Synthetic Ground Truth for Presegmentation of Known Objects for Effortless Pose Estimation

Frederik Haarslev[a], William Kristian Juel[b], Norbert Krüger[c] and Leon Bodenhagen[d]
SDU Robotics, Maersk Mc-Kinney Molle Institute, University of Southern Denmark, Campusvej 55, 5230 Odense, Denmark

Keywords: Synthetic Ground Truth Generation, Object Segmentation, Pose Estimation.

Abstract: We present a method for generating synthetic ground truth for training segmentation networks for presegmenting point clouds in pose estimation problems. Our method replaces global pose estimation algorithms such as RANSAC which requires manual fine-tuning with a robust CNN, without having to hand-label segmentation masks for the given object. The data is generated by blending cropped images of the objects with arbitrary backgrounds. We test the method in two scenarios, and show that networks trained on the generated data segments the objects with high accuracy, allowing them to be used in a pose estimation pipeline.

1 INTRODUCTION

Pose estimation is an important task in the field of robotics. It is used in tasks where a robot needs to manipulate objects in unknown locations such as in bin picking. Pose estimation is usually solved in three steps (Figure 1a). First, 3D features in the scene are found and matched to the model. Then an initial pose is found with a global method – e.g. feature matching and RANSAC (Fischler and Bolles, 1981). Lastly, the initial pose is optimized using a local pose optimizer – often ICP (Besl and McKay, 1992). This approach can lead to problems since classic global pose estimators usually require a lot of fine tuning, while often still resulting in a lot of false positives.

Researchers have started to look towards deep learning, as a way to mitigate the need for manual fine-tuning, as neural networks are known to be very noise resistant. (Xiang et al., 2017; Do et al., 2018) both have succeeded at replacing the global pose estimation pipeline with a single network, which given a 2D image estimates the 3D bounding boxes and 6D poses of all known objects in the image (Figure 1b). These networks are trained end-to-end, meaning that they train on ground truth where the data consists of an image, and the annotation is semantic labels, bounding boxes and poses for all objects. Such data

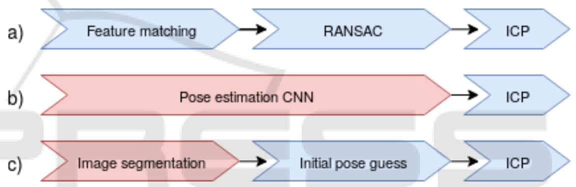


Figure 1: Overview of three different approaches to pose estimation. Blue indicates that the method requires manual fine tuning and red indicates that it relies on training using a ground truth.

can be found in publicly available pose estimation data sets, but it is cumbersome to gather if the object of interest is not part of such a dataset. Since neural networks often require thousands of training examples in order to generalize, it is impractical to hand label enough training data, every time one faces a new pose estimation problem.

Another way deep learning can be used to aid pose estimation, is by only replacing part of the global pose estimation step. (Wong et al., 2017) uses a semantic image segmentation convolutional neural network (CNN) for presegmenting the point cloud. This allows the pose to be estimated using an initial pose guess followed by ICP (Figure 1c). This approach has the benefit of using a generic segmentation network, which can easily be changed to newer models once they are published. The training data is also easier to collect, as only a segmentation mask is needed for each image. However, while the annotations are easier to collect than the pose estimation network counterparts, it is still not practical to create thousands of segmentation masks whenever one needs to estimate

[a] https://orcid.org/0000-0003-2882-0142
[b] https://orcid.org/0000-0001-5046-8558
[c] https://orcid.org/0000-0002-3931-116X
[d] https://orcid.org/0000-0002-8083-0770

Haarslev, F., Juel, W., Krüger, N. and Bodenhagen, L.
Synthetic Ground Truth for Presegmentation of Known Objects for Effortless Pose Estimation.
DOI: 10.5220/0009163904820489
In *Proceedings of the 15th International Joint Conference on Computer Vision, Imaging and Computer Graphics Theory and Applications (VISIGRAPP 2020) - Volume 4: VISAPP*, pages 482-489
ISBN: 978-989-758-402-2; ISSN: 2184-4321

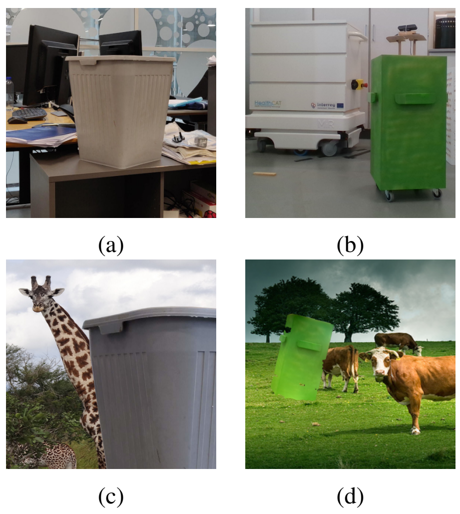


Figure 2: (a) & (b) Examples of two pose estimation scenarios and (c) & (d) their corresponding synthesized training data.

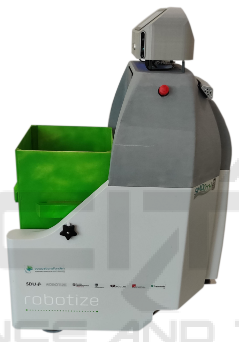


Figure 3: The SMOOTH robot docked with the bin used for use case 1 of the SMOOTH project.

the pose of a new object.

We propose to solve pose estimation by presegmenting the point cloud as in (Wong et al., 2017), but using a network trained on synthetic training data, significantly reducing the manual labor required when estimating the pose of new objects. We exploit the fact that pose estimation problems are concerned with specific objects. This means that the network needs to learn exactly what certain objects looks like, instead of learning broader context dependent object classes such as car and chair. In this work, the data is synthesized by blending images of the objects with arbitrary backgrounds creating images with no apparent context (Figure 2). This way the network learns to ignore the background, and instead focus on the objects. This process reduces the manual labor when using deep learning to solve pose estimation of new objects, from collecting and hand labeling thousands of images, to taking a few images of the objects from different angles.

This is motivated by solving one of the three use cases in the SMOOTH project[1]. In (Juel et al., 2019) the robot itself and the technical modules required to solve the three use cases are theorized and described (Figure 3). In this paper, we focus on the pose estimation method used to solve use case 1 which entails the robot platform detecting, docking, and delivering laundry bins at an elderly care center to ease the heavy work duty for the caregivers. This solution is constrained by the rotational and positional error tolerances of the algorithm used to dock the laundry bin.

2 RELATED WORK

Deep learning is an increasingly pervasive component of modern robotic applications. It is subject to a lot of ongoing research within using object detection and semantic segmentation to robustly crop point clouds of objects, which can be used with the ICP algorithm, as a replacement for RANSAC that requires tedious manual fine-tuning. Extended research also replaces the full pipeline used for classic pose estimation with deep learning methods i.e. learning poses of the detected objects. A common problem for all this research is that a lot of training data is required. Often data is annotated manually, which is a very time-consuming task and does not scale well to the industry. Therefore, various methods for synthesizing training data using deep learning methods and visual simulators (2D and 3D) are common focus points for researchers working with deep learning within a pose estimation pipeline. In the following, selected relevant works are addressed, in particular, related pose estimation where deep learning is used within the pipeline (Section 2.1) and synthetic generation of training data (Section 2.2).

2.1 Deep Learning for Pose Estimation

The research using deep learning within the pose estimation pipeline is motivated by the fact that sophisticated robots still struggle to achieve a fast and reliable perception of task-relevant objects in unconstrained and realistic scenarios. In (Wong et al., 2017) a method called SegICP is proposed – a method for object detection and pose estimation. They train a CNN on 7500 images to do pixel-wise semantic segmentation. They manually label around 5625 objects while the rest is generated using a motion capture system with active markers placed on their cameras and objects. The output of the segmentation is then used to crop the point cloud and thereby allowing pose estimation using multi-hypothesis ICP.

[1]www.smooth-robot.dk

(Xiang et al., 2017) proposes the network PoseCNN for pose estimation. PoseCNN estimates the 3D translation of an object by localizing its center in the image and predicting its distance from the camera. The 3D rotation of the object is estimated by regressing to a quaternion representation.

(Wong et al., 2017; Xiang et al., 2017) replace parts of or the whole classic pose estimation pipe-line. The drawback they have in common is how dependent their methods are on training data i.e. annotation on 2D images and collection of poses for the objects. Creating a data set like this does not scale well to industry since a lot of manual work labeling and producing known poses is required, which prevents its application in real pose estimation tasks, where a known object is to be detected in an arbitrary scene.

2.2 Synthetic Generation of Data

Different approaches like semi-supervised labeling (Russell et al., 2008) and gamification (von Ahn and Dabbish, 2004) have been proposed to ease and streamline the task of annotating data, but a remaining problem these approaches leave the user with is the effort required from humans to manually label or at least supervise the annotation process. Contemporary, approaches that use annotated photo-realistic simulated data to train deep neural networks have shown promising results. In (Johnson-Roberson et al., 2016) a pipeline to gather data from a visual simulator with high realism is established and the data is then used to train deep convolution neural network for object detection. The trained networks, using only the simulated data, were capable of achieving high levels of performance on real-world data. Although this approach gives attractive results, it is limited by the versatility of the visual simulator since it is not possible to generate data of objects and environments not present in it.

Another approach to synthetic ground truth generation uses Generative Adversarial Networks (GAN's) (Goodfellow et al., 2014). One such approach is GeneSIS-RT introduced in (Stein and Roy, 2018). It is a procedure for generating high quality, synthetic data for domain-specific learning tasks, for which annotated training data may not be available. They utilize the CycleGAN algorithm (Zhu et al., 2017) to learn a mapping function G_s between unlabeled real images and unlabeled simulated images. G_s is then used to generate more realistic synthetic training images from labeled simulated images.

While GANs are able to generate data, they still need to be trained. For this a manually annotated ground truth is often required. In the case of GeneSIS-RT no labels are required for the real and simulated image sets. However, in order to generate a ground truth using the method a simulated environment has to be created. A different method described by (Dwibedi et al., 2017) requires even less manual labor than the GAN based methods. They automatically cut object instances and paste them on random backgrounds within the context of the deployment environment. The generated synthetic data gives competitive performance against real data. This method is proved to work when the background images are within the context, requiring the user to take images of the relevant environment that training images can be pasted on.

In this paper, we propose a method where neither human labor nor an external visual simulator is required for synthesizing ground truth data. This data can be used for training a CNN for presegmentation of point clouds in a pose estimation pipeline. We show that the method used in (Dwibedi et al., 2017) works without the use of contextual relevant backgrounds for the synthesized images and without the use of real world images during training. We show that the resulting trained CNN can be used as a link in a pose estimation pipeline. We optimize the CNN so that it can be deployed on a mobile robot by comparing different network backbones and input image sizes. In addition, we show that the developed pose estimation pipeline satisfies the rotational and positional error tolerances required by the use case.

3 METHODS

In this work we try to solve image segmentation of specific objects using as little human labour as possible. This is accomplished by synthesizing the training data and labels, removing the need to do do any annotation by hand. The following subsections will explain how the training data is synthesized, and how the trained segmentation network is used for estimating the pose of the object in the scene.

3.1 Generation of Training Data

The flow of the data generation pipeline is shown in Figure 4. The first step is to take a few images of the desired objects from different angles and removing the background. The images have their background removed automatically by utilizing the depth channel of the RGB-D camera. By comparing images of the object in a scene with images of the scene without the object, the object mask can be determined as the pixels which has been changed between the two images.

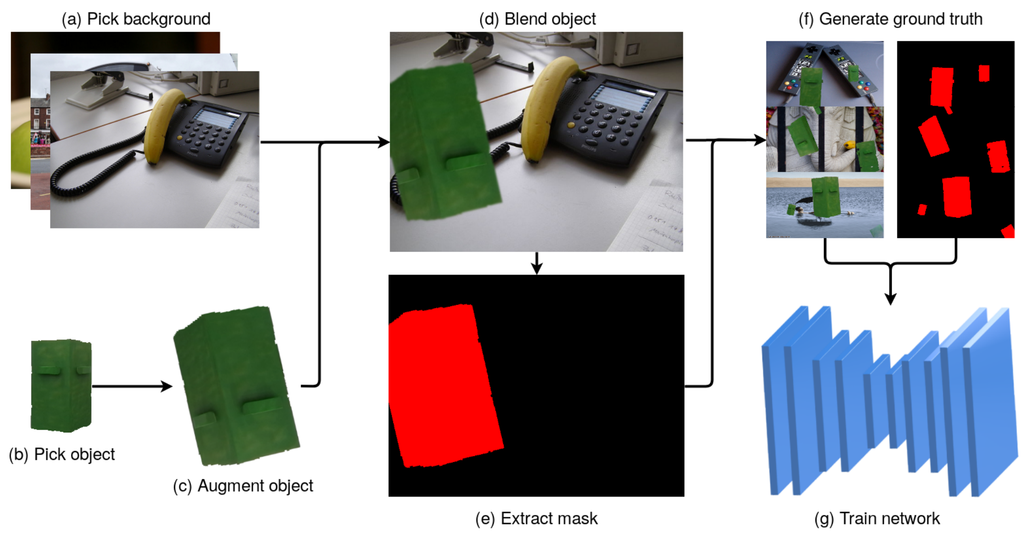


Figure 4: Flow of training data generation. Images of the object are automatically cropped and blended with an arbitrary background. Changed pixels are given the object label.

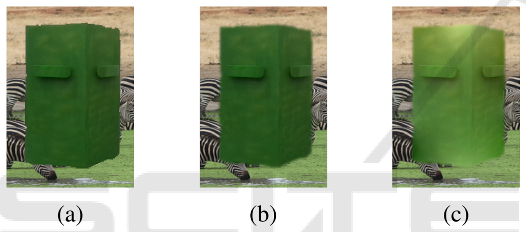


Figure 5: Three methods for blending an object to the background: (a) no blending, (b) Gaussian blurring, and (c) seamless cloning.

The synthetic images are generated by blending the cropped object images with an arbitrary background (Figure 4a). In this work the MS COCO (Lin et al., 2014) dataset has been used for the background images. First, an object image is generated by picking one of the cropped images (Figure 4b) and then augmenting it (Figure 4c). Such augmentations includes flipping, scaling, and rotating. The augmented object is then blended with the background at a random location (Figure 4d). As in (Dwibedi et al., 2017), two different approaches for blending are used: Gaussian blurring of the object edges, or the seamless clone algorithm described in (Pérez et al., 2003). The results of both methods are compared to not using blending in Figure 5.

The Gaussian blurring method blends the object by using a Gaussian filter on the alpha mask of the object image. This removes the hard edges between the object and background by fading the images together. The seamless clone algorithm minimizes the difference in the gradient field of the object image and the desired gradient field of the affected area of the background image, by solving a set of Poisson equations. This smooths the transition between the object and background, by removing any abrupt changes in the gradient field of the resulting image. Another result of the Poisson editing is variations in the lighting of the object, dependent on the background. The seamless clone algorithm therefore inherently performs data augmentation, leading to greater variation in the synthetic data.

Once an object has been blended with the background the segmentation mask can be created as the pixels which has changed after the blending (fig. 4e). For each synthetic image N random object images are blended with the background (fig. 4f). In order to better handle cases where the object is not in the image, a percentage of the synthesized training data contains only the random background. In this work this percentage is set to 5%. As the position of each object image is random some occlusion of the objects occurs, leading to the trained network being able to handle such occurrences. In (Dwibedi et al., 2017) every image they synthesize is generated twice, once with either blending method. I.e. the ground truth contains pairs of images with the same objects blended at the same locations on the same background, with the only difference between them being the blending method. This is stated to help the network ignore learning the blending artifacts, instead learning the actual shape of the object. We did not find this to improve the segmentation performance, so instead the blending method is chosen randomly per object: I.e., a single synthetic image can contain both blending methods on different objects.

Another method used by (Dwibedi et al., 2017) is the inclusion of distractor objects. Distractor objects

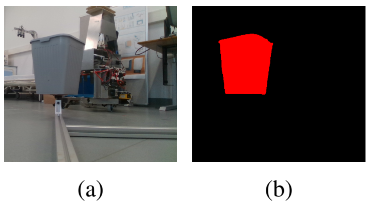


Figure 6: Output of the segmentation network trained on the grey bin.

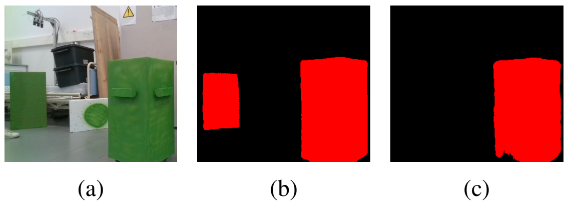


Figure 7: Output of the segmentation network trained on the green bin (b) without and (c) with the decoy as distractor.

are other objects than the objects of interest which are labeled as background. This is stated to also help the network learn to ignore the blending artifacts. We did not find this to improve the segmentation performance, so instead distractor objects are used to mitigate false positives, as explained in Section 3.2.

3.2 Training Network

Before the ground truth could be synthesized, 300 images of the bin are taken using the depth channel to generate a segmentation mask as well as 100 images not containing the bin. These were split into a validation and test set containing 150 object images and 50 background images each. Of the 150 validation object images, 100 were cropped with the segmentation mask and used for synthesizing the training data.

Since only a single object is being learnt, the capacity of the segmentation network does not need to be large. Instead the network is chosen based on fast inference speed and low memory footprint, in order to optimize the performance on embedded platforms like mobile robots. Therefore, the BiSeNet (Yu et al., 2018) real-time segmentation network has been implemented. The network was trained on a ground truth consisting of 5,000 synthesized images for 10 epochs, using the Adam optimizer with 0.001 as the initial learning rate. The dataset was balanced by weighting the loss based on the occurrence of each class, thereby reducing the effect of the background class being abundant in the training data.

In order to validate the method, a network was first trained on the generic grey garbage bin. An example of the network output can be seen on Figure 6. This shows that the network is able to learn to segment an

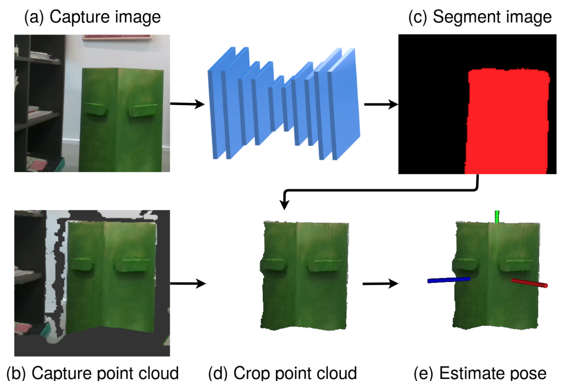


Figure 8: Flow of pose estimation. Point cloud is cropped using the output of the segmentation network, allowing the pose to be found using only ICP.

object correctly using our method of synthetic ground truth generation, and therefore the method was used with the use case specific green bin. In order to verify that the network learns to segment the object – and not just the color green – it was tested on images containing other objects in the same shade of green (Figure 7). The network correctly segments the green circle as background, but mislabels the green rectangle (decoy) which is made to look like the bin. This is remedied by taking 3 images of the decoy, and then synthesizing a new ground truth using the decoy as distractor. This equates to testing the trained network in the deployment environment, and then taking a few images of any false positives.

3.3 Estimating Pose

The flow of the pose estimation pipeline is shown in Figure 8. First step is to capture the 2D scene image and corresponding 3D point cloud. The image is fed through the trained network to produce a segmentation mask. For RGB-D sensors there is a one-to-one correspondence between the pixel locations in the segmentation mask, and the points in the ordered point cloud. This allows for the extraction of all points of the desired class using the segmentation mask. However, the registration is not perfect, meaning that some bin pixels are registered to 3D locations in the background. This creates some systematic artifacts which the network cannot account for, since it only works with the 2D data. Because of this, all points further away than some margin (depending on object size) from the centroid of the segmented point cloud are also removed.

The pose can now be estimated using an initial pose guess followed by ICP. In the use case of this paper, the bin is always positioned upright on the floor. This reduces the pose estimation to 3 Degrees of Free-

Table 1: Intersection over Union (IoU) results and Frames Per Second (FPS) of the segmentation network with varying backbones and input resolution. The mean IoU is calculated over 5 networks with the same hyper-parameters, and the standard deviation is indicated with $\pm$.

Backbone	Mean IoU	FPS
Xception	$\mathbf{0.894 \pm 0.005}$	41.4
MobileNetV2	0.870 ± 0.026	**61.7**
DenseNet121	0.872 ± 0.025	41.9
ResNet50	0.804 ± 0.029	40.5
Input resolution	**Mean IoU**	**FPS**
$\mathbf{288 \times 160}$	0.855 ± 0.018	**141.5**
384×224	0.847 ± 0.025	95.9
512×288	$\mathbf{0.870 \pm 0.026}$	61.7
672×384	0.827 ± 0.004	41.5
896×512	0.779 ± 0.070	24.6

dom (DoF). In addition to this the object is four-way symmetrical. This means that the object model can be placed in the centroid of the cropped cloud as the initial pose guess. Afterwards the pose is found using a three-fold coarse-to-fine ICP scheme. First the scene and model clouds are heavily downsampled using a voxelgrid with leaf size depending on the object size and desired speed. Then ICP is used with a correspondence distance double the leaf size to find an initial transformation. This is repeated twice using the previously found transformation as the initial guess, halving the leaf size and correspondence distance at each step. The final transformation is then an estimation of the pose of the model in the scene.

In case of 6-DoF pose estimation problems with unsymmetrical objects, the multi-hypothesis pose estimation scheme presented in (Wong et al., 2017) can be used. They replace the initial pose guess with multiple random guesses with the same centroid but random rotation. ICP is then used with all guesses, and the best pose is found using a pose evaluation metric.

4 RESULTS

The first step in creating the pose estimation pipeline is the training of the segmentation network. It is desired that the network accurately segments the object, while being lightweight and having a high throughput. Therefore, various network backbones and input resolutions are tested. For each backbone 5 networks are trained on the green bin synthetic ground truth at 512×288 input resolution. The trained networks are used on the test set consisting of 150 images of the bin labeled using the depth channel, and 50 images without the bin. Many of the test images contain the green decoy object and other green objects. The Intersection over Union (IoU) between the segmentation output and ground truth is calculated. The mean IoU, standard deviation and Frames Per Second (FPS) is calculated for the 5 networks using the same backbone, for every backbone. All backbones are pretrained on ImageNet (Russakovsky et al., 2015). These results can be seen in Table 1. For further investigation, we decided to continue with the MobileNet (Howard et al., 2017) implementation due to the high mean IoU and a significantly higher FPS compared to using the other backbones. Since this must run on a mobile robot the computational time is more important than a few percent higher IoU.

Besides the network backbone, the input resolution was also tested. Similar to before, 5 networks using the MobileNet backbone are trained for each input resolution, and the mean IoU, standard deviation and FPS is calculated (Table 1). One result which stands out is the networks trained at 896×512, which leads to low mean IoU and high variance. An explanation for this could be that the pixel quantization hides the blending artifacts in lower resolution images, leading to a larger mismatch between the synthetic training data and real test data at higher resolutions. Similar to the choice of backbone, the gain in lightweightness and inference speed when using 288×160 images, outweighs the slight gain in mean IoU when using 512×288 images. Therefore, the network chosen for the pose estimation tests was the network scoring the highest IoU of the 5 networks with 288×160 input resolution and MobileNet backbone.

The accuracy of the pose estimation pipeline is tested using a dataset with known ground truth poses. The ground truth poses are found using detachable AR markers (Figure 9). The markers needs to be detachable, since they should not be visible when segmenting the image. The detachable markers are calibrated to the center of the bin using two calibration markers. Each marker aligns with a side of the square bin, allowing the center to be calculated by finding the intersection between the pairs of marker planes, and then averaging them. This transformation is calculated for 250 positions of the bin, after which the calibration markers are removed.

The test data is made by placing the bin randomly in a different setting than the one used for capturing the object images which the network was trained on. The ground truth pose is then found by averaging the marker poses over 50 frames. The markers are then removed and 25 image/point cloud sets are recorded. This is repeated 150 times with different bin placements resulting in 3750 image/point cloud pairs. A

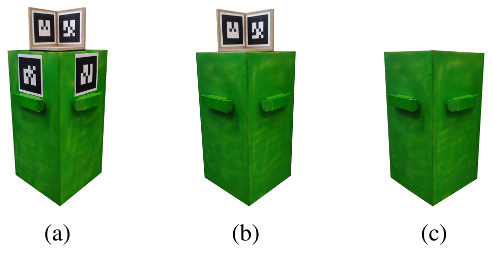
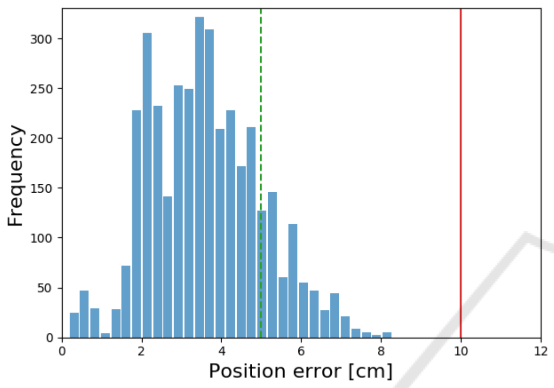


Figure 9: The markers used for the pose estimation ground truth. (a) shows the bin with the calibration and detachable markers, (b) shows the bin with the detachable markers, and (c) shows the bin with no markers.



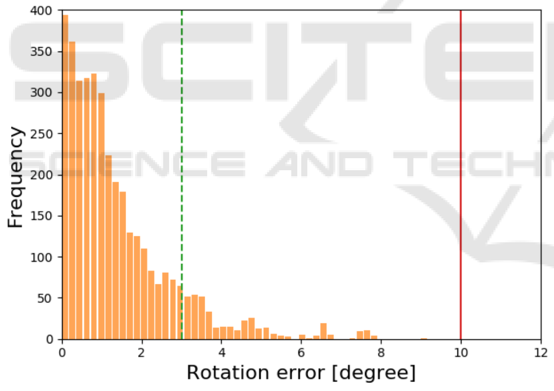


Figure 10: Histograms of the position and rotation errors in the estimated poses on the test set. Errors larger than the red lines will result in a failed docking, while errors lower than the green lines always results in a successful docking.

pose is estimated for each pair using the described pose estimation method.

In the use case the laundry bin is restricted to 3-DoF. I.e, it is limited to an upright position with the wheels on the ground so that it can be picked up by the robot. Therefore, ground truth and estimated poses are projected to an (x,y) position along the ground plane, and a θ rotation around the normal vector to the ground plane. A positional error is calculated as the L2-norm between the ground truth and estimated positions, and a rotational error as the difference between the ground truth and estimated rotations. Because of the four-way symmetrical property of the bin, the estimated pose can be rotated 90° or 180° from the ground truth and still be correct. Therefore, 90° is subtracted from the rotation error until it lies between -45° and 45°, where after the absolute value is found resulting in a rotation error between 0° and 45°. Before testing the pose estimation accuracy, the maximum rotational and positional errors which the docking algorithm used in the use case can handle was found. This was done by placing the bin with a random pose behind the robot, and trying to dock with it. The test was repeated with various poses in order to find a threshold where the docking would always succeed, and a threshold where it would always fail. Figure 10 shows histograms of the position and rotation errors on the test set. The red lines represents the errors over which the docking will fail, and the green lines represents the errors under which the docking is guaranteed to succeed. Most estimates lies below the green lines while no estimates lies above the red lines. It can be seen that the pose estimation has a position bias of about 2 cm. This could be explained by the uncertainty in the marker calibration. This is not present in the rotation errors, since the ground truth rotation is not calibrated, but instead is the average orientation of the two detachable markers. Figure 11 shows results from the segmentation and pose estimation of both the grey and green bins.

5 CONCLUSION

We present a method to effortlessly solve pose estimation of known objects without prior knowledge about the context of the object, by presegmenting the point cloud using a segmentation network. The network is trained on a synthetic ground truth, created by taking a small number of images of the object of interest. The object images are blended with an arbitrary background from MS COCO. We show that the method works on two different objects, a generic garbage bin and a use case specific bin. When the two objects is placed in a real-world context the output of the trained segmentation network is robust enough to crop the point cloud such that it can be used as input to the ICP algorithm for pose estimation. We train the network using different backbones and input resolutions, and show how changing the network affects the intersection over union between the ground truth and segmentation output.

We test the effect of varying the backbone and input size of the segmentation network, and find that the best compromise for our use case is using MobileNet and 288 x 160 image size, since the network is to be

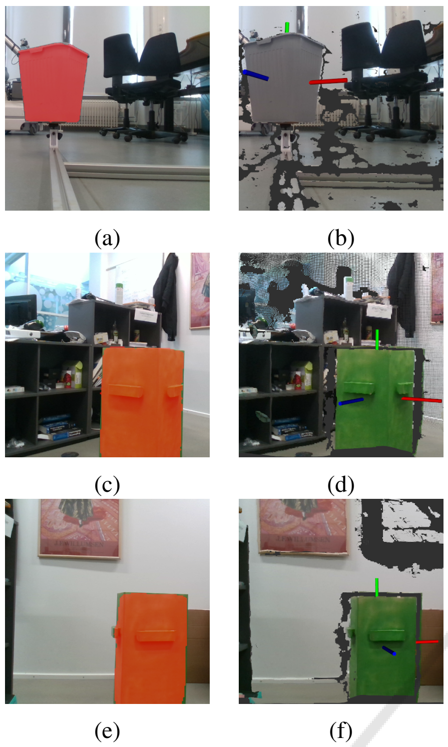


Figure 11: Outputs of the segmentation networks (left column) and corresponding estimated poses (right column) in the scene clouds.

deployed on a mobile platform. We show that the pose of the object can be estimated using the centroid of the presegmented cloud as an initial pose guess followed by ICP, thereby not having to rely on a more sophisticated pose estimation algorithm. We show that the accuracy of the resulting pose estimation is within the constraints of the use case.

REFERENCES

Besl, P. J. and McKay, N. D. (1992). A method for registration of 3-d shapes. *IEEE Transactions on Pattern Analysis and Machine Intelligence*.

Do, T.-T., Pham, T., Cai, M., and Reid, I. D. (2018). Lienet: Real-time monocular object instance 6d pose estimation. In *BMVC*.

Dwibedi, D., Misra, I., and Hebert, M. (2017). Cut, paste and learn: Surprisingly easy synthesis for instance detection. *CoRR*, abs/1708.01642.

Fischler, M. A. and Bolles, R. C. (1981). Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography. *Commun. ACM*.

Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., and Bengio, Y. (2014). Generative adversarial networks. *Advances in Neural Information Processing Systems*.

Howard, A. G., Zhu, M., Chen, B., Kalenichenko, D., Wang, W., Weyand, T., Andreetto, M., and Adam, H. (2017). Mobilenets: Efficient convolutional neural networks for mobile vision applications. *CoRR*, abs/1704.04861.

Johnson-Roberson, M., Barto, C., Mehta, R., Nittur Sridhar, S., and Vasudevan, R. (2016). Driving in the matrix: Can virtual worlds replace human-generated annotations for real world tasks? *CoRR*, abs/1610.01983.

Juel, W. K., Haarslev, F., Ramírez, E. R., Marchetti, E., Fischer, K., Shaikh, D., Manoonpong, P., Hauch, C., Bodenhagen, L., and Krüger, N. (2019). Smooth robot: Design for a novel modular welfare robot. *Journal of Intelligent & Robotic Systems*.

Lin, T., Maire, M., Belongie, S. J., Bourdev, L. D., Girshick, R. B., Hays, J., Perona, P., Ramanan, D., Dollár, P., and Zitnick, C. L. (2014). Microsoft COCO: common objects in context. *CoRR*, abs/1405.0312.

Pérez, P., Gangnet, M., and Blake, A. (2003). Poisson image editing. *ACM Trans. Graph.*

Russakovsky, O., Deng, J., Su, H., Krause, J., Satheesh, S., Ma, S., Huang, Z., Karpathy, A., Khosla, A., Bernstein, M., Berg, A. C., and Fei-Fei, L. (2015). Imagenet large scale visual recognition challenge. *International Journal of Computer Vision*.

Russell, B. C., Torralba, A., Murphy, K. P., and Freeman, W. T. (2008). Labelme: A database and web-based tool for image annotation. *Int. Journal of Computer Vision*.

Stein, G. J. and Roy, N. (2018). Genesis-rt: Generating synthetic images for training secondary real-world tasks. *Int. Conf. on Robotics and Automation (ICRA)*.

von Ahn, L. and Dabbish, L. (2004). Labeling images with a computer game. *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*.

Wong, J. M., Kee, V., Le, T., Wagner, S., Mariottini, G., Schneider, A., Hamilton, L., Chipalkatty, R., Hebert, M., Johnson, D. M. S., Wu, J., Zhou, B., and Torralba, A. (2017). Segicp: Integrated deep semantic segmentation and pose estimation. In *Int. Conf, on Intelligent Robots and Systems (IROS)*.

Xiang, Y., Schmidt, T., Narayanan, V., and Fox, D. (2017). Posecnn: A convolutional neural network for 6d object pose estimation in cluttered scenes. *CoRR*, abs/1711.00199.

Yu, C., Wang, J., Peng, C., Gao, C., Yu, G., and Sang, N. (2018). Bisenet: Bilateral segmentation network for real-time semantic segmentation. *CoRR*, abs/1808.00897.

Zhu, J.-Y., Park, T., Isola, P., and Efros, A. A. (2017). Unpaired image-to-image translation using cycle-consistent adversarial networks. *Int. Conf. on Computer Vision (ICCV)*.