Grasping Unknown Objects by Coupling Deep Reinforcement Learning, Generative Adversarial Networks, and Visual Servoing

Ole-Magnus Pedersen, Ekrem Misimi, François Chaumette

Grasping Unknown Objects by Coupling Deep Reinforcement Learning, Generative Adversarial Networks, and Visual Servoing

Ole-Magnus Pedersen
Norwegian Univ. of Science and Technology
Trondheim, Norway

Ekrem Misimi
SINTEF Ocean
Trondheim, Norway

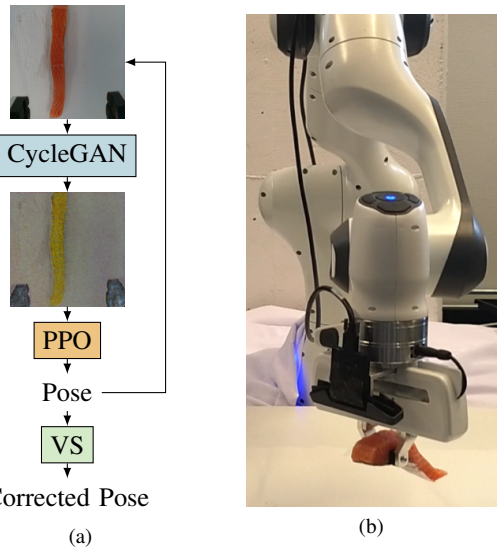
François Chaumette
Inria, Univ. Rennes, CNRS, IRISA
Rennes, France

***Abstract*— In this paper, we propose a novel approach for transferring a deep reinforcement learning (DRL) grasping agent from simulation to a real robot, without fine tuning in the real world. The approach utilises a CycleGAN to close the reality gap between the simulated and real environments, in a reverse real-to-sim manner, effectively "tricking" the agent into believing it is still in the simulator. Furthermore, a visual servoing (VS) grasping task is added to correct for inaccurate agent gripper pose estimations derived from deep learning. The proposed approach is evaluated by means of real grasping experiments, achieving a success rate of 83 % on previously seen objects, and the same success rate for previously unseen, semi-compliant objects. The robustness of the approach is demonstrated by comparing it with two baselines, DRL plus CycleGAN, and VS only. The results clearly show that our approach outperforms both baselines.**

I. Introduction

In recent years, we have witnessed considerable progress in the use of deep learning (DL) for a variety of robotic applications, and specifically in the field of vision-based robotic manipulation [1]–[5], either as a valid addition or an alternative to traditional robot control. Although DL approaches to vision-based robotic manipulation are popular, they require a huge volume of labelled image data with correct grasp poses for training [2], [3]. Substantial work has been done to collect large-scale datasets [2], [6] and utilizing the data more efficiently, e.g. through multitask learning [4], or by making DL methods faster and more applicable to robotics [1]. The need for large realistic data sets has turned out to be time-consuming and expensive, and one way to avoid this is deep reinforcement learning (DRL).

DRL is a class of machine-learning techniques by which a policy for acting in the environment is learned by maximizing a perceived reward. The use of DRL methods, such as for example proximal policy optimization (PPO) [7], applicable to continuous action spaces such as robotic grasping, has become very popular, in particular after the work presented by Mnih *et al.*, with many reported applications specifically on vision-based grasping [7], [9]–[13]. The need for self-exploration makes it impractical to train a DRL-agent on a real robot due to the potential for erratic behavior in the early stages of training, which can result in damage to the robot or its environment. This consideration has promoted the use of simulated data for training. As an example, [14] train a grasp quality convolutional neural network (CNN) solely on synthetic data, and use the model to perform grasp planning with a success rate of 80 % on novel objects.



(a) (b)

Fig. 1. (a) Our approach: A real image from the camera is adapted directly to simulation using a CycleGAN. The adapted image is fed to a PPO-agent, which calculates a new grasping pose before a VS task is activated to refine the final pose. (b) Robot after successfully grasping a salmon fillet, as a challenging object, during testing trials. The VS assisted grasping agent was not presented with any instances of the salmon during training in simulation. This shows that the agent generalizes well to previously unseen objects.

However, these approaches entail a serious drawback, namely the reality gap between the simulator and real world, which creates challenges when transferring the learned interaction experience from simulation into the real world. Pas *et al.* [5] note this for depth data, reporting a large decrease in performance when training on simulated data. A variety of transfer learning techniques are thus being developed with the aim of achieving successful transfer learning from simulation to a real robot. These include domain adaptation (DA) [15]–[17] and domain randomization (DR) [18]–[21].

DR is a transfer learning technique in which the training environment is randomised, with the aim of making the agent robust to handle the real world simply as another state variation [20]. However, several disadvantages have been observed using this technique, such as a failure to detect objects on the edge of the image frame [19] or a failure to grasp irregular objects [20]. It is also shown that domain randomisation is less efficient than a combination involving domain-adaptation methods [22]. DA methods, such as

splitting the model into a perceptual and control module and then retraining the perceptual module for new environments, have been proposed in [23], [24] to improve transfer learning, however the drawbacks include expensive retraining and that the representation connecting the two modules limits the information available to the control module.

Another interesting approach to transfer learning is to make input images from two different domains appear similar to the system. Such an approach can enable an agent to operate in a completely new environment, without the need for fine tuning. This type of domain-adaption has been investigated using generative models such as variations of the generative adversarial network (GAN) [25], which consists of two networks, a generator G that generates images and a discriminator D that discriminates between real and generated images. The networks are trained in an adversarial fashion, until G learns to generate realistic images.

The CycleGAN [26] is a particular extension of the GAN, consisting of two GANs. By training these in tandem, the system learns to map between images in two domains, such as a real and a simulation domain. The CycleGAN has been used extensively for domain-adaptation tasks, such as music genre transfer [15], chest X-ray segmentation [16], and person re-identification [27]. James *et al.* [17] use a conditional GAN (cGAN) [28] trained to adapt randomised simulated environments to a canonical simulated environment. The same cGAN is used to adapt from real to simulated images, combining domain adaptation using both cGAN and DR. They report the presence of artifacts in the adapted images that make grasping objects challenging, and an overall success rate of $70\,\%$ on previously unseen objects. Our work differs from [17] in that we map real to simulated images directly without an adaptation step, and show improved grasping results. The downside is that we need a (small) set of data from the real robot to train our generator, while theirs can be trained only from simulated scenes. However, the collection of such a small dataset is typically cheap, and the gained performance justifies its cost.

Bousmalis *et al.* [29] show the use of a GAN for unsupervised domain adaption of RGB-D-images, introducing the use of a task-specific loss and a content-similarity loss to further guide the training of the GAN. The main difference to our approach is that theirs requires a task-specific network to be trained in parallel to the domain-adaption, which is unrealistic when performing unsupervised learning.

The aforementioned methods for transfer learning of a grasping pose agent to the real world commonly suffer from an inability to correctly position the gripper to the targeted pose [20]. To tackle this issue, we propose to supplement the grasping agent with a visual servoing (VS) task that can be activated in situations where the grasping pose is incorrectly estimated, and whose goal is to refine it prior to grasping. VS is a collection of closed-loop robotic control techniques based on visual feedback [30]. This paper uses a geometric VS task dependent on a segmentation of the scene, similar to [31], however in this work the segmentation network is trained with the domain adaption network, instead of using a separate video object segmentation network.

In this work, a novel approach for the transfer learning of a robotic gripper pose estimation to the real world is presented. First, in a reverse real-to-sim manner, real camera images are transformed to simulated ones by a CycleGAN for domain adaptation. The grasping pose estimation agent is trained in simulation using the PPO algorithm prior to using VS to refine the final grasping pose in the real world (see Fig. 1a). Although a similar approach to reality-to-simulation transfer has been investigated for robotic driving [32], this is, to the best of our knowledge, the first work using a GAN to achieve direct real-to-sim image mapping for domain adaptation of a robotic grasping policy. Moreover, our approach, involving addition of the VS task, largely solves the problem that commonly arises when transferring a grasping agent to the real world [20], namely the failure of the agent to correctly position the gripper to the final target grasping pose. The robustness of our approach using real-to-sim transfer learning and VS refining (Fig. 2) was evaluated by comparing it with two baselines on a YCB object set [33] and a testing set of previously unseen compliant objects. Our approach clearly outperformed both baselines, with a success rate of $83\,\%$ for both sets of objects.

II. Methodology

The method proposed in this paper has two primary stages. For three discrete timesteps, the DRL and CycleGAN is used to iteratively improve the grasping pose. After reaching the pose calculated by the DRL and CycleGAN, the VS task further refines the grasping pose prior to grasping the object. In the following descriptions, the coordinate system is defined as follows: x is the direction parallel to the gripper, the y-axis is perpendicular to the gripper, and the z-axis is the direction the gripper is pointing. For images, the same coordinate axes are used, except that they are relative to the camera.

A. Deep Reinforcement Learning

The DRL agent is trained in simulation using an actor-critic version of the PPO algorithm [7]. The simulator is created in Unity3D, and includes a robot gripper, identical to the one used by the real robot in the experiments, objects to be grasped, cameras for capturing color and depth images, and a semantic segmentation mask used by the CycleGAN. An example of such images is shown Fig. 3.

When run, the DRL system takes the current RGB-D image of the scene as an input, and calculates a Cartesian pose p for the robot to move to, relative to the current pose. The z-component of the poses from the first and second DRL steps are set to move the gripper $15\,\mathrm{cm}$ towards the ground, to ensure a combination of registering more detail in the frame while not moving too close, possibly loosing a part of the object from the image. For the last step, the z component is calculated by the agent when VS is inactive, and set to 0 when VS is active. The reward function during training is based on a set of predefined successful grasp poses, and rewards the agent for being close to these poses. Between

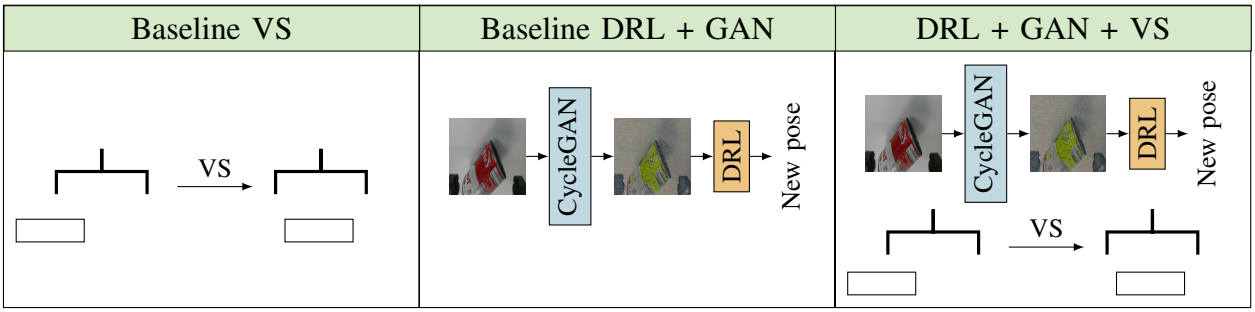


Fig. 2. Our approach (DRL+GAN+VS) compared to two baselines. The first baseline is a classical VS grasping task. The second baseline is the DRL+GAN.

(a) RGB (b) Depth (c) Segment mask

Fig. 3. Example simulated images. The color and depth images are used by the PPO, while the segmentation mask is used to train the CycleGAN.

two and ten good poses were defined for each training object, and some examples of such poses for a single object can be seen in Fig. 4. The reward function is defined as

$$R = \frac{\lambda_d(1-e_d)^2 + \lambda_o(1-e_o) + \lambda_g r_g - \lambda_l}{\lambda_n} \tag{1}$$

where e_d and e_o are the linear and angular distances from a good grasp, respectively, and r_g equals -1 if any point of the gripper is below ground and 0 otherwise. $\lambda_d, \lambda_o, \lambda_g$ are balancing terms, set to 0.5, 0.5 and 0.05, respectively. λ_l and λ_n are used to normalize the reward, both set to 0.5, yielding a reward in the range $[-1.1, 1]$.

Fig. 4. Example of predefined successful gripper poses used for training the DRL agent in simulator.

B. Semantic CycleGAN for Image-to-Image Translation

A CycleGAN was used to map RGB-D images from the real world to the simulated environment. In addition to the adversarial and cycle-consistency losses used in the original CycleGAN model, two other losses were used. First, an identity loss, defined by

$$L_{id} = \mathbb{E}_{\mathbf{x}\in X}\left[||G(\mathbf{x};\theta^{G_y}) - \mathbf{x}||\right] + \mathbb{E}_{\mathbf{y}\in Y}\left[||G(\mathbf{y};\theta^{G_x}) - \mathbf{y}||\right] \tag{2}$$

where, X, Y are the simulated and real image domains, and $\theta^{G_x}, \theta^{G_y}$ are the generator parameters for the generator translating in the $Y \to X$ and $X \to Y$ direction, respectively. The loss is used to ensure the output images are not too far from the input. This idea is similar to the content-similarity loss of [22], [29]. Second, similar to [34], a semantic loss was added. This was achieved by generating a semantic segmentation mask from the discriminator along with the discriminator value, and using a pixel-wise cross-entropy loss. As ground-truth segmentation masks are, in general, only easily available from the simulated environment, this loss was calculated on raw and adapted images from that domain, and not on images from the real robot. The semantic loss is:

$$\begin{aligned} L_{sem} =& \mathbb{E}_{(\mathbf{x},\mathbf{s})\in X'}\Big[H\left(D_{sem}\left(G\left(\mathbf{x};\theta^{G_y}\right);\theta^{D_y}\right),\mathbf{s}\right) \\ &+ H\left(D_{sem}\left(\mathbf{x};\theta^{D_x}\right),\mathbf{s}\right)\Big] \end{aligned} \tag{3}$$

where H is the pixel-wise cross entropy loss, D_{sem} the segment mask output by the discriminator, $\theta^{D_y}, \theta^{D_x}$ are the discriminator parameters for the GANs in the $X \to Y$ and $Y \to X$ directions, respectively, and $X' = \{(\mathbf{x}, seg_{\mathbf{x}}) | \mathbf{x} \in X\}$ is a set consisting of paired images from the simulator and their ground-truth segment mask. The use of these additional losses significantly improved the CycleGAN's ability to preserve geometry in initial experiments.

Altogether, the total loss function becomes

$$L = L_{GAN} + \lambda_c L_{cyc} + \lambda_i L_{id} + \lambda_s L_{sem} \tag{4}$$

where λ_c, λ_i, and λ_s are weights to balance the different terms, respectively 10, 1, 1 in this work. L_{cyc} is the cycle-consistency loss from [26], i.e.,

$$\begin{aligned} L_{cyc} =& \mathbb{E}_{x\in X}||G\left(G\left(\mathbf{x};\theta^{G_y}\right);\theta^{G_x}\right) - \mathbf{x}|| \\ &+ \mathbb{E}_{y\in Y}||G\left(G\left(\mathbf{y};\theta^{G_x}\right);\theta^{G_y}\right) - \mathbf{y}|| \end{aligned} \tag{5}$$

In this work, L_{GAN} is set to the LSGAN loss [35], as it consistently outperformed the WGAN-loss [36] in preliminary experiments. The LSGAN-loss is [35]:

$$\begin{aligned} L_{LSGAN}(D) =& \frac{1}{2}\mathbb{E}_{x\in X}\left[\left(D(x;\theta^{D_x}) - 1\right)^2\right] \\ &+ \frac{1}{2}\mathbb{E}_{y\in Y}\left[D(G(y;\theta^{G_x});\theta^{D_x})^2\right] \\ L_{LSGAN}(G) =& \frac{1}{2}\mathbb{E}_{y\in Y}\left[\left(D(G(y;\theta^{G_x});\theta^{D_x}) - 1\right)^2\right] \end{aligned} \tag{6}$$

Only the loss in the $Y \rightarrow X$ direction is shown, but the definition in the $X \rightarrow Y$ direction is identical.

The network architectures for both the generator and discriminator networks, in this work, are U-nets [37], with seven layers in the encoder and seven in the decoder. In the generator, dropout is applied to all layers in the decoder except the first. The discriminator value is calculated as the mean of the hyperbolic tangent of the third layer in the discriminator, while the segment mask is the output of the full network. For training the CycleGAN, a dataset of 655 RGB-D images were manually collected from the real robot, and the same amount of images were automatically generated from the simulator. Images were augmented by horizontal flipping, and training was done for 200 epochs.

C. Visual Servoing

The aim of the VS task in this work is to refine the final grasp pose estimated by the DRL system. To do this, the following visual features were selected:

- The x coordinate x_g of the object centroid should be positioned at the value x_g^*, chosen as the coordinate of the midpoint between the gripper fingers.
- The y coordinate y_g of the object centroid should be positioned at the value y_g^*, chosen so the object is between the gripper fingers when moving the gripper towards the object in the camera depth direction.
- The orientation α of the objects primary axis in the image should reach the angle α^* to be perpendicular to the angle of the line between the gripper fingers. If the difference of the sizes of the objects primary and secondary axes in the image is small, this feature is disregarded, and no angular correction is performed.

The features are illustrated in Fig. 5. From the features, the lateral translational components t_x and t_y and the rotational component θ around the optical axis are given by:

$$t_x = Z_g(x_g - x_g^*)\,,\; t_y = Z_g(y_g - y_g^*)\,,\; \theta = -(\alpha - \alpha^*) \quad (7)$$

where Z_g is the distance in the z axis from the camera to the object, measured in the depth image. By using a classical pose-based visual strategy [30], the control output $\mathbf{v} = (v_x, v_y, \omega_z)$ has the following simple form:

$$\mathbf{v} = -\lambda \begin{pmatrix} \cos\theta t_x + \sin\theta t_y \\ -\sin\theta t_x + \cos\theta t_y \\ \theta \end{pmatrix} \quad (8)$$

where λ is a positive gain.

The image measurements used for VS are calculated using the image moments of a segmentation mask. Specifically, the semantic discriminator trained with the CycleGAN was used to obtain a segmentation mask. Then the largest segment with the class "object" was chosen as the object and used for tracking, feature extraction, and control law computation using the ViSP library [38]. The segmentation and tracking procedure is shown in Fig. 6.

Since the segmentation network is trained with the CycleGAN, its use comes at no additional cost. Nevertheless it was compared to other methods. Fig. 7 compares our method to a histogram-based segmentation using K-means and the Watershed algorithm, showing that DL is most efficient in separating the background, object and gripper fingers.

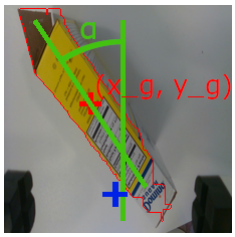
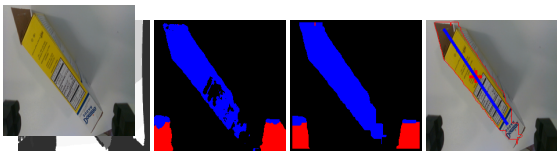


Fig. 5. Illustration of the features used for the VS in this work: the object's centroid and orientation. The blue cross shows the target position (x_g^*, y_g^*).



(a) Input Images (b) Raw segments (c) Cleaned segments (d) Tracked blob

Fig. 6. Tracking objects in the VS module. RGB-D images (a) are fed to a network that generates a segmentation mask (b). The mask is cleaned up using morphological operations (c). The largest segment in this mask is found, and the blob tracker of ViSP is used to track the object (d). The blob tracker calculates the image moments of the segment, which are used to find the centroid and orientation of the object.

(a) Color (b) Histogram (c) Watershed (d) Semantic Discriminator

Fig. 7. Segmentation using a variety of methods. It is shown how the semantic discriminator, in this work, outperforms the traditional methods.

In preliminary experiments, it was observed that large angular errors led to an excessive velocity being applied to the end-effector during VS, leading to the object disappearing from the image frame. Hence, a two-step VS-approach was adopted, where the first step consists of using only the linear features (x and y position of the object centroid) for VS, while the second step uses the full feature set. Initially, the linear step is performed, and when it convergences the full VS is initiated. If at any point of the full VS the object moves too close to the image border, the system reverts to the linear step to avoid losing the object, and this process is repeated until VS converges. For the experiments in this work, the VS policy uses an adaptive gain [39], so λ varies between 0.7 and 0.3 depending on the feature error.

The final step of the VS is to move the gripper down along the camera z axis, to grasp the object. The distance to move is obtained from the depth camera.

III. Results and Discussion

Testing trials were performed to evaluate our approach for transfer of the grasping agent from the simulation to the real robot, in a reverse real-to sim manner, with VS.

A. Experimental Setup

The experiments were performed with a Panda robot arm with 7 DoF, equipped with a simple, two-fingered gripper. An Intel RealSense SR300 camera was mounted to the gripper, providing RGB-D images of the current scene. Objects were placed on a white background without clutter. Although outside of the scope of this paper, some experiments with a non-uniform background were attempted, but the trained CycleGAN was unable to generalize well to such environments, since no such backgrounds were included in the training set. The real-world setup can be seen in Fig. 1b.

The method was evaluated on the task of grasping ten objects from the YCB dataset [33], and six previously unseen objects. The objects were chosen to be similar in shape and size to relevant compliant food objects, and to fit in the generic gripper on the Panda robot. At the beginning of each episode, the object was moved to an arbitrary position in the camera's field of view, and the pose estimation was run. Three grasping episodes were run for each object, each baseline and our approach, resulting in a total of 144 tests.

B. Results

The results of the experiments with the known objects are shown in Fig. 8, and the results of the experiments with the previously unseen objects are shown in Fig. 9. The figures show the number of successful pose estimations out of three attempts when using only VS, only DRL with the CycleGAN for domain adaption, and our approach using both VS and DRL + CycleGAN. We chose 3D semi-compliant objects, previously unseen for the grasping agent, to validate our approach, in contrast to the majority of state-of-the art work in robotic manipulation, primarily focusing on rigid objects.

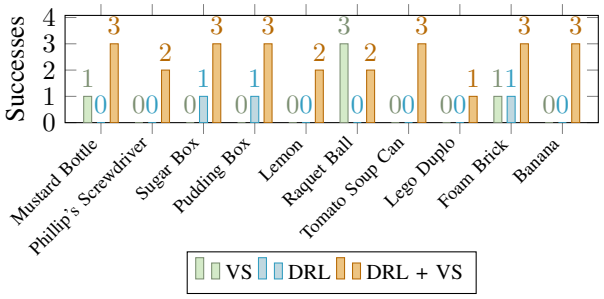


Fig. 8. Results of grasping previously seen objects. Three grasp attempts for each object. The total success rate was 17 % (VS), 10 % (DRL), and 83 % with our method.

C. Baseline 1: VS grasping task

The target y-position y_g of the object in the image was set for the case where VS was run after the DRL grasping agent, where the robot had already moved 30 cm towards the ground. This led to an error of about 4 cm in that direction

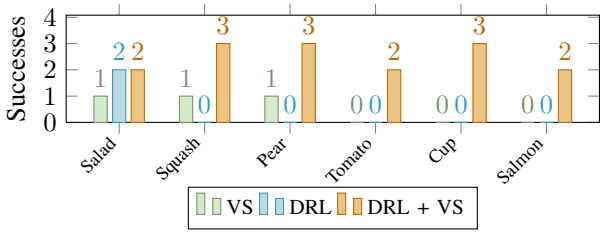


Fig. 9. Results of grasping previously unseen objects. Three grasp attempts for each object. The total success rate was 39 % (VS), 11 % (DRL), and 83 % with our method.

for all nearly successful grasps in the VS-only trials. These grasping attempts were counted as successes, given that fine-tuning the target would easily account for this discrepancy.

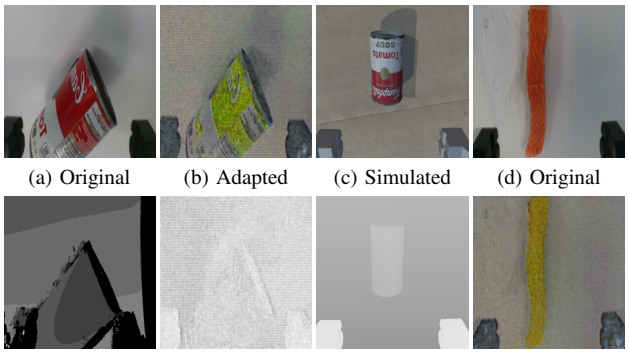
As the result plots show, the VS grasping task in these experiments did not succeed in grasping objects. In general, VS worked well during the linear servoing step, but when correcting for angular errors, it had a tendency to make too large adjustments, often losing the object from the camera field of view or registering parts of the surrounding environment as the object.

In addition to the problems with the angular errors, it was also shown that the VS system was much more dependent on a very good segmentation mask, due to a worse starting pose. VS managed the linear pose estimation well, confirmed by the fact that the cases where it was successful were typically grasping round objects where the orientation doesn't matter, but it failed when correcting for large angular errors.

D. Baseline 2: DRL and CycleGAN

It is immediately apparent from the results that using the CycleGAN alone was not sufficient for transfer of the pose estimation policy. However, while the pose estimation was unsuccessful, we observed that the system was typically successful or close to successful in positioning the gripper correctly in the x and y directions, as well as orienting the gripper correctly. However, large errors in the linear z axis were observed, leading to the gripper stopping between 2 and 10 cm above the object in most of the grasping trials.

Looking at the adapted images in Fig. 10 ((b) and (e)), we see that the color image is adapted with a relatively high success, while the CycleGAN is less effective in adaptation of the depth images. Specifically, the depth image is very noisy and lacks the clear definition present in the simulated images. There is also a large possibility that the distance the image encodes is incorrect, as there is no explicit mechanism to ensure this is kept the same during the adaptation. These aspects give light as to why the system using only the CycleGAN fails in the z direction. However, the fact that the color image translation is of such a high quality geometry-wise shows how this method is well-suited to finding planar and angular poses. Furthermore, Fig. 10 (d) and (h) show that this property holds true for previously unseen objects as well.



(a) Original (b) Adapted (c) Simulated (d) Original

(e) Original (f) Adapted (g) Simulated (h) Adapted

Fig. 10. Example RGB (top) and depth (bottom) images from the real robot ((a) and (e)), adapted from the real robot using the CycleGAN ((b) and (f)) and from the simulator ((c) and (g)). (d) and (h) show an example of adaption of a previously unseen salmon object. Note that while the colors of the salmon change, the network manages to keep the geometry.

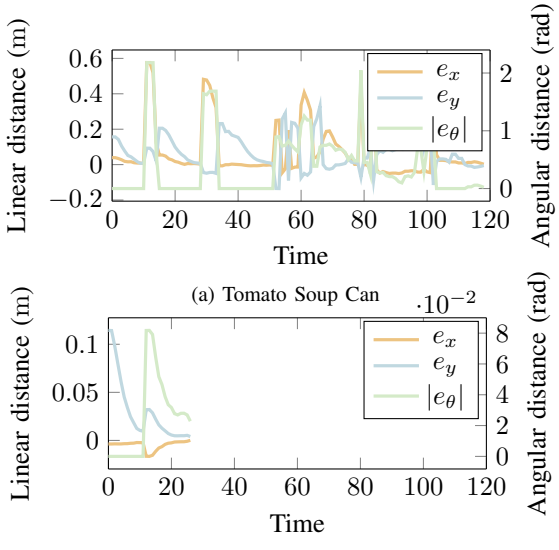
E. Our approach: DRL, CycleGAN and VS

When using both the CycleGAN and VS for transfer learning, the system is successful in most of the grasping trials, and for all but two trials, the gripper was close to a good grasp, with a mean distance of approximately $3\,\mathrm{cm}$ for the failed attempts. The final two grasps failed early due to errors in the VS stage, and the distance was not measured. We observed that our approach takes advantage of the strong points of each of the parts. Specifically, the DRL + CycleGAN works exceptionally well for finding an approximate angular and planar pose. This pose becomes the starting point for the VS, meaning there is seldom need for large angular corrections, which is where the VS-only system typically failed. With such a good initial pose, the VS task is able to refine the pose leading to a successful grasp.

The fact that our approach works well is further illustrated in Fig. 11. The figure shows the error plots for two runs of the VS system, one where the DRL + CycleGAN was unsuccessful in finding a good initial angular pose, and one where it was successful. The plots clearly show how the run with a good initial grasping pose converges quickly, while the other uses a long time, needing to revert to the linear servo several times before finally converging.

While the system tends to work very well, we observed two cases where it either failed completely or took a long time to converge. The first is when the DRL + CycleGAN failed to provide a good inital pose for the VS, and the VS taking more time to converge as a consequence. The second is when the segmentation mask had a low quality. This could lead to oscillations of the measured features. Both of these errors were rare in our trials and can be seen as minor.

Finally, we observed that our approach performed as well on previously unseen, semi-compliant objects as on objects from the training set. This shows that our method is able to learn general features from the images and can generalize to a variety of new objects, implying a potential for the reuse of the system for novel grasping tasks without retraining, which is a desired property of such systems.



(a) Tomato Soup Can

(b) Salmon

Fig. 11. Two error plots from the VS stage. (a) The DRL stage failed in estimating the correct orientation of the gripper, leading to very large angular velocities. This led to the system reverting to the linear VS, to avoid losing the object from the image frame, repeating this procedure before convergence. (b) The DRL successfully found a good starting pose for VS, leading to a very fast and well-behaved convergence.

IV. Conclusion

In this paper, we presented a novel approach for transferring a DRL agent for gripper pose estimation from simulation to a real robot, in a reverse real-to-sim manner, involving a combination of a CycleGAN and VS. We presented the VS task for refining the final grasping pose prior to grasp to address the challenges linked to final gripper pose, reported in literature related to transfer learning from simulation to the real world. Results demonstrate the efficiency of our approach, and the ablation study shows how our approach outperforms selected baselines, achieving a grasping accuracy of $83\,\%$ on previously unseen semi-compliant objects. This is on par with or better than recent state-of-the-art methods [14], [20], [22], [31]. To the best of our knowledge, this work is the first to use a GAN in combination with VS for the transfer of a DRL grasping policy to a real robot.

For future work, we intend to investigate the adaptation of depth images with the CycleGAN, to improve the grasping accuracy of the agent and to explore the combination of our approach with other techniques. Finally, we are interested in investigating a variety of sparse and dense visual features for VS, and to explore various VS strategies that have the potential to mitigate some of the challenges caused by large angular errors.

Acknowledgements

We thank T. Olsen and B. Ottesen for their previous work on training a PPO-agent in simulation, and RCN for the iProcess (255596) and GentleMAN (299757) projects.

References

[1] 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.

[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.

[3] S. Kumra and C. Kanan, "Robotic grasp detection using deep convolutional neural networks," in *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Sep. 2017, pp. 769–776.

[4] L. Pinto and A. Gupta, "Learning to push by grasping: Using multiple tasks for effective learning," in *2017 IEEE International Conference on Robotics and Automation (ICRA)*, May 2017, pp. 2161–2168.

[5] 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.

[6] L. Pinto and A. Gupta, "Supersizing self-supervision: Learning to grasp from 50k tries and 700 robot hours," in *2016 IEEE International Conference on Robotics and Automation (ICRA)*, May 2016, pp. 3406–3413.

[7] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov, "Proximal policy optimization algorithms," 2017. arXiv: 1707.06347 [cs.LG].

[8] V. Mnih, K. Kavukcuoglu, D. Silver, *et al.*, "Human-level control through deep reinforcement learning," *Nature*, vol. 518, no. 7540, p. 529, 2015.

[9] T. P. Lillicrap, J. J. Hunt, A. Pritzel, *et al.*, "Continuous control with deep reinforcement learning," in *4th International Conference on Learning Representations ICLR*, 2016.

[10] J. Schulman, S. Levine, P. Abbeel, M. Jordan, and P. Moritz, "Trust region policy optimization," in *Proceedings of the 32nd International Conference on Machine Learning*, ser. Proceedings of Machine Learning Research, vol. 37, PMLR, 2015, pp. 1889–1897.

[11] Y. Duan, X. Chen, R. Houthooft, J. Schulman, and P. Abbeel, "Benchmarking deep reinforcement learning for continuous control," in *Proceedings of The 33rd International Conference on Machine Learning*, ser. Proceedings of Machine Learning Research, vol. 48, PMLR, Jun. 2016, pp. 1329–1338.

[12] T. Inoue, G. D. Magistris, A. Munawar, T. Yokoya, and R. Tachibana, "Deep reinforcement learning for high precision assembly tasks," in *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2017, pp. 819–825.

[13] A. Taitler and N. Shimkin, "Learning control for air hockey striking using deep reinforcement learning," in *2017 International Conference on Control, Artificial Intelligence, Robotics Optimization (ICCAIRO)*, 2017, pp. 22–27.

[14] J. Mahler, J. Liang, S. Niyaz, *et al.*, "Dex-net 2.0: Deep learning to plan robust grasps with synthetic point clouds and analytic grasp metrics," in *Proceedings of Robotics: Science and Systems*, Jul. 2017.

[15] G. Brunner, Y. Wang, R. Wattenhofer, and S. Zhao, "Symbolic music genre transfer with cyclegan," in *2018 IEEE 30th International Conference on Tools with Artificial Intelligence (ICTAI)*, Nov. 2018, pp. 786–793.

[16] C. Chen, Q. Dou, H. Chen, and P.-A. Heng, "Semantic-aware generative adversarial nets for unsupervised domain adaptation in chest x-ray segmentation," in *Machine Learning in Medical Imaging*, Springer International Publishing, 2018, pp. 143–151, ISBN: 978-3-030-00919-9.

[17] S. James, P. Wohlhart, M. Kalakrishnan, *et al.*, "Sim-to-real via sim-to-sim: Data-efficient robotic grasping via randomized-to-canonical adaptation networks," in *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Jun. 2019.

[18] X. B. Peng, M. Andrychowicz, W. Zaremba, and P. Abbeel, "Sim-to-real transfer of robotic control with dynamics randomization," in *2018 IEEE International Conference on Robotics and Automation (ICRA)*, 2018, pp. 1–8.

[19] 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," in *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Nov. 2017, pp. 23–30.

[20] J. Tobin, L. Biewald, R. Duan, *et al.*, "Domain randomization and generative models for robotic grasping," in *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Oct. 2018, pp. 3482–3489.

[21] J. Matas, S. James, and A. J. Davison, "Sim-to-real reinforcement learning for deformable object manipulation," in *Proceedings of The 2nd Conference on Robot Learning*, ser. Proceedings of Machine Learning Research, vol. 87, PMLR, Oct. 2018, pp. 734–743.

[22] K. Bousmalis, A. Irpan, P. Wohlhart, *et al.*, "Using simulation and domain adaptation to improve efficiency of deep robotic grasping," in *2018 IEEE International Conference on Robotics and Automation (ICRA)*, May 2018, pp. 4243–4250.

[23] F. Zhang, J. Leitner, B. Upcroft, and P. I. Corke, "Vision-based reaching using modular deep networks: From simulation to the real world," 2017. arXiv: 1610.06781v4 [cs.RO].

[24] Z.-W. Hong, Y.-M. Chen, H.-K. Yang, *et al.*, "Virtual-to-real: Learning to control in visual semantic segmentation," in *Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence, IJCAI-18*, Jul. 2018, pp. 4912–4920.

[25] I. Goodfellow, J. Pouget-Abadie, M. Mirza, *et al.*, "Generative adversarial nets," in *Advances in Neural Information Processing Systems 27*, 2014, pp. 2672–2680.

[26] J. Zhu, T. Park, P. Isola, and A. A. Efros, "Unpaired image-to-image translation using cycle-consistent adversarial networks," in *2017 IEEE International Conference on Computer Vision (ICCV)*, Oct. 2017, pp. 2242–2251.

[27] W. Deng, L. Zheng, Q. Ye, G. Kang, Y. Yang, and J. Jiao, "Image-image domain adaptation with preserved self-similarity and domain-dissimilarity for person re-identification," in *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition. Proceedings*, 2018, pp. 994–1003.

[28] P. Isola, J. Zhu, T. Zhou, and A. A. Efros, "Image-to-image translation with conditional adversarial networks," in *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Jul. 2017, pp. 5967–5976.

[29] K. Bousmalis, N. Silberman, D. Dohan, D. Erhan, and D. Krishnan, "Unsupervised pixel-level domain adaptation with generative adversarial networks," in *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Jul. 2017.

[30] F. Chaumette and S. Hutchinson, "Visual servo control. I. basic approaches," *IEEE Robotics Automation Magazine*, vol. 13, no. 4, pp. 82–90, Dec. 2006.

[31] B. Griffin, V. Florence, and J. J. Corso, "Video object segmentation-based visual servo control and object depth estimation on a mobile robot," in *IEEE Winter Conference on Applications of Computer Vision (WACV)*, 2020.

[32] J. Zhang, L. Tai, P. Yun, *et al.*, "Vr-goggles for robots: Real-to-sim domain adaptation for visual control," *IEEE Robotics and Automation Letters*, vol. 4, no. 2, pp. 1148–1155, Apr. 2019.

[33] B. Calli, A. Singh, J. Bruce, *et al.*, "Yale-CMU-Berkeley dataset for robotic manipulation research," *The International Journal of Robotics Research*, vol. 36, no. 3, pp. 261–268, 2017.

[34] P. Z. Ramirez, A. Tonioni, and L. D. Stefano, "Exploiting semantics in adversarial training for image-level domain adaptation," in *2018 IEEE International Conference on Image Processing, Applications and Systems (IPAS)*, Dec. 2018, pp. 49–54.

[35] X. Mao, Q. Li, H. Xie, R. Y. K. L. anand Zhen Wang, and S. P. Smolley, "Least squares generative adversarial networks," in *The IEEE International Conference on Computer Vision (ICCV)*, Oct. 2017.

[36] M. Arjovsky, S. Chintala, and L. Bottou, "Wasserstein generative adversarial networks," in *Proceedings of the 34th International Conference on Machine Learning*, ser. Proceedings of Machine Learning Research, vol. 70, PMLR, 2017, pp. 214–223.

[37] O. Ronneberger, P. Fischer, and T. Brox, "U-net: Convolutional networks for biomedical image segmentation," in *Medical Image Computing and Computer-Assisted Intervention (MICCAI)*, ser. LNCS, vol. 9351, Springer, 2015, pp. 234–241.

[38] E. Marchand, F. Spindler, and F. Chaumette, "Visp for visual servoing: A generic software platform with a wide class of robot control skills," *IEEE Robotic Automation Magazine*, vol. 12, no. 4, pp. 40–52, 2005.

[39] O. Kermorgant and F. Chaumette, "Dealing with constraints in sensor-based robot control," *IEEE Transactions Robotics*, vol. 30, no. 1, pp. 244–257, 2014.