Grasping Virtual Fish: A Step Towards Robotic Deep Learning from Demonstration in Virtual Reality

Jonatan S. Dyrstad[1] and John Reidar Mathiassen[2]

Abstract— **We present an approach to robotic deep learning from demonstration in virtual reality, which combines a deep 3D convolutional neural network, for grasp detection from 3D point clouds, with domain randomization to generate a large training data set. The use of virtual reality (VR) enables robot learning from demonstration in a virtual environment. In this environment, a human user can easily and intuitively demonstrate examples of how to grasp an object, such as a fish. From a few dozen of these demonstrations, we use domain randomization to generate a large synthetic training data set consisting of 76 000 example grasps of fish. After training the network using this data set, the network is able to guide a gripper to grasp virtual fish with good success rates. Our domain randomization approach is a step towards an efficient way to perform robotic deep learning from demonstration in virtual reality.**

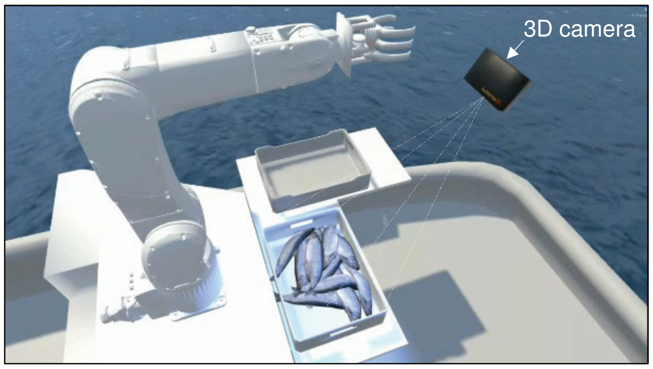

Fig. 1. The virtual robot is tasked with grasping virtual fish from a box and placing them in another box. The sensory input to the robot consists of a virtual 3D camera that generates depth images of the fish in the box.

I. INTRODUCTION

In robotics, robust grasping and manipulation of objects is still a challenging task to automate, in particular for biological and deformable objects such as fish. Inspired by the ability of humans to perform such tasks, we investigate how to efficiently transfer the knowledge of a human to the robot, via a combination of 1) a virtual reality (VR) interface for demonstrating the task, 2) domain randomization over components of the task to generate a large data set, and 3) deep learning on this large data set. Our hypothesis is that through our approach, VR can serve as a efficient medium for demonstrating complex tasks to robots. A first step towards testing this hypothesis is presented in this paper. We describe an approach for deep learning from demonstration in VR, where a human can easily teach a robot how to grasp fish. Grasping of fish from a box is an example of a challenging grasping task involving a cluttered scene of multiple highly deformable objects. In today's fishing industry, many simple and repetitive tasks are still performed by human workers due to difficulties in automating handling of the fish. For tasks with a large throughput, there exist processing machinery that moves fish and handles them automatically. Compared to a robotic solution, these systems are neither compact nor adaptive. The fish picking task is therefore ideal for demonstrating the capabilities of our approach applied to relevant industrial problems.

An essential aspect of learning from demonstration is the system in which the human teacher demonstrates the task. This system should be intuitive and easy to use and additionally, the teacher should not have to demonstrate the task many times, which is often necessary when training deep learning models. Therefore, we have developed a system where the teacher's actions are not used directly to train the robot, but rather used to generate large amounts of synthetic training data through domain randomization over relevant components of the grasping task. This enables us to train a deep neural network (DNN) for grasp detection, from scratch, using only a few dozen manually-demonstrated example grasps. In this paper we use our approach to robotic deep learning from demonstration to create a grasp detector with success rates that are sufficient to enable improvements in future research directions. These directions include using reinforcement learning algorithms to rapidly improve the performance of the system in VR and later in a real world scenario.

In this paper, a grasp is defined by the 6 degrees of freedom (DOF) needed to define a robotic gripper in 3D-space. This is a simplification and successful grasps in a real world situation might require good path planning and a well regulated and agile gripper, capable of holding objects with different weights, textures and sizes.

Both training and testing of our system has been done on synthetic data, since this enables efficient development and testing of our approach. Future work will focus on the transferability of the features learned on the synthetic data to data from the real world.

Robot grasping is a field with a lot of research being done. There are several methodologies applicable to robot grasping based on visual input, such as learning from demonstration

[1]Jonatan S. Dyrstad is with the Processing department, SINTEF Ocean AS, Brattrkaia 17c, 7010 Trondheim, Norway `jonatan.dyrstad@sintef.no`

[2]John Reidar Mathiassen is with the Processing department, SINTEF Ocean AS, Brattrkaia 17c, 7010 Trondheim, Norway `john.reidar.mathiassen@sintef.no`

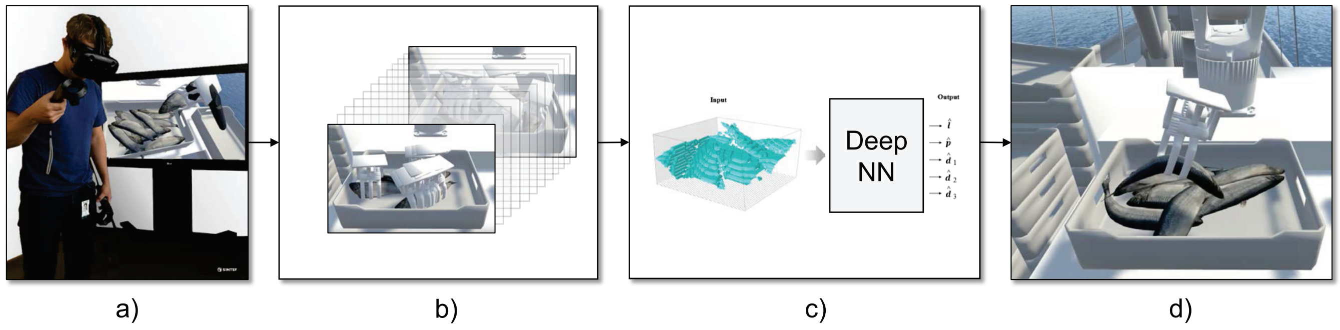

Fig. 2. The presented approach to deep learning from demonstration in VR, where a) the user provides a few example grasps by grasping fish from the box, b) domain randomization is used to generate a large data set based on the few example grasps, c) the deep neural network is trained on the large data set, and d) the trained deep neural network controls a gripper to grasp virtual fish.

(LfD) and reinforcement learning, based on real or virtual data. Learning from demonstration is a generic approach [1], [2], [3] in which a human or virtual teacher demonstrates a task, e.g. a grasping action specified by a pose and gripper configuration, with a corresponding state space consisting of visual information. Based on a set of demonstrations, a learning algorithm, such as support vector machines (SVM) and regularized regression [4] or artificial neural networks [5], [6], [7], [8], [9], [10], learns the mapping between the visual state and the grasping action. In reinforcement learning (RL) there is no teacher to demonstrate how to perform the task, instead there is a reinforcement signal provided to the learning algorithm. LfD has the advantage of efficiently discovering state-action mappings that work reasonably, with the disadvantage that this may require a large data set of demonstration examples. Contrary to this, RL, e.g. using artificial neural networks [11], has the disadvantages of slow learning speed and difficulty of accurately defining a suitable objective function for more complex tasks. The advantages of RL are that they do not require a human teacher, and they are capable of exploring and learning how to perform potentially beyond the capabilities of a teacher.

Synthetic data generation is a well-known and successful approach to training deep learning systems [13], [14], prior to applying them to real-world data. In particular, an approach called domain randomization has been shown to enable robust training on simulated data that directly works in the real-world without additional training [15]. Domain randomization involves randomizing over the relevant components of a task. A novelty of our approach for domain randomization is that it begins with an intuitive virtual reality interface within which a human teacher can easily provide *a low or moderate number* of demonstration examples. These examples are then used in a domain randomization process over the camera viewpoints, the physical interactions of the fish and the box, and the previously demonstrated intent of the teacher, to generate the large data sets required for deep learning.

To the best of our knowledge, we present the first work where a deep learning algorithm has solved robotic grasping using a 3DCNN, which has previously been successfully applied to 3D shape recognition [4], [12]. Our main motivation in working with voxel grid representations of point clouds, and 3DCNNs to analyze them, is to develop the foundation for deep learning in robotics applications that are camera- and viewpoint-agnostic. Hence the 3DCNN can work in a single voxel image including 3D information obtained from multiple depth images from multiple cameras and/or viewpoints. Multiple viewpoints and cameras can provide a more complete coverage of a scene. A single-view depth image will have occluded regions, and moving the depth camera to one or more other locations will provide a better view of the occluded regions. Fusing these two views into a single point cloud will provide a more complete view. The advantage of the 3DCNN approach is thus that can be invariant or agnostic to the number of views or the number of cameras that generate the 3D data, and it can work on the complete view, as long as domain randomization is done over the types of views that are possible.

Our main contributions are; 1) a novel domain randomization approach and its application to learning from demonstration in virtual reality, 2) using a deep 3D convolutional neural network (3DCNN) to detect potential grasps and estimate their pose. We develop and test our contributions in a virtual reality environment in this paper, as preparation for future research focusing on development and test in the real world.

II. METHOD

We use a deep 3DCNN to estimate grasps from a point cloud. We propose the use of VR to generate large amounts of synthetic training data in order to be able to train a deep learning model with many parameters. An illustration of our approach is shown in Fig. 2.

A. *User interface for demonstrating the task*

A virtual environment was created where a user, using a virtual reality head mounted display (HMD) and tracked hand controllers, can enter and demonstrate the task for the robot as shown in Fig. 2a. The user has a controller in his hand, which in VR appears to him as a gripper, similar to the end effector on the robot. The environment was created with the Unity game engine and the VR-equipment used the HTC-Vive head-mounted display and hand-held motion controllers.

Fish are instantiated in mid-air and dropped using simulated physics that model the deformation and friction characteristics of the pelagic fish species herring (*Clupea harengus*). This ensures that the fish land in natural poses in a fish box placed in front of the robot. The user's task is to grasp the fish and move the fish from this box to another box, using the gripper in his hand (see Fig. 2a). In this way, the user gets the impression that he is *showing* the robot how to perform the task, in an easy and intuitive way. Since the user is told to grasp the fish in a way that enables him to pick it up and place it in a second box, he will naturally use a grasp that is suited for that task. If e.g. the task had been a different one, such as placing the fish in a narrow hole, the user would probably grasp the fish differently. Hence, this is an effective way of getting the user to demonstrate grasps that are suitable for a given task. For each fish grasped by the user, the grasp is logged with regards to its position and orientation relative to the fish. An example of these logged grasps can be seen in Fig. 4. This is the demonstration part of our learning from demonstration (LfD) approach. In traditional LfD, a large number of demonstration examples are required. The number of required examples scales with the number of parameters in the model that is being taught. Models with very many parameters, such as large neural networks, may require on the order of tens of thousands of demonstrations in order to adequately learn the task without overfitting to the training data. For a user this is clearly too much work, and an alternative approach is needed to generate a sufficiently large and realistic training data set.

B. Generating Large Amounts of Synthetic Data by Domain Randomization

Based on the logged grasps, we propose to use domain randomization that includes information on the user's grasp intent, as a method for generating a large training data set from a few demonstrations, with no further human supervision. As in the previous section, the fish are instantiated and dropped into the fish box, as shown in Fig. 3a. By randomizing over the number of fish, and the position and orientation of each fish before dropping them into the box, this provides domain randomization over the possible ways in which fish can realistically be positioned relative to each other in a box.

Instead of the user demonstrating the grasp for each randomly generated box of fish, we instantiate all of the previously logged grasps onto each of the fish in the box, as shown in Fig. 3b. However, not all of the grasps are valid for all of the fish, given their current pose and position in the box (i.e. closeness to the walls etc.). Therefore, for every fish, all of the logged grasps are automatically checked using collision heuristics to see if they collide with the environment or with the other objects in the scene in any way. The ones that do not are kept and the rest are discarded, resulting in a set of plausible grasps as shown in Fig. 3c. This three-step approach provides domain randomization over the possible ways a user would *probably* grasp the fish, based on what know from the previously logged grasps.

An orthographically projected depth image is rendered of the entire fish box and the list of valid grip vectors are recorded along with the depth image (an example is shown in Fig. 7). The field of view and resolution of the virtual 3D camera is such that each pixel in the orthographically projected depth image can be read as an xyz-coordinate in millimeters given in camera coordinates (with an offset of $\frac{imagewidth/height}{2}$ in the xy-direction). Current 3D cameras, such as the Microsoft Kinect and Intel RealSense, work by projecting a light pattern from a projector that is offset from the actual camera. Because some of the scene is visible to the camera but occluded to the projector, the result is *depth shadows*, areas in the depth image with unknown depth values. This effect is simulated with the virtual 3D camera as well. The depth data we generate in simulation can therefore be thought of as coming from a perfectly calibrated real 3D camera. To provide robust learning, we randomize the position and orientation of the virtual 3D camera as well, thus creating variations in the amount of missing data in the depth images due to the occlusion of the projector illumination. This is our final component of domain randomization.

C. Neural Networks

The depth images are projected into a voxel grid, and we use a 3DCNN to estimate grasps from a volume in the voxel grid, and split the problem up into three sub-problems

- Detecting possible grasp locations
- Finding the precise grip point
- Finding the orientation of the gripper

The architecture of the network is shown in Fig. 5. For a volume of size $50 \times 50 \times 25$, the output is a vector

$$\hat{\mathbf{y}} = \begin{bmatrix} \hat{l} & \hat{\mathbf{p}} & \hat{\mathbf{d_1}} & \hat{\mathbf{d_2}} & \hat{\mathbf{d_3}} \end{bmatrix}, \tag{1}$$

where $\hat{l} \in [0,1]$ estimates the probability of the input volume containing a valid grasp, $\hat{\mathbf{p}}$ estimates the precise position of a grasp within the input volume and $\hat{\mathbf{d_1}}$, $\hat{\mathbf{d_2}}$ and $\hat{\mathbf{d_3}}$ estimate the orientation of the grasp.

The network is *fully convolutional* and has sliding dense layers, meaning that the dimensions of the output from the network is dependent on the dimensions of the input volume. For larger inputs, the result is a grasp detector, rather than a classifier, capable of detecting multiple grasps within the input volume.

The estimations for the three sub problems are output from the same network and trained jointly because of the high dependence between the different objectives. The total cost for training example i is given by

$$J^{(i)} = \alpha J_C^{(i)} + l^{(i)}(\beta J_O^{(i)} + \gamma J_P^{(i)}), \tag{2}$$

where $J_C^{(i)}$ is the classification cost, $J_O^{(i)}$ the orientation cost and $J_P^{(i)}$ the position cost, $l^{(i)}$ is the true label for training example i and the parameters α, β and γ are simple weighing factors used to prioritize the relative importance of the

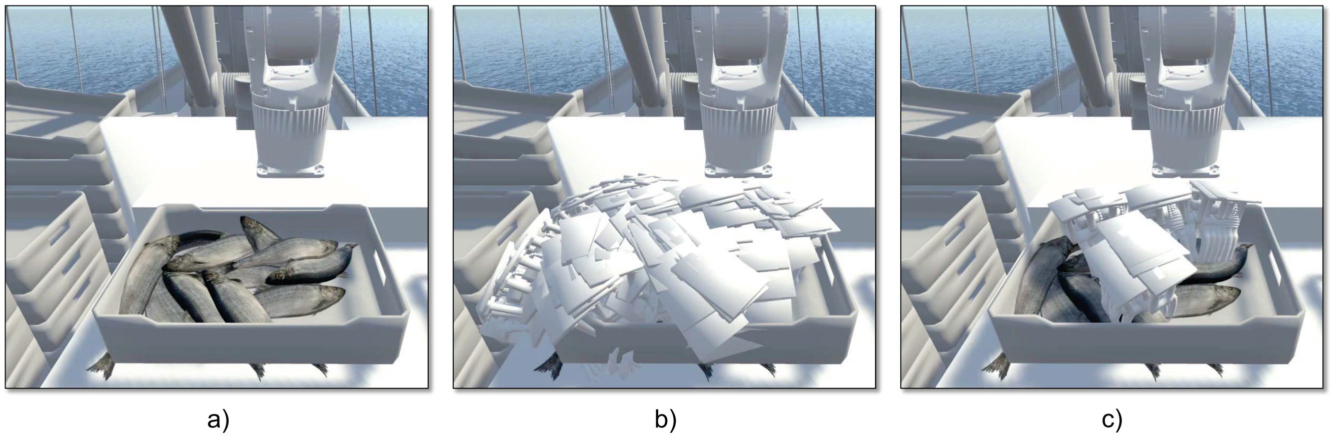

Fig. 3. The domain randomization approach for generating a large data set, by a) dropping a random number of fish in the box, using realistic fish physics, b) placing each of the logged grasps onto each fish, c) pruning the grasps based on collision heuristics.

different sub-problems. Note that for false examples, no updates are done to the position and orientation estimators. For classification of valid grasps, the binary cross entropy function

$$J_C^{(i)} = -l^{(i)} \log(\hat{l}^{(i)}) + (1 - l^{(i)}) \log(1 - \hat{l}^{(i)}) \tag{3}$$

is used, and for regression on the precise grasp point $\mathbf{p}^{(i)}$ within the given volume we use the squared error cost function

$$J_P = \frac{1}{2} ||\hat{\mathbf{p}}^{(i)} - \mathbf{p}^{(i)})||^2. \tag{4}$$

The orientation of the gripper is defined unambiguously with two three dimensional vectors, each describing a direction in 3D-space (see Fig. 6). However, for an object like a fish, which is almost a mirror image of itself along one axis, there are always two correct answers to any situation (i.e. an ”overhand” and an ”underhand” grip). To avoid any contradicting labels in the data set we assume a symmetrical gripper and can therefore simply reverse the sign of the vectors on one half of the data manifold. This leads to a discontinuity which we solve by having two estimates of one vector, each with the discontinuity at different places in the data manifold. In this way, one estimate becomes increasingly reliable as the other goes towards its uncertain region. Empirical observations of the training examples in the data set shows that one of the target vectors defining the grip orientation ($\mathbf{v_2}$ in Fig. 6) draws out a diffuse disk in 3D-space (as opposed to a sphere). This lets us get away with two estimates of the vector to avoid discontinuities ($\hat{\mathbf{d_1}}$ and $\hat{\mathbf{d_2}}$ in (1)). The variance in the other orientation vector ($\mathbf{v_1}$ in Fig. 6) is small (because most of the grasps are pointing up in camera coordinates) and therefore we only use one estimate for this vector ($\hat{\mathbf{d_3}}$ in (1)). The total orientation cost for the $N = 3$ orientation vectors is given by

$$J_O^{(i)} = \sum_{k=1}^{N} 1 - \frac{dot(\hat{\mathbf{d_k}}^{(i)}, \mathbf{d_k}^{(i)})}{\sqrt{dot(\hat{\mathbf{d_k}}^{(i)}, \hat{\mathbf{d_k}}^{(i)}) dot(\mathbf{d_k}^{(i)}, \mathbf{d_k}^{(i)})}}, \tag{5}$$

where, $\mathbf{d_k}^{(i)}$ and $\hat{\mathbf{d_k}}^{(i)}$ for $k \in 1,2,3$ respectively denote the true and estimated orientation vectors for training example i.

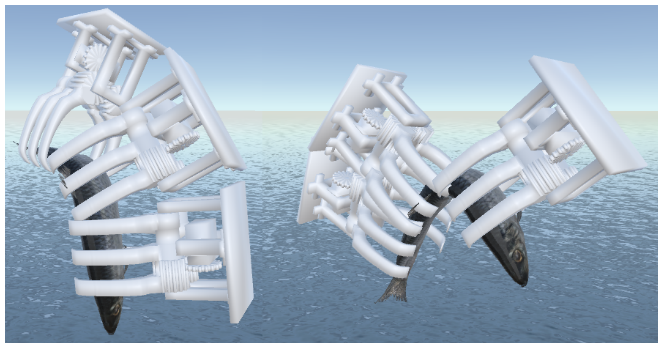

Fig. 4. The logged grasps after demonstration of three grasps in virtual reality. As the fish bends and twists, the grips follow, making them valid for the fish regardless of pose.

D. Preparing Data For Training

During training, the input to the network is a volume of size $50 \times 50 \times 25$ and the classification label $l^{(i)}$ is either 1 or 0 (i.e. the volume does or does not contain a valid grasp). Training examples with true labels are simply created by cropping volumes of the synthetically created depth images centered around one of the valid grasp points for that image. The crop is offset randomly from the middle by some amount in order to create training vectors for the grip point estimator as well. Generation of false training examples is more problematic. The virtual environment outputs a depth image and some, but not all conceivable grip vectors for the given image. Thus, there is no way of knowing which parts of the image that are guaranteed not to contain a valid grasp. In our experiments we generate false training examples (i.e. areas with a low probability of containing a grasp), simply by cropping random parts of the image and labelling them as false examples. Because the volumes that contain valid grasp are vastly outnumbered by the volumes that do not,

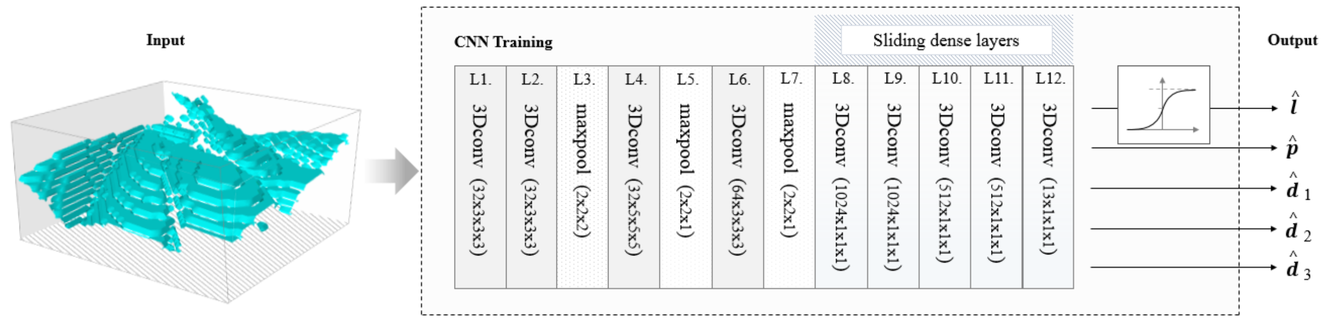

Fig. 5. The architecture of the 3DCNN consists of stacked convolutional and max pooling layers. The dense layers are swapped for $1 \times 1 \times 1$ convolutions to enable inputs of varying sizes. The activation functions for the feature extraction layers are rectified linear units, and in the top layer, $\hat{l}$ has a sigmoid activation function and the rest have linear activations.

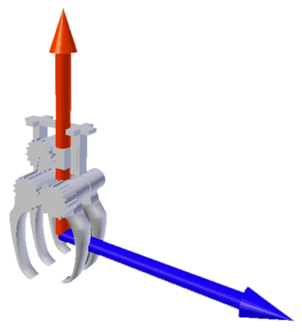

Fig. 6. The orientation of the gripper is defined by two vectors $\mathbf{v_1}$ (red) and $\mathbf{v_2}$ (blue).

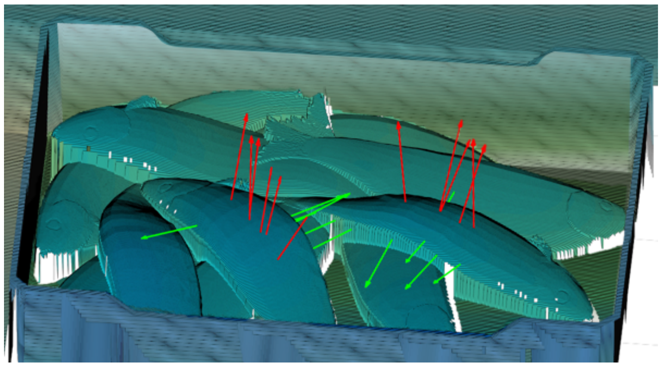

Fig. 7. An example of a synthetically generated depth image with the labeled grip vectors overlaid.

the result is a false-data set with mostly true, but also some false, negatives.

III. RESULTS

In our experiments, 43 grasps were shown in VR. With these grasps, 5,000 images were rendered of fish boxes containing between one and fifteen fish. Randomness was introduced in the rotation and position of the depth camera when the images were rendered. From these depth images, 76,000 training examples were cropped where 2/3 of the data set did not contain a grasp. The input to the network during testing was a volume of size $197 \times 197 \times 50$ and the the depth images were decimated so that the resolution was 3.6 mm per voxel.

Testing was done in real time with the trained model attempting to pick fish in the virtual environment as shown in Fig. 9. Because no paths are output from the network, the gripper simply approached the estimated grasp point from the direction given by $\mathbf{v_1}$ in Fig. 6. If the gripper could not reach the grip point because it hit the environment, or if it failed to capture and hold the fish when the gripper was closed, the grasp was considered unsuccessful. If several grasps were detected in the image, the one with the largest detection certainty, $\hat{l}$, was chosen.

The neural network was created in Python with the Theano[1] and Lasagne[2] libraries and training was done on an NVIDIA Titan X GPU.

A. Generated Training Examples

The logic for generation of large amounts of labelled training data with only a few shown grasps works well. Many examples were inspected visually and all of the inspected grasps in the training set look plausible. An example of a rendered image with labelled grasps is shown in Fig. 7. In this case, and in general, the labeled grips are good, but the virtual environment only outputs a subset of all possible grips in the scene.

The generated depth images with added noise and simulated depth shadows still looks like stylized and ideal depth data. Visual comparison (not included in this paper) to real data from the Microsoft Kinect and Intel RealSense depth cameras, reveals that specular effects are what separates the real from the synthetic data the most. The large difference in albedo on different parts of the fish leads to areas in the real depth images with undefined values which are not present in the synthetic ones.

B. Testing the Trained Neural Network

The neural network was trained with stochastic batch gradient decent with momentum, and cross-validated on a

[1]http://deeplearning.net/software/theano/
[2]https://lasagne.readthedocs.io/en/latest/

testing set with respect to minimizing the objective function. This function is however not an exact representation of grasping success, and our primary test involves testing the trained network in virtual grasping scenarios that were not seen during training of the neural network.

The trained network was used to attempt 100 grasps with 1, 5, 10 and 15 fish in the box and the results are shown in Tab. I. These results show that the performance is similar regardless of the number of fish in the box. Hence the neural network handles scenarios with many fish, in overlapping and challenging conditions, almost equally well as with only a single fish in the box.

Observations of the grasping attempts during live testing reveals that most unsuccessful grasps are a result of

- The gripper hitting the environment. For the most part with the top of the gripper hitting the walls and sometimes with the tip of the fingers hitting the bottom of the box.
- The fish sliding out of the gripper because the fish was picked to close to the head or tail.
- The chosen grip was a valid grip, but not the best candidate in the scene. Meaning, the local grip area was good, but other fish should have been picked first. E.g. the gripper sometimes push other fish around on its way to the grip point changing the state of the box before arrival.

For almost all fish boxes there are a lot of false positives. Even with no fish in the box, several valid grasps are often detected. However, when fish is present, the most certain grasps are often good. A typical result for a fish box with the 10 most certain grasps overlaid is shown in Fig. 8.

The detector, grasp point and grasp orientation estimates are good, and surprisingly good grasps can be found for almost vertical fish, frozen in mid air, even if this has never been seen in the data set used for training the neural network.

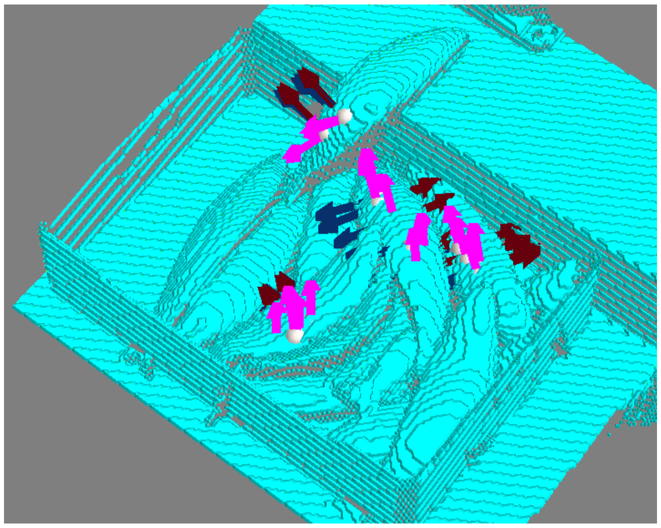

Fig. 8. The decimated volume input to the network, with the 10 most certain grasps overlaid. The pink arrow corresponds to $\mathbf{v_1}$ in Fig. 6, and the blue and red ones are the two estimates of $\mathbf{v_2}$ in Fig. 6.

TABLE I
GRASP SUCCESSES IN VR

No. of fish in the box	1	5	10	15
Success rate	70 %	74 %	61 %	71 %

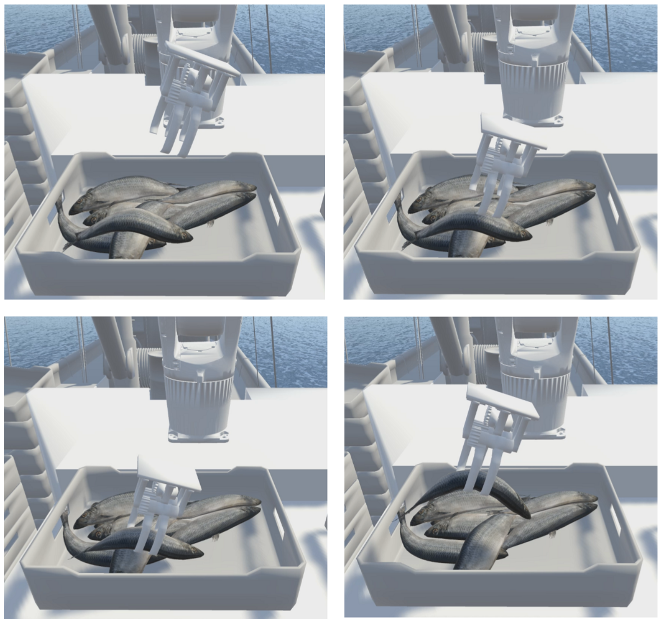

Fig. 9. The trained 3DCNN grasping a fish successfully in the virtual environment.

IV. DISCUSSION AND FUTURE WORK

The results of our work, suggest that our approach to domain randomization in virtual reality works intuitively and well. Based on only a few demonstration examples, a sufficiently large and diverse data set was generated that was capable of training a large 3D convolutional neural network. This network was tested on previously unseen scenarios, with success rates on the order of 70 %. This is good, but not sufficient for a working system. However, the results are good enough for us to proceed with implementation of this system on a real-world robot, and for refining the neural network using reinforcement learning in virtual reality as well as in the real world. One may argue that demonstrations in virtual reality do not prove the validity of our approach. Contrarily, we may argue that a methodology such as deep learning already has proven itself capable in transfer learning to the real world, as long as domain randomization is realistic [15]. As such, the domain randomization approach and neural network presented in this paper lay the groundwork for future implementation in a real-world scenario.

Our hypothesis from the onset is that through our approach, VR can serve as a efficient medium for demonstrating complex tasks to robots. A first step towards testing this hypothesis was presented in this paper, and the results are promising enough to maintain our hypothesis. However, more work is needed to develop this further. Future work will focus on transfer learning to apply what is learned in

VR to a real-world setting of picking fish from a box using an industrial robot. In addition to what we have presented in this paper, this may require better modelling of the depth images of fish. Fish are specularly reflective objects, and therefore affect the quality of depth images acquired with real 3D cameras such as the Microsoft Kinect or Intel RealSense. Accurate modeling of these reflective properties will enable more accurate training in virtual reality.

Beyond our application to grasping fish, we will also expand the domain randomization methodology and neural network architecture presented to the more generic problem of grasp detection for multiple types of objects and also to learning from demonstration of more complex task sequences.

ACKNOWLEDGMENTS

This project was funded by the Research Council of Norway, through its basic funding scheme, and in part from Research Council of Norway Grant No. 262900.

REFERENCES

[1] R. Pelossof, A. Miller, P. Allen and T. Jebara, ”An SVM learning approach to robotic grasping,” In Proceedings of the 2004 IEEE International Conference on Robotics and Automation (ICRA), 2004, Vol. 4, pp. 3512-3518.

[2] B. D. Argall, S. Chernova, M. Veloso and B. Browning, ”A survey of robot learning from demonstration,” Robotics and autonomous systems, 57(5), 469-483. May 2009.

[3] S. Choi, K. Lee and S. Oh, ”Robust learning from demonstration using leveraged Gaussian processes and sparse-constrained optimization,” In 2016 IEEE International Conference on Robotics and Automation (ICRA), 2016, pp. 470-475.

[4] S. Song and J. Xiao, ”Sliding shapes for 3d object detection in depth images,”. In European Conference on Computer Vision, 2014, pp. 634-651. Springer International Publishing.

[5] N. Rezzoug and P. Gorce, ”Robotic grasping: A generic neural network architecture”, 2006, INTECH Open Access Publisher.

[6] A. Saxena, J. Driemeyer and A. Y. Ng, ”Robotic grasping of novel objects using vision”. The International Journal of Robotics Research, 2008, 27(2), 157-173.

[7] Y. Jiang, S. Moseson and A. Saxena, ”Efficient grasping from rgbd images: Learning using a new rectangle representation,” In 2011 IEE International Conference on Robotics and Automation (ICRA), 2011, pp. 3304-3311.

[8] P. C. Huang, J. Lehman, A. K. Mok, R. Miikkulainen and L. Sentis, ”Grasping novel objects with a dexterous robotic hand through neuroevolution,” In 2014 IEEE Symposium on Computational Intelligence in Control and Automation (CICA), 2014, pp. 1-8.

[9] I. Lenz, H. Lee and A. Saxena, ”Deep learning for detecting robotic grasps,” The International Journal of Robotics Research, 2015, 34(4-5), 705-724.

[10] J. Dyrstad, ”Deep learning in virtual reality for grasp detection”, Master’s thesis, University of Stavanger, 2016.

[11] S. Levine, P. Pastor, A. Krizhevsky and D. Quillen, ”Learning Hand-Eye Coordination for Robotic Grasping with Deep Learning and Large-Scale Data Collection”, arXiv preprint arXiv:1603.02199. (2016)

[12] Z. Wu, S. Song, A. Khosla, F. Yu, L. Zhang, X. Tang and J. Xiao, ”3d shapenets: A deep representation for volumetric shapes,” In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2015, pp. 1912-1920.

[13] J. Shotton et al. ”Real-Time Human Pose Recognition in Parts from a Single Depth Image”. In CVPR. IEEE, (2011)

[14] E. Wood et al. ”Rendering of Eyes for Eye-Shape Registration and Gaze Estimation”. In CoRR abs/1505.05916 (2015)

[15] J. Tobin, R. Fong, A. Ray, J. Schneider, W. Zaremba and P. Abbeel, ”Domain Randomization for Transferring Deep Neural Networks from Simulation to the Real World”. arXiv preprint arXiv:1703.06907. (2017)