2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)
Madrid, Spain, October 1-5, 2018

Teaching a Robot to Grasp Real Fish by Imitation Learning from a Human Supervisor in Virtual Reality

Jonatan S. Dyrstad[1,2], Elling Ruud Øye[1], Annette Stahl[2] and John Reidar Mathiassen[1,*]

Abstract— **We teach a real robot to grasp real fish, by training a virtual robot exclusively in virtual reality. Our approach implements robot imitation learning from a human supervisor in virtual reality. A deep 3D convolutional neural network computes grasps from a 3D occupancy grid obtained from depth imaging at multiple viewpoints. In virtual reality, a human supervisor 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 100 000 example grasps of fish. Using this data set for training purposes, the network is able to guide a real robot and gripper to grasp real fish with good success rates. The newly proposed domain randomization approach constitutes the first step in how to efficiently perform robot imitation learning from a human supervisor in virtual reality in a way that transfers well to the real world.**

I. Introduction

In robotics, robust grasping and manipulation of objects is still a challenging task to automate, in particular for biological, non-rigid 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 synthetic data set, and 3) deep learning on this large data set. Our hypothesis is that through our approach, VR can serve as an efficient medium for demonstrating complex tasks to robots. A first step towards testing this hypothesis was presented in earlier work [16], where both training and testing was done entirely in VR. In this paper we take another step, testing this hypothesis, by demonstrating that a robot trained entirely in virtual reality can grasp real fish. We describe an approach where a human supervisor can easily teach a robot how to grasp fish in a VR environment. Grasping and picking 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. This task is therefore ideal for demonstrating the capabilities of our approach applied to relevant industrial problems.

An essential aspect of imitation learning, from a human supervisor, is the system in which the human demonstrates the task. This system should be intuitive and easy to use and additionally, the supervisor 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 supervisor'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.

Training of our system is done exclusively on synthetic data, since this enables efficient development and testing of our approach. Testing of the system is done on real data and with a real robot and gripper. This succeeds in our case, since domain randomization, over the possible poses and dimensions of the virtual fish and the parameters of the virtual 3D camera, ensures that a 3D CNN can perform well on real data encountered during tests with real fish.

A. Related work

Robot grasping is an active ongoing research field. There are several methodologies applicable to robot grasping based on visual input, such as imitation learning and reinforcement learning, based on real or synthetic data. Imitation learning is a generic approach [1], [2], [3] in which a human or algorithmic supervisor 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), regularized regression [4] or artificial neural networks [5], [6], [7], [8], [9], [10], learns to find or evaluate the mapping between the visual state and the grasping action. In reinforcement learning (RL) there is no supervisor to demonstrate how to perform the task, instead there is a reinforcement signal provided to the learning algorithm. Imitation learning 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 algorithms are that they do not require a human supervisor, and RL algorithms can potentially learn to perform beyond the capabilities of a supervisor.

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

*Corresponding author, `John.Reidar.Mathiassen@sintef.no`
[1]SINTEF Ocean AS, Trondheim, Norway
[2]NTNU, Department of Engineering Cybernetics, Trondheim, Norway

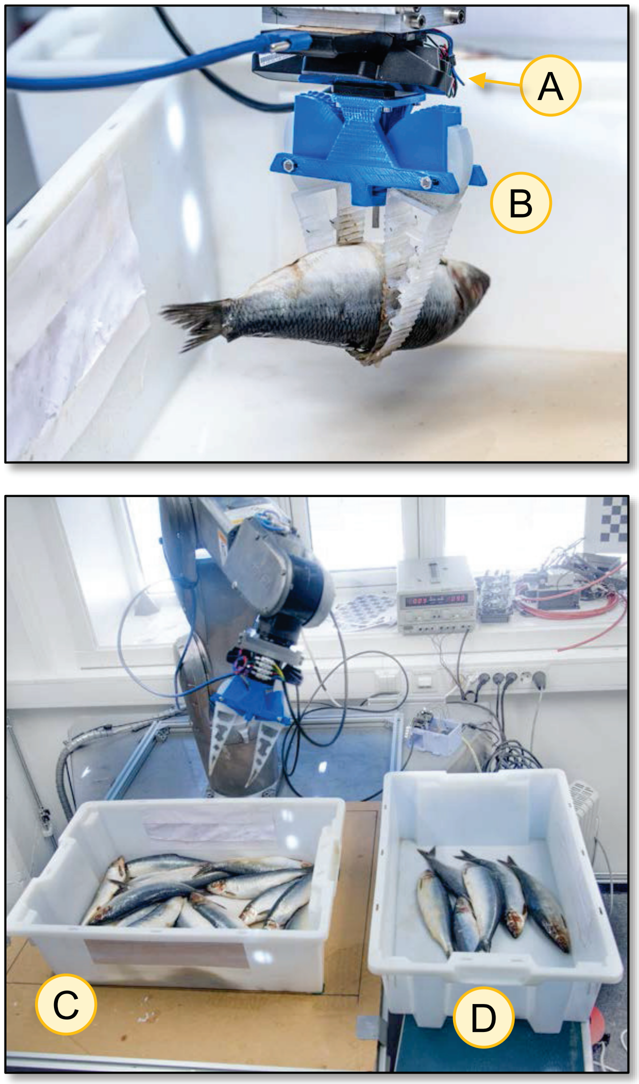

Fig. 1. A robot is tasked with picking fish from a box. The robot has a RealSense 3D camera (A) and gripper (B) mounted on its end effector. The task is to pick fish from a box (C) and place them in a second box (D).

called domain randomization has been shown to enable robust training on synthetic data that directly works on real-world data without additional training [15]. Domain randomization involves randomizing over the relevant components of a task to generate synthetic data that has enough variation to generalize to real data. A novelty of our approach for domain randomization is that it begins with an intuitive virtual reality interface where a human supervisor 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, projector-camera occlusion, the physical interactions of the fish and the box, and the previously demonstrated grasps of the human supervisor. This enables us to generate the large data sets required for deep learning, while including the intent of the human supervisor. In principle, this is similar to the algorithmic supervisor approach presented in [17]. A few differences between our work and [17] are: 1) in our work the grasps are placed by a human supervisor in VR, instead of an algorithmic supervisor; 2) in our work an end-to-end 3D CNN directly computes 6-DOF grasps from the input 3D occupancy grid, whereas [17] uses a GQ-CNN to evaluate multiple randomly-sampled and pre-aligned parallel-jaw grasps on depth images and select the grasp with the highest quality.

Our previous work [16] applied 3D convolutional neural networks (3D CNN) to robot grasping. Previously, 3D CNNs have been successfully applied to e.g. 3D shape recognition [4], [12]. Our main motivation in working with volumetric occupancy grid representations of point clouds, and 3D CNNs to analyze them, is to develop the foundation for deep learning in robotics applications that are camera- and viewpoint-agnostic. Hence the 3D CNN can work in an occupancy grid that is constructed by integrating 3D information obtained from multiple depth images, multiple cameras and/or multiple 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 occupancy grid will provide a more complete view. The advantage of the 3D CNN 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. Compared to our previous work [16], we have added domain randomization over the projector-camera offset, as well as coded an entirely new implementation of the 3D CNN in TensorFlow [18].

Our main contribution is to show that our approach to robot imitation learning from a human supervisor in VR transfers well to the real world, with a low or moderate number of demonstrations by the human supervisor. This validates our previous work [16], which was done entirely in VR.

II. System and Task Description

Our experiments are carried out on a 6-DOF Denso VS 087 robot arm mounted on a steel platform. An overview of the system can be seen in Fig. 1. The task is for a robot to pick up small fish of the species Atlantic herring (*Clupea harengus*) out of a box and place them in a second box, so that each fish can be weighed and processed individually. In this paper we focus on the first part of this task - grasping and picking the fish out of the box and placing them in the second box. A custom two-finger wormdrive gripper was designed, with compliant 3D-printed finger tips. The gripper consists of a single stepper motor attached to a 3D-printed hand. A wormscrew is mounted on the stepper motor axle and this drives two wormwheels - one for each finger. With the exception of the stepper motor, the entire gripper is 3D printed. Fig. 1 shows a closeup of the gripper and how the fingers are compliant enough to conform to the shape of small fish. The gripper is controlled by an Arduino, and has an adjustable gripper opening.

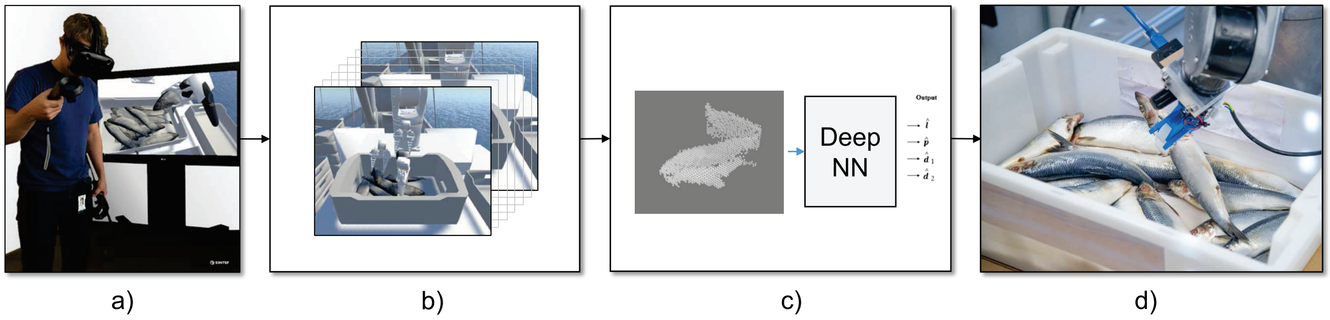

Fig. 2. The presented approach to robot imitation learning in VR, where a) the human supervisor provides a few example grasps by grasping fish from the box, b) domain randomization is used to generate a large synthetic data set based on the few example grasps, c) the deep neural network is trained on the large data set, and d) the grasp output from the trained deep neural network is used to control a gripper to grasp real fish.

An Intel RealSense SR300 depth camera is placed on the robot end-effector. The robot moves to three different poses and the camera acquires three depth images that are projected into an occupancy grid. The occupancy grid is processed by a 3D CNN implemented in TensorFlow. The output of the 3D CNN is a grasp certainty for each location in a downsampled 3D grid, and corresponding 6-DOF grasps defined by grasp placement position and orientation vectors for each location in that grid. A grasp is selected at the 3D location with the highest grasp certainty. The robot is commanded to perform this grasp, by placing the gripper, closing it and picking up the fish.

III. Robot learning

We use a deep 3D CNN to estimate grasps from an occupancy grid. 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. VR interface for demonstrating the task

A VR interface was created where a user - in this case a human supervisor - can enter a virtual environment 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 gripper mounted on the robot end effector. 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*). The fish physics were modelled using joints and colliders in the Unity game engine. Each fish consists of seven 3-DOF spring-damper revolute joints connecting eight rigid collider segments that approximate the fish shape. The coefficients of the spring-damper joints were hand-tuned until they visually matched the pose and dynamics of real fish recorded in various static and dynamic deformation scenarios. For rendering a realistic fish mesh, a neutral-pose fish mesh was rigged and weight-painted using the colliders. The purpose of weight painting is to smoothly deform the continuous neutral-pose fish mesh using a discrete set of colliders as its ”skeleton”. Friction coefficients of each collider were hand-tuned to visually match the sliding motion of real wet fish. This relatively simple physics modelling was complex enough to provide visually realistic images, while simple enough to render tens of fish at interactive rates (90 Hz).

Simulated fish physics ensures that the fish land in natural poses in a fish box placed in front of the robot. The task of the human supervisor 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. 5. This is the demonstration part of our imitation learning approach. In traditional imitation learning, 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 human supervisor'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

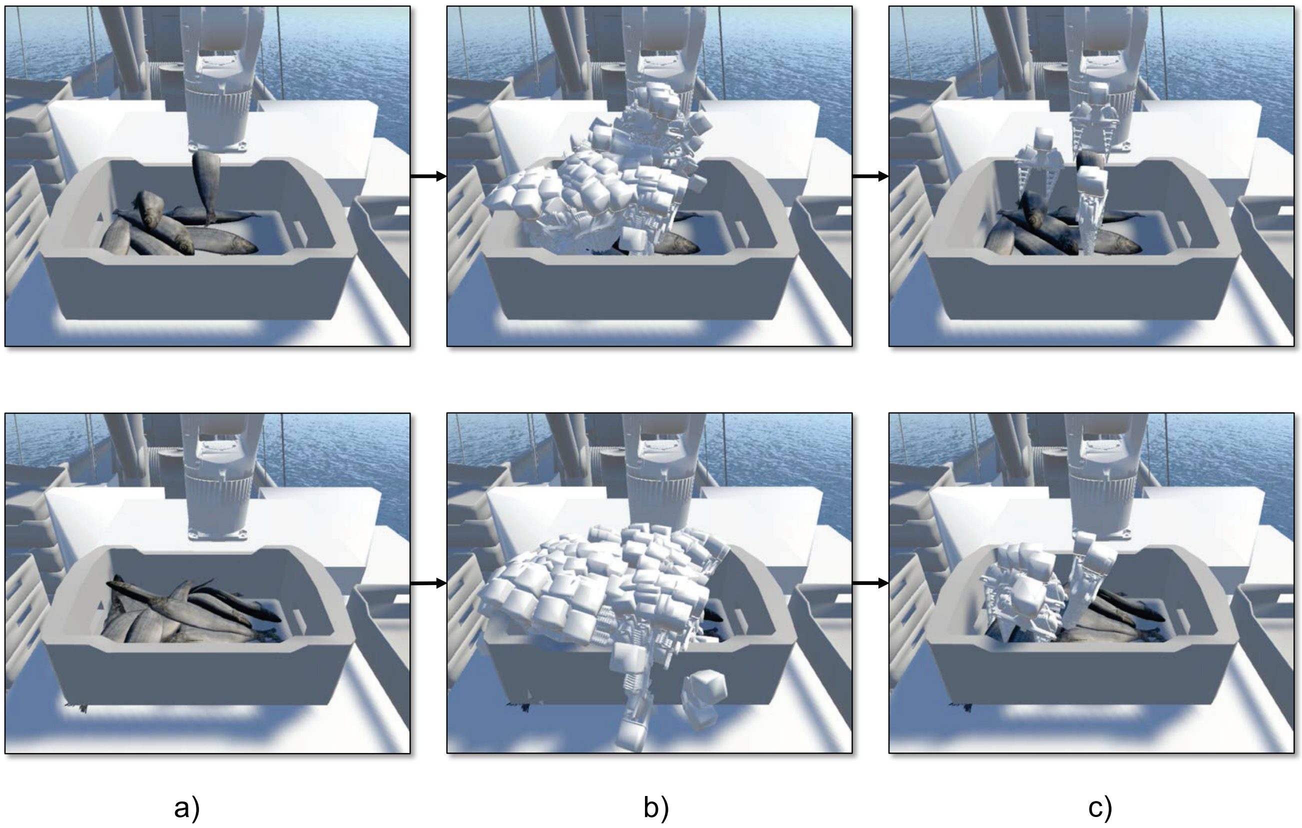

Fig. 3. Two examples (top row and botom row) illustrating the domain randomization approach for generating a large data set, by a) dropping a random number of fish in the box, using realistic fish physics, b) placing each of the logged grasps onto each fish, c) pruning the grasps based on collision heuristics.

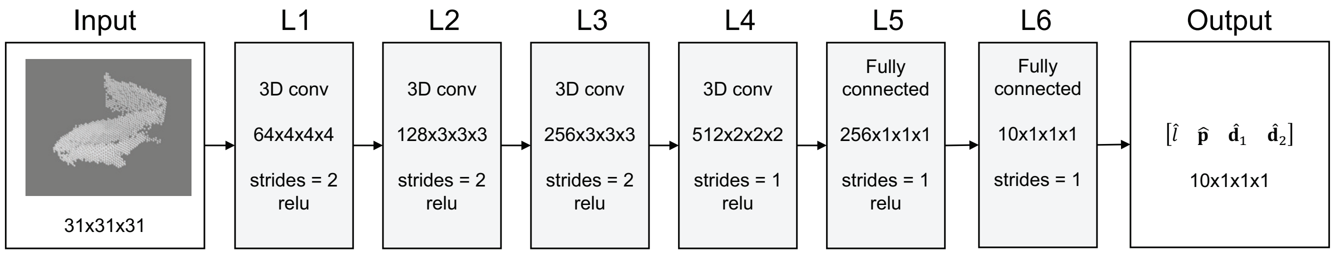

Fig. 4. The architecture of the 3D CNN consists of stacked convolutional layers, with striding (instead of max pooling) used to reduce the output resolution. The dense layers are swapped for $1 \times 1 \times 1$ convolutions to enable inputs of varying sizes. The activation functions for the feature extraction layers are rectified linear units, and in the last layer, $\hat{l}$ has a sigmoid activation function and the rest have linear activations.

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 human supervisor 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 human supervisor 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. 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

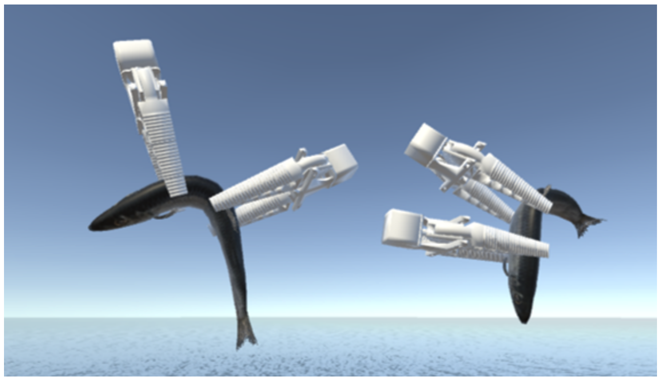

Fig. 5. The logged grasps after demonstration of two grasps in virtual reality. As the fish bends and twists, the grasps follow, making them valid for the fish regardless of pose.

(with an offset of $\frac{imagewidth/height}{2}$ in the xy-direction). Some 3D cameras, such as the Intel RealSense SR300, 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 and the offset between the projector and the camera, 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 network

The depth images are projected into a 3D occupancy grid, and we use a 3D CNN to estimate grasps from a receptive field volume in the occupancy grid, and split the problem up into three sub-problems

- Detecting probable grasp locations
- Estimating the precise grip point
- Estimating the orientation of the gripper

The architecture of the network is shown in Fig. 4. For a volume of size $31 \times 31 \times 31$, the output is a vector

$$\hat{\mathbf{y}} = \begin{bmatrix} \hat{l} & \hat{\mathbf{p}} & \hat{\mathbf{d}}_1 & \hat{\mathbf{d}}_2 \end{bmatrix}, \tag{1}$$

where $\hat{l} \in [0, 1]$ is a label that estimates the certainty that the input volume contains a valid grasp, $\hat{\mathbf{p}}$ estimates the position of a grasp within the input volume, $\hat{\mathbf{d}}_1$ and $\hat{\mathbf{d}}_2$ 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, capable of detecting multiple grasps within the input volume.

The predictions 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

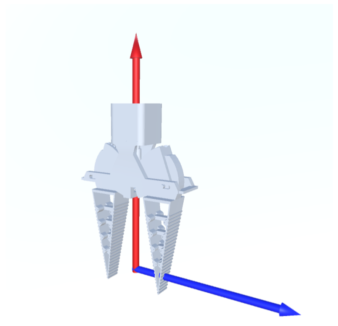

Fig. 6. The orientation of the gripper is defined by two vectors $\mathbf{d_1}$ (red) and $\mathbf{d_2}$ (blue). The position $\mathbf{p}$ is at the intersection of these two vectors.

$$J^{(i)} = J_C^{(i)} + l^{(i)}(0.1 \cdot J_O^{(i)} + 0.1 \cdot J_P^{(i)}), \tag{2}$$

where $J_C^{(i)}$ is the classification cost, $J_O^{(i)}$ the orientation estimation cost, $J_P^{(i)}$ the position estimation cost, $l^{(i)}$ is the true label for training example i. 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

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

The orientation of the gripper is defined unambiguously with two three dimensional vectors, each describing a direction in 3D-space (see Fig. 6). The total orientation cost for the $N = 2$ orientation vectors is given by

$$J_O^{(i)} = \sum_{k=1}^{N} \frac{1}{2}||\hat{\mathbf{d}}_k^{(i)} - \mathbf{d}_k^{(i)}||^2, \tag{5}$$

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

D. Preparing data for training

During training, the input to the network is a receptive field volume of size $31 \times 31 \times 31$ and the ground truth grasp certainty $l^{(i)}$ 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. In our experiments we generate false training examples (i.e. areas with a low probability of containing a grasp), simply by cropping random volumes of the occupancy grid and

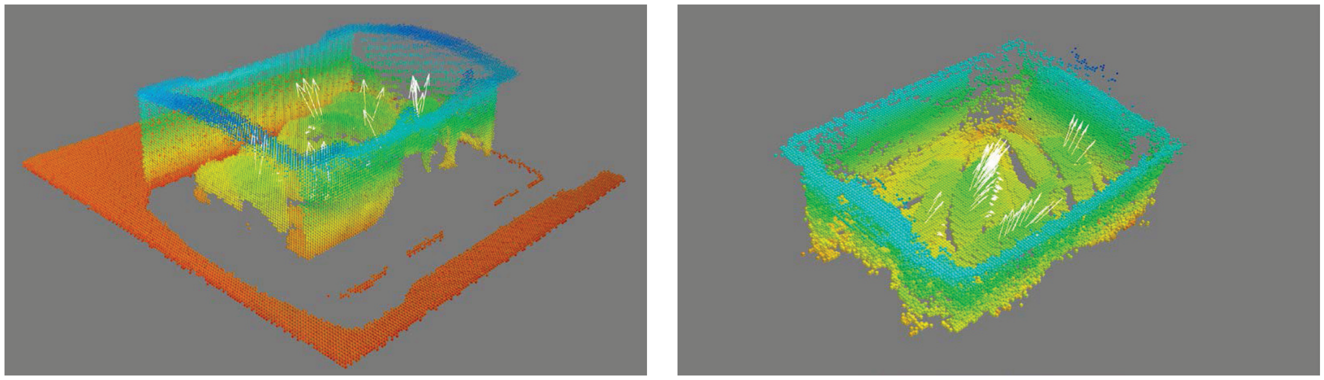

Fig. 7. Examples of grasp vectors placed by the 3D CNN in the occupancy grid of synthetic (left) and real (right) data.

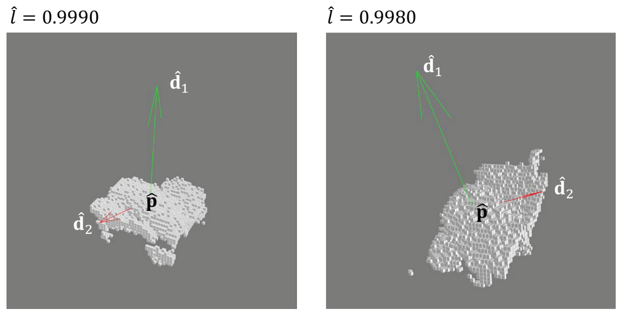

Fig. 8. Examples of predicted grasps on synthetic data, showing the contents of the receptive field and the predicted grasp vectors.

labelling them as false examples. Because the volumes that contain valid grasp are vastly outnumbered by the volumes that do not, the result is a false-data set with mostly true, but also some false, negatives.

IV. Experiments and Results

A. Generating synthetic data

Domain randomization was used to prepare the synthetic data set. In our experiments, 35 grasps were shown in VR. With these grasps, 3334 different random scenes were generated, each with a random number of fish between 1 and 25. For each of these scenes, 3 depth images were generated by randomly selecting a viewpoint and projector-camera offset. The result of this is more than 10000 synthetic depth images of fish boxes containing between 1 and 25 fish. From these depth images, 100000 examples were cropped where 50% of the data set did not contain a grasp.

B. Training the 3D CNN

The data was split into a training set of 100000 examples and a validation set of 4000 examples. The 3D CNN was trained with the Adam [19] optimizer on the training set. Early stopping was used to stop training before the cost function started increasing on the validation set. After a few hours the training was stopped, and the 3D CNN was tested on some synthetic data and visualized to inspect the outputs of the 3D CNN, before proceeding to real-world tests. The 3D CNN takes as input an occupancy grid and outputs grasp certainties and grasps for the entire volume of the occupancy grid. Examples of a full synthetic occupancy grid with predicted grasps is shown in Fig. 7 (left), and examples of two receptive fields with predictions can be seen in Fig. 8.

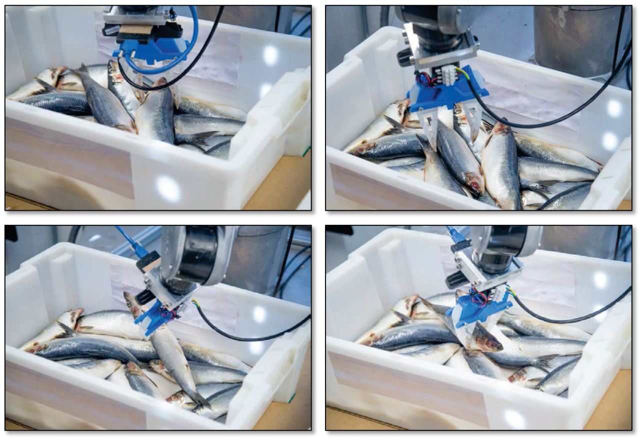

Fig. 9. Examples showing grasping of real fish.

C. Experimental protocol

The 3D CNN was evaluated in an experiment with real fish. The setup included the robot with a box in front of it, where fish are placed. The fish box also contains some water. The robot is tasked with picking fish, one at a time, from that box and placing them into a second box. This setup is shown in Fig. 1. The goal of the experiment was to measure the grasping success rate and failure rate, in picking up the fish from the first box and placing it into the second box. We also wanted to determine the types of failures. The experimental protocol was as follows:

1) Place a box of 25 fish in front of the robot.
2) Scan the box and compute the occupancy grid.
3) Detect grasps for the entire occupancy grid and select the most certain grasp.
4) Attempt the grasp.

5) If successful grasp, goto point 2.
6) If unsuccessful, log the failure type and randomize box:
   a) Remove the box from in front of the robot.
   b) Pour contents of the box into a second box.
   c) Pour contents of the second box back into the first box.
   d) Place the box back in front of the robot.
   e) Goto point 2.

The grasping was continued until the box was empty.

D. Failure types

A grasp was judged as a success if a fish was successfully moved from one box to another. If the robot failed to do so the grasp was judged as a failure. The two main reasons for failure were bad grasps and collisions. A failure was logged in the bad grasps category if:

1) The robot failed to pick up the fish.
2) The fish was dropped during transfer to the second box.

Collisions were logged in three separate subcategories:

1) Gripper collisions within the 3D CNN's receptive field.
2) Gripper collisions on approach to an otherwise valid grasp.
3) Robot and camera collisions.

The category "NN failures" in Table I excludes the failures from the second and third collision failure types. These failures should not be credited to the 3D CNN because the conditions for success are unobservable for the neural network. Additionally failed grasps where the highest predicted grasp certainty was less than 0.50 were not included in 'NN failures'.

E. Real-world test results

Example grasps can be seen in Fig. 9, showing how the gripper approaches the grasp point, places the grasp and begins picking up the fish. Fig. 7 (right) shows an occupancy grid computed during the real-world tests.

The experimental protocol was performed on a total of seven boxes, and the successes and failures were categorized. The results are summarized quantitatively in Table I. Referring to that table, we see successes and failures of the grasping task, as well as the success and failures attributed to the neural network (NN). There are two boxes that deviate significantly from the others. Box 2 contained fish that were not very fresh, resulting in an oily film on the fish and a very greasy box after repeated randomization of it. This affected the depth imaging and resulted in more slippery and soft fish that were difficult to grasp. The average grasp certainty $\hat{l}$ is significantly lower for box two, suggesting the quality of the imaging was affected by the oily film. During the experiment on box 3, the gripper failed after 19 grasp attempts and a replacement part had to be 3D printed, and the experiments continued the next day.

The overall success rate was 74 %, and this increased to 80 % when excluding failures that could not be attributed to the neural network. On average the successful grasps had a higher predicted grasp certainty than the unsuccessful ones, 0.98 and 0.92 respectively. This suggests that a threshold on grasp certainty values could yield better results by minimizing failed grasp attempts.

TABLE I
GRASPING RESULTS ON REAL FISH

Box	Success (%)	Success	Failure	NN success (%)	NN failure	Avg. $\hat{l}$
1	71 %	25	10	78 %	7	1.00
2	57 %	25	19	71 %	10	0.84
3	74 %	14	5	74 %	5	1.00
4	86 %	25	4	86 %	4	1.00
5	81 %	25	6	83 %	5	0.99
6	83 %	25	5	96 %	1	0.99
7	76 %	25	8	76 %	8	1.00
All	74 %	164	57	80 %	40	0.97

V. Discussion and future work

The results of our work, suggest that our approach to domain randomization in virtual reality is an efficient method of transferring knowledge to a robot. Based on only a few demonstration examples, a sufficiently large and diverse synthetic data set was generated that was capable of training a large 3D convolutional neural network. The real-world experiments showed an overall grasping success rate of 74 %, which increases to 80 % when considering only the failures that could be attributed to the neural network. The task of grasping slippery fish, with a narrow elastomer gripper, is very unforgiving with respect to grasp placement errors. If the grasp is placed with a significant offset, in any direction, from fish's center of gravity, the probability of the fish sliding or dropping out of the gripper increases. Therefore the task requires precise grasp placement. To achieve this at greater than 80 % success rate, our conclusion is that the receptive field size will have to be increased. Increasing the receptive field size will enable the neural network to observe more of the fish and its pose and position relative to the box and other fish. These factors can reduce collisions and improve grasp placement accuracy. This results in a larger 3D CNN. For example, increasing the receptive field from $31 \times 31 \times 31$ to $63 \times 63 \times 63$ will result in an eightfold increase in the number of parameters in the neural network, and a similar increase in training time and the required amount of training data. Considering the limited observations that can be made within a small receptive field (see Fig. 8) it is understandable that an larger receptive field may improve the ability of the neural network to estimate accurate grasp placements. For this reason, our future work will be focused on increasing the receptive field size. We will also adjust the dimensions of the gripper used in the domain randomization, so that it better matches the dimensions of the actual end-effector and robot gripper. We expect that this will reduce the number of errors attributed to collisions with the box. Other avenues of future work, include learning sequences of actions, such as patterned packing (i.e. packing fish in a box according to a predefined number of layers and orientation of the fish in each layer), without re-grasping the fish after picking it out of the first box.

Our hypothesis from the onset is that through our approach, VR can serve as a efficient medium for demonstrating complex tasks to robots. A step towards testing this hypothesis was presented in this paper, and the results are promising enough to maintain our hypothesis. One may question whether the presented approach is unnecessarily complex for picking fish, and that we should have compared it to a baseline, such as antipodal grasp sampling with automated grasp placement. Investigating this was outside the scope of this paper, since our focus was on testing our hypothesis as a step towards more challenging applications of our approach.

Beyond our application to grasping and handling fish, we will also expand the domain randomization methodology and neural network architecture presented to the more generic problem of grasping and handling multiple types of objects.

ACKNOWLEDGMENT

This project was funded by 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 IEEE 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, "Training convolutional neural networks in virtual reality for grasp detection from 3D images", Master's thesis, University of Stavanger, 2016. URL: http://hdl.handle.net/11250/2413962

[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] Shotton, J., Fitzgibbon, A., Cook, M., Sharp, T., Finocchio, M., Moore, R., Kipman, A. and Blake, A., 2011, June. Real-time human pose recognition in parts from single depth images. In Computer Vision and Pattern Recognition (CVPR), 2011 IEEE Conference on (pp. 1297-1304). IEEE.

[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)

[16] J. Dyrstad and J. R. Mathiassen, "Grasping Virtual Fish: A Step Towards Robotic Deep Learning from Demonstration in Virtual Reality," In 2017 IEEE International Conference on Robotics and Biomimetics (ROBIO), 2017, In press.

[17] J. Mahler and K. Goldberg, "Learning Deep Policies for Robot Bin Picking by Simulating Robust Grasping Sequences," 1st Conference on Robot Learning (CoRL). Mt. View, CA. Nov 2017. Proceedings of Machine Learning Research volume 78.

[18] A. Martn, P. Barham, J. Chen, Z. Chen, A. Davis, J. Dean, M. Devin et al. "TensorFlow: A System for Large-Scale Machine Learning." In OSDI, vol. 16, pp. 265-283. 2016.

[19] Kingma, D., and Ba, J. (2014). Adam: A method for stochastic optimization. arXiv preprint arXiv:1412.6980.