality $\rho(g)$ aside from collisions checked after the final grasp orientation has been decided.

We then, like in DexNet2.0 [1], proceed to collision checking (Fig. 2, lines 9-10), setting the robust Ferrari-Canny value for grasps colliding with the object or table $\rho(g) = 0$ and thereby marking the grasp as ground-truth negative. After collision checking, the gripper TCP is projected into the image plane in order to calculate its (u, v) position in the image in pixel coordinates.

Note that we render $n = 100$ images for each stable resting pose $s \in \mathcal{S}(o)$ of each object mesh $o \in \mathcal{O}$, while $n = 50$ for the DexNet2.0 dataset generation. Since our data includes a wider range of camera poses and grasp orientations, sampling more images per stable pose allows us to subsample the resulting dataset in order to create specific dataset characteristics as explained in the following section.

B. Dataset sampling

The 130.8 million grasps from the dataset rendering stage need to be undersampled to generate VG-dset, a dataset for VGQ-CNN. This undersampling process is not needed in DexNet2.0 [1] due to their dataset parameters as described in Table I. To train a network on our new problem formulation, undersampling the datapoints becomes necessary for three major reasons:

- Filtering grasp and/or camera configurations for training VGQ-CNN, e.g. removing grasps with $\Psi \geq 90°$.
- Adjusting the positivity rate $pos_r = \frac{100 \times \#positive\ grasps}{\#all\ grasps}$, since it affects the training loss and therefore the network performance.
- Balancing the number of grasps across the camera and grasp configurations.

The large size of the rendered dataset enables differing dataset compositions to be sampled and their effect on the network performance to be tested.

To generate VG-dset, grasps with $\Psi \geq 90°$ are removed from the dataset. This maximum value is set so that grasps approaching the object from behind relative to the camera position are excluded, since they are likely to be occluded by the object. Second, we undersample negative grasps to ensure a consistent positivity rate pos_r over beta. Prior to undersampling, pos_r declines with increasing β, since grasps with a high β tend to collide with the table more often. The native, rendered dataset has a varying positivity rate with a mean of $pos_r = 6\%$, while the positivity rate for DexNet2.0 is $pos_r = 19\%$. We adjust pos_r by undersampling negative/positive grasps based on the sampling rate $sample\_rate_x$. Since pos_r is dependent on β, we calculate the sampling rate for undersampling negative grasps as,

$$sample\_rate_{neg}(\Delta\beta) = \frac{\frac{100}{19} \times pos_r(\Delta\beta) - pos_r(\Delta\beta)}{100 - pos_r(\Delta\beta)}$$

where, $\Delta\beta$ is varied in steps of 5°. The undersampling rate for positive grasps is set to $sample\_rate_{pos} = \frac{1}{sample\_rate_{neg}}$. Undersampling rates of $sample\_rate_x > 1$ are set to 1. We then skip positive/negative grasps randomly based on their sampling rate for the given β. As a third step, we undersample the remaining grasps to have a uniform sample size over φ and β. The resulting dataset, VG-dset, consists of 7.2 million grasps. As shown in section VII-C, training on datasets with 2 million or more grasps shows a constant performance, while training on dataset with fewer than 1 million grasps shows signs of overfitting the dataset on VGQ-CNN.

We divide both DexNet2.0 and VG-dset in an object-wise training, validation and test split of 80-10-10, using the same objects for the test sets in DexNet2.0 and VG-dset. This allows us to compare performance between VGQ-CNN and GQ-CNN on the same test data in section VII-A.

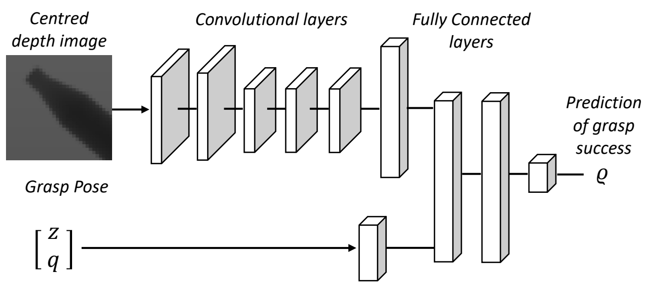

Fig. 3. Architecture of VGQ-CNN. The network consists of an image stream (upper), a pose stream (lower) and a merge stream (down-end) to predict grasp success. The input image is a 32×32 pixel depth image and the pose is given by the distance between the camera and the gripper TCP, z, and the orientation quaternion, $q \in R^4$, where $|q| = 1$. It outputs a prediction of grasp success $\varrho \in [0, 1]$.

V. Network architecture

In DexNet2.0 [1], as well as in other image based grasp predictors [4], [5], [8], [9], [13], the TCP of each grasp proposal is projected into the image to be represented in the 2-D image plane, e.g. as an oriented rectangle. In these approaches, the grasp orientation is constrained to some arbitrary axis that is not explicitly specified as a network input. To represent grasp orientations that are not constrained to align with an arbitrary axis (such as the camera principal ray), requires an alternative grasp specification for the network.

To represent the varying grasp orientations in VGQ-CNN we use a grasp representation that defines the full 3-DOF

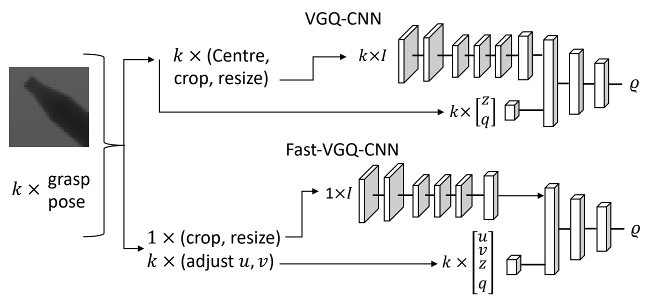
VGQ-CNN
$k \times$ (Centre, crop, resize)
$k\times I$
$k\times \begin{bmatrix} z \\ q \end{bmatrix}$
ϱ
Fast-VGQ-CNN
$k \times$ grasp pose
$1\times I$
$1 \times$ (crop, resize)
$k \times$ (adjust u, v)
$k \times \begin{bmatrix} u \\ v \\ z \\ q \end{bmatrix}$
ϱ

Fig. 4. Comparison of pipelines for grasp quality evaluation with VGQ-CNN and Fast-VGQ-CNN. For VGQ-CNN, k images have to be generated from a single image and fed to the network with k poses. For Fast-VGQ-CNN, the image is processed once and fed into the image stream of the network. The output is then fed to the rest of the network together with k poses.

grasp orientation in the form of a quaternion q. (While we use quaternions as orientation representations, also used in VGN [16], this could be exchanged with other representations like Euler angles.)

Using the approach of GQ-CNN we specify the grasp image coordinates (u, v) implicitly in the image. This is achieved by projecting the gripper TCP into the image $\mathcal{I}$, cropping the image around the grasp coordinates (u, v) and subsequently resizing it to a 32×32 pixel image. The remaining variables of the 6-DOF grasp, the distance of the TCP to the camera, z, and the grasp quaternion, q, are provided to VGQ-CNN as an input in the pose stream, see Fig 3.

We keep the general network structure of GQ-CNN, consisting of parallel image- and pose-streams with convolutional and fully connected layers. The two streams are combined into a single stream towards the end of the network. Apart from including the quaternion as an extra input, we add another fully-connected layer with 1024 nodes before the last layer. This extra layer adds representational capacity to account for the additional grasp information and increases network performance by 4% as shown in section VII-C. VGQ-CNN has a total of 6.5m and GQ-CNN has a total of 5.4m trainable parameters. The full architecture of VGQ-CNN can be seen in Fig. 3.

VI. Fast-VGQ-CNN

While discriminative grasp proposal methods have several advantages, one of their main disadvantages is that there is typically a high latency for proposing grasps, as explained in section II. For time-critical applications, we propose an alternative network architecture, Fast-VGQ-CNN, that can decrease latency when evaluating multiple grasps on a given object.

Instead of cropping the depth images such that the grasp centre is centred in the image, we decouple the grasp centre from the image cropping. To do this we make the u- and v-coordinates of the grasp centre within the image explicit inputs to the network. We apply random cropping [30] of the image around the grasp centre and specify the full 6-DOF pose input with $g \in (u, v, z, q)$ as an input to the pose stream of Fast-VGQ-CNN.

Applying random cropping to the image ensures grasp- and object placement variety within the image during training, resulting in a network which can evaluate grasp proposals for varying grasp locations within the image. The coordinates of the grasp centre in the cropped and resized image (u, v) are sampled uniformly. The range over which these coordinates are uniformly sampled, and therefore the maximum distance $\kappa = max(|u|, |v|)$ between the grasp centre and the image centre, influences the performance of Fast-VGQ-CNN. This is shown in the ablation studies in section VII-C.

By introducing the new grasp representation, as well as the randomised grasp image coordinates (u, v) during training, we can split the image stream with the high-dimensional convolutional layers from the rest of the network during inference. The image stream can be used as a shared encoder for multiple grasps, processing the image once and using the output for a batch of k grasp poses. Utilising shared network layers to reduce run-time has been proposed and used for other network structures, e.g. for various object detection networks [31].

The new architecture also changes the requirements for image preprocessing starting from a single 300×300 pixel image and a variable number of grasp poses k, as indicated in Fig 4. For each of the k grasps in VGQ-CNN, the image needs to be centred, cropped and resized, resulting in k centred 32×32 pixel depth images and k grasp poses $g \in (z, q)$. In contrast, for Fast-VGQ-CNN, the original image needs to be cropped and resized just once. The grasp centre coordinates (u, v) of each of the k grasp poses are determined relative to the single image. Therefore the network input comprises a single 32×32 depth image and k grasp poses $g \in (u, v, z, q)$. In both preprocessing pipelines, the image and pose values are normalised, excluding the quaternion.

For prediction, the resulting images and poses are fed to the network. For VGQ-CNN, the k depth images $\mathcal{I}$ and k grasp poses go into a single, multi-stream network. For Fast-VGQ-CNN, the single, depth image $\mathcal{I}$ goes into the shared encoder while the resulting tensor in combination with the k grasp poses are fed to the remaining fully connected layers of the network.

VII. Experiments

We train both VGQ-CNN and GQ-CNN for 150 million training iterations on VG-dset and DexNet2.0, respectively. One training epoch on a DexNet2.0-sized dataset is equivalent to approximately 6 million training iterations. The training takes approximately 23 hours to complete on one NVIDIA RTX 2060 GPU. We use a stochastic gradient decent optimiser with a momentum rate of 0.9, a base learning rate of 0.001 decaying every 4 million iterations by 0.95, a $L2$-regularisation of 0.0005 and sparse categorical cross-entropy loss.

In standard fashion, the depth images and z values are normalised before being fed into the network. The values of the quaternion q are not normalised, since they range within $-1, 1$

TABLE II
RESULTS

Network	Evaluation dataset	TPR	TNR	Balanced accuracy
GQ-CNN	DexNet2.0	70.0%	89.1%	79.5%
VGQ-CNN	VG-dset	73.9%	90.2%	82.1%
GQ-CNN	TG-Tset	63.6%	89.6%	76.7%
VGQ-CNN	TG-Tset	64.6%	85.7%	75.2%

naturally. We check performance on the validation split every 1 million iterations. For evaluating the resulting networks we use the balanced accuracy metric, which is calculated as the mean of true positive rate (TPR) and true negative rate (TNR). We motivate the use of the balanced accuracy metric due to the high imbalance of positive and negative grasps, with a network classifying all grasps as negative achieving an accuracy of 81% due to that imbalance. We choose the best model according to the balanced accuracy metric of the validation results for all our evaluations.

We investigate the performance of VGQ-CNN on a test split of VG-dset and compare performance between GQ-CNN and VGQ-CNN on a dedicated, DexNet2.0-like test set. In addition, we show the robustness of VGQ-CNN to varying camera poses, proving that VGQ-CNN does not need retraining when moving the camera within the range of the parameters specified in Table I. In a set of ablation studies, we show the effects of dataset size and the extra fully-connected layer on VGQ-CNN, as well as the effects of the range of randomised grasp image coordinates (u, v) on Fast-VGQ-CNN. Finally, we demonstrate the speed up that can be achieved by using a shared image encoder in Fast-VGQ-CNN.

A. Overall performance

For measuring the overall performance, we evaluate VGQ-CNN on the test split of VG-dset (700K grasps), and GQ-CNN on the test split of DexNet2.0 (700K grasps). In addition, to compare performance of VGQ-CNN and GQ-CNN, we evaluate them on a separate, new dataset with 300K grasps named Top Grasp Testset (TG-Tset). The objects in TG-Tset have not been seen by VGQ-CNN or GQ-CNN before, since they belong to the test split used in DexNet2.0 and VG-dset. TG-Tset is rendered using the grasp and camera parameters of DexNet2.0 as detailed in Table I.

Due to the different input requirements in the networks, images are centred and rotated before being fed to GQ-CNN and only centred for use in VGQ-CNN. Note that the collision checking for VG-dset differs slightly from the approach used for DexNet2.0. In DexNet2.0, grasps are classified as collision free if any of four linear approaches within $\pm 10°$ around ω are collision free. In VG-dset, grasps are classified as collision free only if a linear approach along the z-axis is collision free. For the purpose of comparison between GQ-CNN and VGQ-CNN, we exclude grasps from TG-Tset which differ in outcome of the collision checking by these two methods, corresponding to 0.3% of all generated grasps.

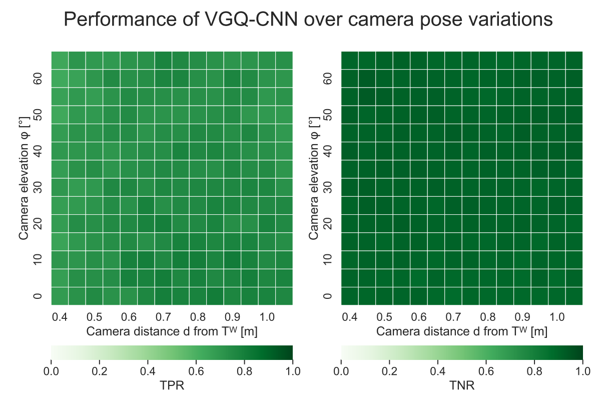

Fig. 5. TPR (left) and TNR (right) for VGQ-CNN, tested on an object-wise test split of VG-dset.

We calculate the true positive rate (TPR), true negative rate (TNR) and balanced accuracy. The results can be seen in Table II. Note that the balanced accuracy results for GQ-CNN are lower than the accuracy reported in [1], since they used different training parameters and report accuracy rather than balanced accuracy.

VGQ-CNN exhibits robust performance on VG-dset with a balanced accuracy of 82.1%, while being able to generalise to a wide range of camera poses and 6-DOF grasp orientations. On the much restricted range of camera poses and grasp orientations in TG-Tset, VGQ-CNN achieves competitive performance with a balanced accuracy of 75.2% compared to GQ-CNN's 76.7%. Note that the performance of Fast-VGQ-CNN depends on κ, as shown in section VII-C.

B. Performance over the camera parameter space

In addition to VGQ-CNN producing good results in VG-dset and being comparable to GQ-CNN on TG-Tset, it generalises well to varying camera positions. We report the TPR and TNR over the spherical coordinate variables of the camera, d and φ, to show the applicability of VGQ-CNN over varying camera poses in Fig. 5. Over the full target range of camera poses with steps of $\Delta d = 0.05m$ and $\Delta\varphi = 5°$, VGQ-CNN attains $TNR = 90.2\% \pm 0.9\%$ and $TPR = 73.9\% \pm 3.5\%$.

This robustness of VGQ-CNN in terms of camera poses shows that it does not require retraining for new camera poses within its target range. Hence, VGQ-CNN can be used for grasp quality prediction when moving camera poses and can even be used with mobile manipulators.

C. Ablation studies

For further insight into the network performance, we conduct a set of ablation studies on the effect of the dataset size on VGQ-CNN, the extra fully-connected layer in VGQ-CNN and the randomised grasp image coordinates (u, v) used for training our alternative network architecture, Fast-VGQ-CNN. Note that all networks are trained from scratch for 10 million training iterations. Each network configuration was trained 8 times with the shading in Fig. 6 corresponding to

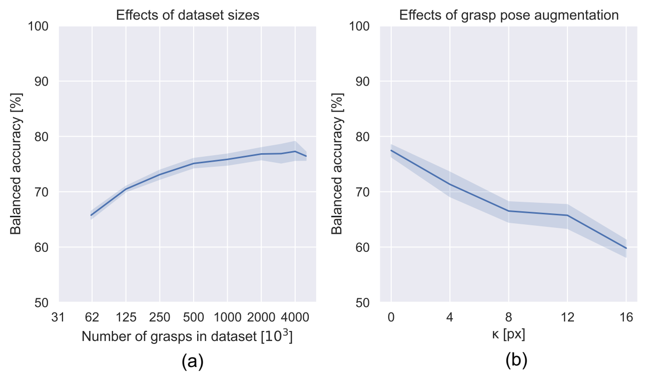

Fig. 6. Influence of the dataset size (a) and maximum grasp distance $\kappa = max(|u|, |v|)$ to image centre (b) on network performance.

the 95% confidence interval over the results. The datasets for the experiments of the dataset size influence were randomly undersampled from VG-dset. While the training and validation data differs for all networks in the dataset size experiments, they were tested on the same test split of VG-dset to ensure comparability. The base dataset for the experiments regarding the grasp image coordinates is VG-dset, with different κ applied to the same train-validation-test split.

Fig. 6 (a) shows how the number of grasp training samples affects the trained network performance. The minimum number of training samples for reasonable network performance is relevant especially when evaluating different sampling strategies in section IV-B, as some of these result in fewer training samples being available. When training the network with 1 million or fewer grasps, the network shows signs of overfitting and accuracy on the test set drops. Training VGQ-CNN with 2 million grasps or more shows relatively stable results with a mean balanced accuracy of 77%. Note that a balanced accuracy of 50% can be achieved by classifying all grasps as positive or negative.

Adding a second fully-connected layer with 1024 nodes before the last layer, as described in section V, increases balanced accuracy from $72.5\% \pm 1.8\%$ to $76.7\% \pm 1.1\%$.

When using Fast-VGQ-CNN, the grasp image coordinates are given as an explicit input to the network with (u, v) and $\kappa = max(|u|, |v|)$ describes the maximum displacement of the grasp centre within the image. $\kappa = 0px$ corresponds to the data used for VGQ-CNN since each grasp is centred in its image. Fig. 6 (b) exhibits a reduction in balanced accuracy as the displacement of the grasp centre in the image increases. For $\kappa = 0px$, Fast-VGQ-CNN reaches a balanced accuracy of 77% which drops to 60% for grasps uniformly distributed over the full image with $\kappa = 16px$.

D. Fast-VGQ-CNN

In section VI we propose speeding up inference by using a shared encoder with our modified network architecture Fast-VGQ-CNN. We measure the inference time of VGQ-CNN and Fast-VGQ-CNN on a machine with a 2.6 GHz Intel Core i7-10750 CPU and a NVIDIA GeForce RTX 2060 GPU. Tensorflow 1.15 is used for all experiments. We run the tests with differing numbers of grasps, $k \in \{32, 64, 96, 128\}$, as batch sizes and measure inference as the mean over 1000 runs. All image preprocessing steps are implemented with the pillow library in python. The resulting inference times for preprocessing and prediction can be seen in Table III.

TABLE III
INFERENCE TIME [MS]

	Prediction			
k	32	64	96	128
CPU VGQ-CNN	41	72	103	138
CPU Fast-VGQ-CNN	**11**	**11**	**12**	**12**
GPU VGQ-CNN	8	12	17	21
GPU Fast-VGQ-CNN	8	**8**	**8**	**8**
	Preprocessing			
VGQ-CNN	7	14	21	28
Fast-VGQ-CNN	**0.3**	**0.5**	**0.6**	**0.8**

The speed up of Fast-VGQ-CNN relative to VGQ-CNN increases as we increase the number of grasps predicted per batch, k. The Fast-VGQ-CNN preprocessing pipeline is up to 35 times faster than the preprocessing for VGQ-CNN, taking just $0.8ms$ to preprocess 128 grasps.

Of special interest when working with devices without a specialised GPU, Fast-VGQ-CNN can predict 128 grasps within $12ms$, while VGQ-CNN shows a significant increase for the prediction of more grasps taking $138ms$ for predicting 128 grasps on a CPU. In combination with an efficient grasp sampling strategy, Fast-VGQ-CNN could enable real-time grasping.

VIII. Discussion & future work

In this work we present VGQ-CNN, a network for predicting the quality of 6-DOF grasps as observed from a wide range of camera poses. Removing limitations for grasp orientation and camera pose in previous methods [1]–[6] and available grasp datasets [1], [9], [13] allows for VGQ-CNN to be used with 6-DOF grasps and a wide range of camera poses without the need to retrain the network. This is especially useful when working with mobile manipulators, which constantly change the relationship between the camera and object.

VGQ-CNN achieves a balanced accuracy of 82.1% on a test split of VG-dset while being able to generalise to camera poses within $2.1m^3$ above the planar surface. Further, we make a non-sampled version of our dataset VG-dset available for public use to train alternative versions of VGQ-CNN, e.g. focusing on special camera or grasp configurations. By using a shared encoder with our alternative network architecture Fast-VGQ-CNN, we can predict 128 grasps within $12ms$ using a CPU compared to $138ms$ with VGQ-CNN. Although our ablation studies in section VII-C show that there is a trade-off between network performance and the number of grasps that can be presented in a single image (which is related to κ), this speed up could be of particular interest when working with edge devices without dedicated GPUs.

In common with other approaches described in the literature, our approach has some limitations. One of these is the position of the objects on a table below the camera. While this is suitable for objects placed on normal tables, it does not reflect situations where the camera is level with or below the object such as when a mobile manipulator should fetch an object from a shelf. The current network also excludes objects being placed very close to the edge where the table edge would be visible in the depth image.

As VGQ-CNN is a grasp quality predictor, we aim to incorporate it with a 6-DOF grasp sampling technique to generate a full grasp proposal pipeline in future work. We then plan to apply this full pipeline to a mobile manipulator, e.g. the PAL TIAGo robot [25], and test performance on a grasping benchmark dataset such as EGAD [32]. Of special interest here is the comparison to point-cloud based 6-DOF grasp proposal systems like GPD [17], with differences in run-time and performance being crucial indicators of their usage for real-world applications.

References

[1] J. Mahler, J. Liang, S. Niyaz, M. Laskey, R. Doan, X. Liu, J. A. Ojea, and K. Goldberg, "Dex-net 2.0: Deep learning to plan robust grasps with synthetic point clouds and analytic grasp metrics," in *Robotics: Science and Systems (RSS)*, 2017.

[2] S. Levine, P. Pastor, A. Krizhevsky, J. Ibarz, and D. Quillen, "Learning hand-eye coordination for robotic grasping with deep learning and large-scale data collection," *The International Journal of Robotics Research*, vol. 37, no. 4-5, pp. 421–436, 2018. [Online]. Available: https://doi.org/10.1177/0278364917710318

[3] J. Zhang, M. Li, Y. Feng, and C. Yang, "Robotic grasp detection based on image processing and random forest," *Multimedia Tools and Applications*, vol. 79, 01 2020.

[4] J. Redmon and A. Angelova, "Real-time grasp detection using convolutional neural networks," in *2015 IEEE International Conference on Robotics and Automation (ICRA)*, 2015, pp. 1316–1322.

[5] S. Kumra, S. Joshi, and F. Sahin, "Antipodal robotic grasping using generative residual convolutional neural network," in *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2020, pp. 9626–9633.

[6] Y. Song, L. Gao, X. Li, and W. Shen, "A novel robotic grasp detection method based on region proposal networks," *Robotics and Computer-Integrated Manufacturing*, vol. 65, p. 101963, 2020. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0736584519308105

[7] V. Satish, J. Mahler, and K. Goldberg, "On-policy dataset synthesis for learning robot grasping policies using fully convolutional deep networks," *IEEE Robotics and Automation Letters*, 2019.

[8] D. Morrison, P. Corke, and J. Leitner, "Learning robust, real-time, reactive robotic grasping," *The International Journal of Robotics Research*, vol. 39, no. 2-3, pp. 183–201, 2019. [Online]. Available: https://doi.org/10.1177/0278364919859066

[9] A. Depierre, E. Dellandréa, and L. Chen, "Jacquard: A large scale dataset for robotic grasp detection," in *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2018, pp. 3511–3516.

[10] J. Mahler and K. Goldberg, "Learning deep policies for robot bin picking by simulating robust grasping sequences," in *Proceedings of the 1st Annual Conference on Robot Learning*, ser. Proceedings of Machine Learning Research, S. Levine, V. Vanhoucke, and K. Goldberg, Eds., vol. 78. PMLR, 13–15 Nov 2017, pp. 515–524.

[11] J. Mahler, M. Matl, X. Liu, A. Li, D. Gealy, and K. Goldberg, "Dex-net 3.0: Computing robust robot suction grasp targets in point clouds using a new analytic model and deep learning," 2018, pp. 5620–5627.

[12] J. Mahler, M. Matl, V. Satish, M. Danielczuk, B. DeRose, S. McKinley, and K. Goldberg, "Learning ambidextrous robot grasping policies," *Science Robotics*, vol. 4, no. 26, 2019. [Online]. Available: https://robotics.sciencemag.org/content/4/26/eaau4984

[13] I. Lenz, H. Lee, and A. Saxena, "Deep learning for detecting robotic grasps," *The International Journal of Robotics Research*, vol. 34, no. 4-5, pp. 705–724, 2015. [Online]. Available: https://doi.org/10.1177/0278364914549607

[14] J. Bohg, A. Morales, T. Asfour, and D. Kragic, "Data-driven grasp synthesis—a survey," *IEEE Transactions on Robotics*, vol. 30, no. 2, pp. 289–309, April 2014.

[15] K. Kleeberger, R. Bormann, W. Kraus, and M. F. Huber, "A survey on learning-based robotic grasping," *Current Robotics Reports*, pp. 1–11, 2020.

[16] M. Breyer, J. J. Chung, L. Ott, S. Roland, and N. Juan, "Volumetric grasping network: Real-time 6 dof grasp detection in clutter," in *Conference on Robot Learning*, 2020.

[17] A. ten Pas, M. Gualtieri, K. Saenko, and R. Platt, "Grasp pose detection in point clouds," *The International Journal of Robotics Research*, vol. 36, no. 13-14, pp. 1455–1473, 2017. [Online]. Available: https://doi.org/10.1177/0278364917735594

[18] A. Mousavian, C. Eppner, and D. Fox, "6-dof graspnet: Variational grasp generation for object manipulation," in *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, October 2019.

[19] J. Lundell, F. Verdoja, and V. Kyrki, "Beyond top-grasps through scene completion," in *2020 IEEE International Conference on Robotics and Automation (ICRA)*, 2020, pp. 545–551.

[20] J. Varley, C. DeChant, A. Richardson, J. Ruales, and P. Allen, "Shape completion enabled robotic grasping," in *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2017, pp. 2442–2447.

[21] L. Berscheid, C. Friedrich, and T. Kröger, "Robot learning of 6 dof grasping using model-based adaptive primitives," in *2021 IEEE International Conference on Robotics and Automation (ICRA)*, 2021, pp. 4474–4480.

[22] U. Viereck, A. Pas, K. Saenko, and R. Platt, "Learning a visuomotor controller for real world robotic grasping using simulated depth images," in *Conference on Robot Learning*. PMLR, 2017, pp. 291–300.

[23] S. Song, A. Zeng, J. Lee, and T. Funkhouser, "Grasping in the wild: Learning 6dof closed-loop grasping from low-cost demonstrations," *Robotics and Automation Letters*, 2020.

[24] X. Zhu, L. Sun, Y. Fan, and M. Tomizuka, "6-dof contrastive grasp proposal network," in *2021 IEEE International Conference on Robotics and Automation (ICRA)*, 2021, pp. 6371–6377.

[25] J. Pages, L. Marchionni, and F. Ferro, "Tiago: the modular robot that adapts to different research needs," in *International workshop on robot modularity, IROS*, 2016.

[26] T. Yamamoto, T. Nishino, H. Kajima, M. Ohta, and K. Ikeda, "Human support robot (hsr)," in *ACM SIGGRAPH 2018 Emerging Technologies*, ser. SIGGRAPH '18. New York, NY, USA: Association for Computing Machinery, 2018. [Online]. Available: https://doi.org/10.1145/3214907.3233972

[27] A. Kasper, Z. Xue, and R. Dillmann, "The kit object models database: An object model database for object recognition, localization and manipulation in service robotics," *The International Journal of Robotics Research*, vol. 31, no. 8, pp. 927–934, 2012. [Online]. Available: https://doi.org/10.1177/0278364912445831

[28] W. Wohlkinger, A. Aldoma, R. B. Rusu, and M. Vincze, "3dnet: Large-scale object class recognition from cad models," in *2012 IEEE International Conference on Robotics and Automation*, 2012, pp. 5384–5391.

[29] D. Seita, F. T. Pokorny, J. Mahler, D. Kragic, M. Franklin, J. Canny, and K. Goldberg, "Large-scale supervised learning of the grasp robustness of surface patch pairs," in *2016 IEEE International Conference on Simulation, Modeling, and Programming for Autonomous Robots (SIMPAR)*, 2016, pp. 216–223.

[30] C. Shorten and T. M. Khoshgoftaar, "A survey on image data augmentation for deep learning," *Journal of Big Data*, vol. 6, no. 1, pp. 1–48, 2019.

[31] Z.-Q. Zhao, P. Zheng, S.-t. Xu, and X. Wu, "Object detection with deep learning: A review," *IEEE transactions on neural networks and learning systems*, vol. 30, no. 11, pp. 3212–3232, 2019.

[32] D. Morrison, P. Corke, and J. Leitner, "Egad! an evolved grasping analysis dataset for diversity and reproducibility in robotic manipulation," *IEEE Robotics and Automation Letters*, vol. 5, no. 3, pp. 4368–4375, 2020.