TUM School of Computation, Information and Technology

Scalable Learning of 6-DoF Object and Robotic Grasp Poses

Martin Bernd Sundermeyer

Vollständiger Abdruck der von der TUM School of Computation, Information and Technology der Technischen Universität München zur Erlangung eines

Doktors der Naturwissenschaften (Dr. rer. nat.)

genehmigten Dissertation.

Vorsitz: **Prof. Dr. Eckehard Steinbach**

Prüfende der Dissertation:
1. Prof. Dr. Rudolph Triebel
2. Prof. Dr. Daniel Cremers
3. Prof. Dr. Vincent Lepetit

Die Dissertation wurde am 07.02.2024 bei der Technischen Universität München eingereicht und durch die TUM School of Computation, Information and Technology am 02.10.2024 angenommen.

Acknowledgments

A PhD resembles a long road that is not always straight but has amazing views along the way. I was fortunate enough to go on this road-trip with my fellow colleagues, students and friends which helped me driving and writing through the darker nights.

I want to especially thank Rudolph for giving me the opportunity, freedom and support to pursue this thesis at DLR. Thank you, Zoltan, for the early support by supervising my master thesis and first publications, I could learn a lot from you. Thank you, Max Du., for being always there with a positive attitude, I enjoyed working under and with you from being your first student to being in your department group. I am sure your passion for robots will open their eyes!

Thank you, Max De., for sharing so many good and bad moments from the master thesis to our shared office. I loved your passion and at the same time enjoyed the best PhD vents I got to experience. I learned a lot developing BlenderProc together with you and your student army, especially with Dominik. Thank you, Manuel, for the great collaborations towards the end of my PhD and best of luck with your new endeavors.

Thank you, Wout, and Sebastian for being such great students and colleagues, it's great to see how you are continuing the research and completely integrated yourself into the RMC laboratory. Also thanks to my students Philipp, Ahsan, Amrutha and Jakob!

I want to thank my family for paving the way for so many years and for being a role model in persevering through obstacles. And finally a special thanks to Elli for the love, understanding and mental support, also in more difficult periods.

Abstract

This dissertation addresses the problem of 6-DoF Object Pose and 6-DoF Grasp Pose estimation from visual sensor data, which is crucial for tasks such as robotic manipulation and Augmented Reality. We present novel learning-based methods that are fast, reliable, and scalable concerning training data, test environments, and target objects. Instead of relying on real pose annotated data, we train our models in simulation which provides an abundant source of variably steerable data with exact 3D annotations.

Our first contribution, the Augmented Autoencoder, integrates sim2real transfer techniques and resolves pose ambiguities by training appearance-based encodings without explicit pose supervision. The approach, which won the Best Paper Award at ECCV 2018, achieves real-time 6-DoF object pose estimation on embedded hardware. Our second contribution, the Multi-Path Encoder, extends Augmented Autoencoders to scale efficiently across multiple objects and estimates the poses of objects unseen during training, while maintaining a small memory and inference time footprint. Next, we analyze the results of the Benchmark for 6D Object Pose Estimation (BOP Challenge) which we organize to produce fair comparisons between pose estimation methods under realistic conditions. We address the limiting synthetic-to-real domain gap by introducing BlenderProc, an open-source, procedural, physically-based rendering tool that significantly advances synthetic data realism and transfer to real sensor data. In the BOP Challenge methods trained on our provided BlenderProc data achieved up to 25.5% improvement over the same amount of rasterized training data. We further extend BlenderProc to create the Robot Tracking Benchmark (RTB) for evaluating 6-DoF articulated object tracking in complex scenarios. Subsequently, we present Contact-GraspNet, an end-to-end network for efficient 6-DoF grasp generation from a single depth view, achieving over 90% success rates in unknown object grasping tasks. Finally, we demonstrate the practical utility of our methods through the deployment on robotic systems, including humanoid robots, industrial arms and mobile robots such as an assistive wheelchair.

Kurzfassung

Diese Dissertation befasst sich mit dem Problem der 6-DoF Objektposen- und 6-DoF Greifposen-Schätzung aus visuellen Sensordaten, welche entscheidend sind für Anwendungen wie autonome robotische Manipulation und Augmented Reality. Wir präsentieren neue, lernbasierte Methoden, die schnell, zuverlässig und skalierbar in Bezug auf Trainingsdaten, Testumgebungen und Anzahl der Zielobjekte sind. Anstatt auf realen, posen-annotierten Daten, trainieren wir unsere Modelle in Simulationen, aus denen vielfältige Daten mit genauen Annotationen synthetisiert werden können.

Unser erster Beitrag, der Augmented Autoencoder, integriert Sim2Real-Transfer-Techniken und löst Posen-Ambiguitäten, indem er auf erscheinungsbasierten Kodierungen ohne explizite Posen-Supervision trainiert. Dieser Ansatz, der den Best Paper Award auf der ECCV 2018 gewann, ermöglicht eine 6-DoF-Objektposen Schätzung in Echtzeit auf eingebetteter Hardware. Unser zweiter Beitrag, Multi-Path Encoder, erweitert Augmented Autoencoders, um effizient auf viele Objekte zu skalieren und die Posen von Objekten zu schätzen, die zur Trainingszeit nicht bekannt waren während gleichzeitig Speicherressourcen geschont werden. Als nächstes analysieren wir die Ergebnisse des Benchmark for 6D Object Pose Estimation (BOP Challenge), der standardisierte Vergleiche unter realistischen Bedingungen durchführt und in Verbindung mit den "Recovering 6D Pose" Workshops organisiert wird. Wir adressieren die Domänenlücke zwischen synthetischen und echten Daten, indem wir BlenderProc einführen, ein Open-Source Tool für prozedurales Rendering, das die Realitätsnähe synthetischer Daten erheblich verbessert. In der BOP Challenge erzielten auf BlenderProc-Daten trainierte Methoden eine Verbesserung von bis zu 25,5% gegenüber der gleichen Menge an gerasterten, synthetischen Trainingsdaten. Wir entwickeln BlenderProc weiter, um das Robot Tracking Benchmark (RTB) für die Evaluierung von 6-DoF-Objektverfolgung in komplexen Szenarien zu schaffen und stellen eine effiziente Baselinemethode zur Verfügung. Wir präsentieren Contact-GraspNet, ein Ende-zu-Ende trainiertes Netzwerk für effiziente 6-DoF-Greifvorhersagen aus einzelnen

Tiefenansichten, das über 90% Erfolg bei unbekannten Objektgreifaufgaben erzielt. Abschließend demonstrieren wir den praktischen Nutzen unserer Methoden auf Robotersystemen, einschließlich humanoider Roboter, Industriearme und mobiler Roboter wie beispielsweise einem Assistenzrollstuhl.

Contents

1 Introduction

Robotic systems that can autonomously perceive and act in complex environments have the potential to solve important problems of mankind. They could automate repetitive or dangerous tasks currently performed by humans, assist elderly and disabled people in their households, enable driver-less transport, revolutionize medical surgery and allow for extraplanetary colonization. Especially in our aging society automating physical tasks can avoid accidents, reduce human workload and thus increase available time for human interactions. But, to realize these goals, we not only require robust robotic hardware, planning and control but also substantial progress in robotic perception.

In practice, autonomous robots need to perceive the open, physical world from on-board sensor streams. They face unique challenges such as power constraints, sensor disturbances and constantly changing environments. Nevertheless, we already see the adoption of several semi-autonomous consumer products such as robot vacuums and robot lawn mowers. These affordable robots can create accurate maps of their surroundings and navigate through them while bypassing obstacles. For their navigation capabilities, a simple 2D LiDAR stream that is geometrically fused to a map is typically sufficient.

Autonomous vehicles (AVs) demand a much finer visual understanding to navigate within dynamic environments. Yet, their primary goal is similar, i.e. avoiding obstacles at any cost which they already do similarly well as human drivers [211].

Despite this progress, many impactful robotic applications are still out of reach since they would require autonomous robots to not only avoid objects but to physically interact with them based on vision and tactile feedback. Current robotic systems struggle greatly with autonomous manipulation tasks, in particular when objects and the environment are not fixed. Guaranteeing the behavior of complex physical interactions fully in advance is extremely challenging. At the same time, failure cases can have serious consequences for both the robot itself and its surroundings, especially if humans are involved. While a simple model of

Figure 1.1: Humanoid robot David empties a dishwasher based on object detection [225], 6-DoF pose estimation [181] and refinement/tracking [214].

the robot often exists, our knowledge of the environment and the manipulated objects is usually limited and needs to be inferred from noisy, partial observations. Traditional perception pipelines based on geometric matching are often unable to handle those demanding settings because they lack the required accuracy, robustness, generality, efficiency and scalability. These limitations motivate the goal of this thesis to develop and analyze new methods that could fulfill these requirements and thus enable robotic manipulation and related applications.

The last ten years have brought significant advances in the field of machine learning and its application to computer vision [41]. This progress is primarily driven by the development of artificial neural networks that allow learning complex representations from large, high-dimensional datasets. But despite the advances in the semantic analysis of large datasets, many robotic perception problems are still open.

Besides semantic, image-level understanding, robots require a rich spatial, object-centric understanding within their 6-DoF workspace to tackle autonomous manipulation tasks. However, it is challenging to directly apply common learning-based methods since annotating sufficient robotic sensory data with 6-DoF information is expensive and usually performed in controlled lab environments [93]. Con-

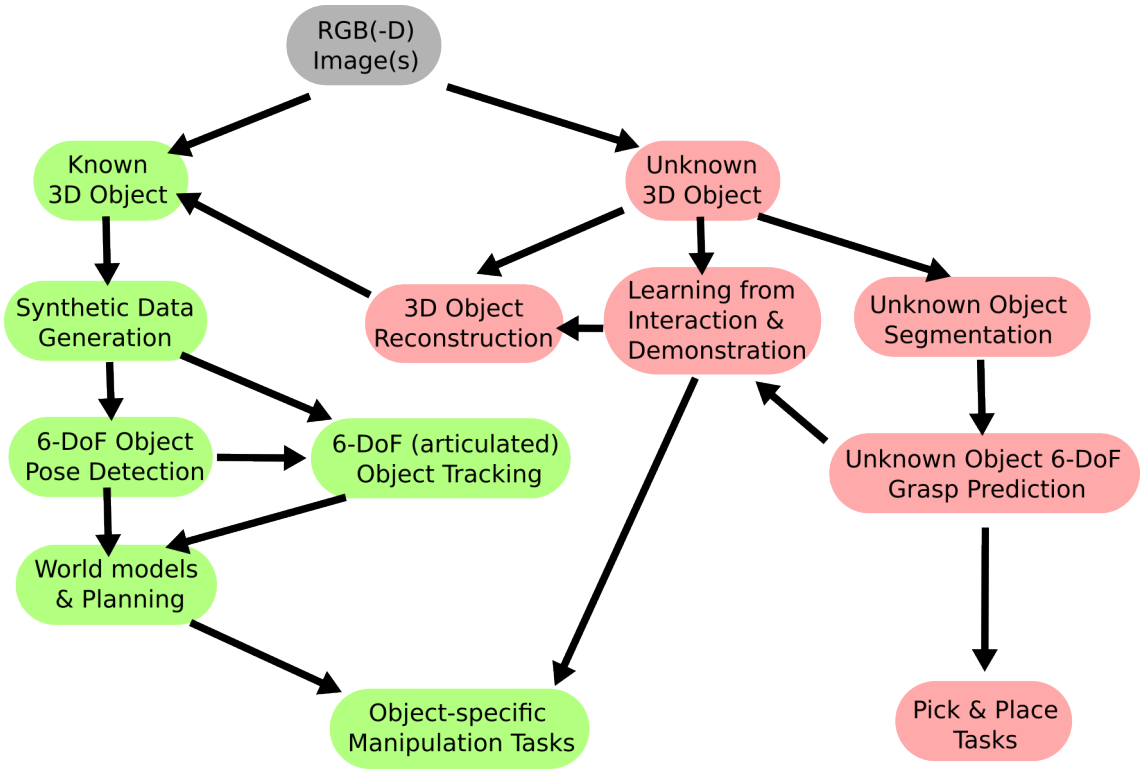


Figure 1.2: Taxonomy of object-centric perception tasks relevant to robotic manipulation. Given a sensor image or stream of RGB(-D) image(s) (grey) and either a 3D model (green) or no 3D model (red), we propose different perception modules in this thesis that enable robots to interact with objects.

sequently, the amount and variation of the training data is often insufficient to generalize to novel environments where i.i.d. assumptions are violated. Still, the majority of proposed methods in the computer vision literature is disconnected from the lack of labeled data as they only evaluate on closely related, 6-DoF annotated train and test sets under narrowly defined conditions [40].

Instead of labeling real data, arbitrary amounts of 6-DoF annotated training data can be generated from 3D object meshes in a procedural simulation. 3D object meshes are ubiquitous in manufacturing environments or can be 3D reconstructed from image collections [83]. By rendering these meshes from virtual viewpoints we can scale up perfectly annotated datasets to sizes where machine learning algorithms excel. Consequently, the new challenge becomes to create sufficiently realistic and diverse data such that the learning algorithm trained in simulation generalizes to real scenarios. Throughout this thesis we will develop and analyze techniques to improve transfer from synthetic to real domains such as domain randomization and physically-based simulation.

1.1 Problem Formulation

Two of the most elementary perception problems in robotic manipulation are **6-DoF Object Pose Estimation** and **6-DoF Grasp Pose Generation**. These tasks are the main applications studied in this thesis and are summarized in the following. An introduction into the underlying fundamentals is given in Chapter 2.

1.1.1 6-DoF Object Pose Estimation

In 6-DoF [1]Object Pose Estimation the transformation between a sensor and a known object is estimated, given a single sensory image and its intrinsics. While numerous applications exist, it is crucial in autonomous robotic manipulation for tasks like assembly, planning and task-oriented grasping. Most methods that estimate absolute poses require a 3D model of the object instance or at least 3D models of other instances within the same category. Given that the sensor is calibrated to the robot, the 3D models can then be placed into a world model at the estimated pose which allows reasoning about manipulation tasks. In contrast, relative or few-shot 6-DoF object pose estimation is a subfield where the relative transformation between two or more views of the same object is sought. Here, 3D models are not necessarily required, and the methods can be quickly adapted to novel objects. While this relative formulation is less compatible to world models and traditional motion planning algorithms, it circumvents the effort of designing or reconstructing a 3D model. The appropriate formulation always depends on the requirements and prerequisites of a given application which will be discussed throughout this thesis.

1.1.2 6-DoF Grasp Generation

In vision-based 6-DoF Grasp Generation we estimate a distribution of stable 6-DoF grasp poses and gripper configurations over an object or a scene, given a single input view as well as 3D models of the robot and gripper. The gripper configuration for parallel-jaw grippers is simply the grasp width, while for robotic hands it is the full finger configuration. In contrast to object pose estimation, grasps can be estimated on completely unknown objects. This is crucial in

[1]Note: 6-DoF and 6D refer to a pose in SE(3) and are used interchangeably throughout this thesis.

scenarios where the robot must act in a more unconstrained setting, e.g. for pick and place, packaging, bin picking or delivery tasks. Grasps can also be conditioned on specific tasks or objects to perform functional manipulation. In this context we also discuss how to generalize demonstrated tasks to novel views and environments.

1.2 Challenges and Existing Limitations

Both tasks constitute unique challenges but are also highly related. Existing algorithms often lack robustness against occlusion and background clutter. They fail if the environment and lighting conditions change. Saturation, noise and missing values in color and depth sensors impede robust estimations – further aggravated by reflective, transparent or light absorbing materials. There are challenging objects that lack a distinctive texture or shape where the features extracted by traditional computer vision methods such as edges [25], color histograms [105] or point correspondences [11] cannot disambiguate hypotheses in the large 6-DoF prediction space.

To tackle these limitations learning-based methods have been developed to extract more robust, descriptive and thus distinctive features [77, 107]. However, the learning process itself is non-trivial and creates many new challenges. For both object and grasp pose estimation, an optimizable output representation needs to be found that does not suffer from the curse of dimensionality and still covers the full 6-DoF space.

6-DoF object pose estimation methods specifically suffer from object or view symmetries that introduce pose ambiguities. Two identical views can have different pose annotations assigned which can harm the common uni-modal supervised learning schemes. Therefore, current algorithms often require certain object properties such as enough textural surface structure or an asymmetric shape to avoid ambiguities and therefore ensure convergence.

6-DoF grasping of unknown objects from a partial 3D view is challenging because (1) it is necessary to separate unknown objects from each other, (2) unobservable regions need to be inferred from the visible ones which is often ambiguous, (3) the stability of grasps needs to be predicted with limited knowledge about material properties, (4) sufficient grasp coverage is required to ensure the existence of kinematically feasible and non-colliding approach trajectories in cluttered scenes.

Separating unknown objects and inferring their invisible parts $(1 + 2)$ can benefit from data-driven methods that can reason about what constitutes an object and how its geometry completes in the unobserved regions. To learn stable grasps with sufficient coverage $(3 + 4)$, we can either rely on data from heuristics, physics simulations or robotic interactions in the order of increasing quality and cost. However, even if dense and high quality grasp labels can be obtained, it is still open how to constrain the distribution of 6-DoF grasps plus gripper width to a learnable output representation. Since simple multi-target regression fails for such large, imbalanced output spaces, we will introduce new techniques to reduce the dimensionality and imbalance of this problem.

To leverage learning-based methods for these tasks, the lack of available annotated training data and resulting over-fitting issues must be overcome. There are two major strategies attempting to solve this issue:

1. Train on real world data with no, few or automatically obtained labels.
2. Simulate realistic and diverse 6-DoF annotated data and train with domain randomization/adaptation to ensure sim2real transfer.

The main challenge of weakly supervised real world training is to obtain sufficient, relevant and diverse data that generalizes outside the controlled training domain. Even if labels can be generated automatically, the required supervision, wear and interaction time of the robotic system can still be a major hurdle that needs to be overcome.

On the other hand, training in simulation requires to faithfully replicate the real world while also covering its corner cases. Furthermore, the trained models need to become invariant against the sim2real gap. The factors for successful sim2real transfer are generally understudied, but crucial for the final performance.

1.3 Chronological List of Publications

[124] **M. Sundermeyer**, Z.-C. Marton, M. Durner, M. Brucker, and R. Triebel. "Implicit 3D Orientation Learning for 6D Object Detection from RGB Images". In: *European Conference on Computer Vision (ECCV)*. 2018.

[125] **M. Sundermeyer**, E. Y. Puang, Z.-C. Marton, M. Durner, and R. Triebel. "Learning Implicit Representations of 3D Object Orientations from RGB". In: *International Conference on Robotics and Automation (ICRA) Workshops*. 2018.

[163] M. Denninger, **M. Sundermeyer**, D. Winkelbauer, D. Olefir, T. Hodan, Y. Zidan, M. Elbadrawy, M. Knauer, H. Katam, and A. Lodhi. "Blenderproc: Reducing the reality gap with photorealistic rendering". In: *International Conference on Robotics Sciene and Systems (RSS) Workshops*. 2020.

[168] T. Hodaň, **M. Sundermeyer**, B. Drost, Y. Labbé, E. Brachmann, F. Michel, C. Rother, and J. Matas. "BOP challenge 2020 on 6D object localization". In: *European Conference on Computer Vision Workshops (ECCVW)*. 2020.

[180] **M. Sundermeyer**, M. Durner, E. Y. Puang, Z.-C. Marton, N. Vaskevicius, K. O. Arras, and R. Triebel. "Multi-path Learning for Object Pose Estimation across Domains". In: *Conference on Computer Vision and Pattern Recognition (CVPR)*. 2020.

[181] **M. Sundermeyer**, Z.-C. Marton, M. Durner, and R. Triebel. "Augmented Autoencoders: Implicit 3D Orientation Learning for 6D Object Detection". In: *International Journal of Computer Vision (IJCV)*. 2020.

[198] **M. Sundermeyer**, A. Mousavian, R. Triebel, and D. Fox. "Contact-Graspnet: Efficient 6-DoF Grasp Generation in Cluttered Scenes". In: *International Conference on Robotics and Automation (ICRA)*. 2021.

[214] M. Stoiber, **M. Sundermeyer**, and R. Triebel. "Iterative Corresponding Geometry: Fusing Region and Depth for Highly Efficient 3D Tracking of Textureless Objects". In: *Conference on Computer Vision and Pattern Recognition (CVPR)*. 2022.

[222] M. Stoiber, **M. Sundermeyer**, W. Boerdijk, and R. Triebel. *"A Multi-body Tracking Framework–From Rigid Objects to Kinematic Structures"*. Submitted to *Transactions on Pattern Analysis and Machine Intelligence* (TPAMI). 2023.

[223] **M. Sundermeyer**, T. Hodaň, Y. Labbe, G. Wang, E. Brachmann, B. Drost, C. Rother, and J. Matas. "BOP Challenge 2022 on Detection, Segmentation and Pose Estimation of Specific Rigid Objects". In: *Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*. 2023.

1.3.1 Other publications not explicitly discussed in this thesis:

[110] M. Brucker, M. Durner, Z. Marton, F. Bálint-Benczédi, **M. Sundermeyer**, and R. Triebel. "6DoF Pose Estimation for Industrial Manipulation based on Synthetic Data". In: *International Symposium on Experimental Robotic (ISER)*. 2018.

[131] W. Boerdijk, **M. Sundermeyer**, M. Durner, and R. Triebel. "Self-Supervised Object-in-Gripper Segmentation from Robotic Motions". In: *Conference on Robot Learning (CoRL)*. 2019.

[188] W. Boerdijk, **M. Sundermeyer**, M. Durner, and R. Triebel. "What's This?– Learning to Segment Unknown Objects from Manipulation Sequences". In: *International Conference on Robotics and Automation (ICRA)*. 2021.

[193] M. Durner, W. Boerdijk, **M. Sundermeyer**, W. Friedl, Z.-C. Marton, and R. Triebel. "Unknown Object Segmentation from Stereo Images". In: *International Conference on Intelligent Robots and Systems (IROS)*. 2021.

[202] W. Boerdijk, M. Durner, **M. Sundermeyer**, and R. Triebel. "Towards Robust Perception of Unknown Objects in the Wild". In: *ICRA Workshop: Robotic Perception and Mapping: Emerging Techniques*. 2022.

[224] M. Ulmer, M. Durner, **M. Sundermeyer**, M. Stoiber, and R. Triebel. "6D Object Pose Estimation from Approximate 3D Models for Orbital Robotics". In: *International Conference on Intelligent Robots and Systems (IROS)*. 2023.

2 Fundamentals and Problem Definitions

2.1 Pinhole Camera Model

The pinhole camera model describes how a point in 3D camera coordinates $P_c = (X, Y, Z)^T$ projects to a 2D point $p = (x, y)^T$ on the image plane of an ideal pinhole camera, as shown in Fig. 2.1. The aperture of an ideal pinhole camera is assumed to be a point with no lenses to focus light. Under these assumptions, we can simply use the focal length f, defined as the distance between image plane and pinhole, to compute the 2D image coordinates as $p = (x, y)^T = (f\frac{X}{Z}, f\frac{Y}{Z})^T$.

In contrast, real perspective cameras have lenses and non-zero aperture which violates the ideal pinhole assumptions and produces images with lens distortion. However, for most perspective cameras these effects can be corrected by using e.g. the Brown–Conrady distortion model [3] to produce mostly undistorted images. These undistorted images can then again be approximated well by the ideal pinhole camera model.

Since all points on an infinite 3D projection ray map to the same point in 2D, we usually describe the pinhole relation using homogeneous coordinates $P = (kX, kY, kZ, k)$ with $k > 0$ defining the point on the ray which intersects the image plane at $k = 1$. This also simplifies many downstream derivations and lets us formulate the perspective projection as a linear system

$$p = \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = \begin{bmatrix} f\frac{X}{Z} \\ f\frac{Y}{Z} \\ 1 \end{bmatrix} \sim \begin{bmatrix} fX \\ fY \\ Z \end{bmatrix} = \underbrace{\begin{bmatrix} f & 0 & 0 & 0 \\ 0 & f & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix}}_{C_{pin}} \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix}_c \tag{2.1}$$

with C_{pin} being the camera matrix of the ideal pinhole model. For real perspec-

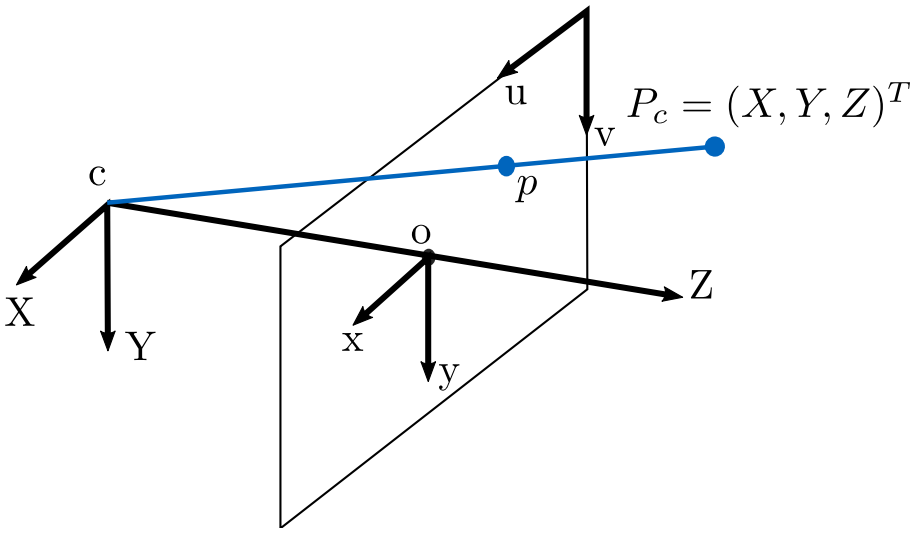


Figure 2.1: Pinhole Camera Model using the OpenCV coordinate convention.

tive projections the camera intrinsics matrix K is commonly used to map to a coordinate system defined at the top-left corner of the image. It also accounts for lateral offsets of the image sensor which shifts the optical center to $o = (c_x, c_y)^T$ in pixels:

$$p = \begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \begin{bmatrix} f_x \frac{X}{Z} + c_x \\ f_y \frac{Y}{Z} + c_y \\ 1 \end{bmatrix} \sim \begin{bmatrix} f_x X + c_x Z \\ f_y Y + c_y Z \\ Z \end{bmatrix} = \underbrace{\begin{bmatrix} f_x & \gamma & c_x & 0 \\ 0 & f_y & c_y & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix}}_{K} \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix}_c = KP_c \quad (2.2)$$

where γ represents the skew coefficient which is ≈ 0 for modern cameras. The focal length f is divided into f_x and f_y in pixels accounting for slightly different projections along coordinate directions.

2.1.1 Extrinsics

So far we assumed that the point P_c is given in camera coordinates. To project a point P_w defined in world coordinates, i.e. any other Cartesian coordinate system, we need to first transform it back to camera coordinates. The full projection based

on the camera matrix C is therefore defined as

$$p = \begin{bmatrix} u \\ v \\ 1 \end{bmatrix} \sim \underbrace{\begin{bmatrix} f_x & \gamma & c_x & 0 \\ 0 & f_y & c_y & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix}}_{K} \underbrace{\begin{bmatrix} r_{11} & r_{12} & r_{13} & t_1 \\ r_{21} & r_{22} & r_{23} & t_2 \\ r_{31} & r_{32} & r_{33} & t_3 \\ 0 & 0 & 0 & 1 \end{bmatrix}}_{H_w^c = [R|t]} \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix}_w = \underbrace{K[R|t]}_{C} P_w \tag{2.3}$$

where $H_w^c = [R|t]$ is the homogeneous transformation from world to camera coordinates formed by the rotation matrix R and the translation vector t.

2.2 6-DoF Object Pose Estimation

H_w^c can also be interpreted as the pose of the world frame in camera coordinates. Now, if we define the world frame to be the coordinate frame of an object, H_w^c corresponds to the 6-DoF pose of the object in camera coordinates H_{obj}^c. Consequently, a point P_{obj} in 3D object coordinates projects to a 2D point p in the image plane as

$$p = \begin{bmatrix} u \\ v \\ 1 \end{bmatrix} \sim K H_{obj}^c P_{obj} \tag{2.4}$$

The 6-DoF object pose H_{obj}^c is the interest of many applications, so the questions arises on how it can be estimated?

2.2.1 Perspective-n-Point

Given the camera intrinsics K, one way to solve projection 2.4 for the object pose H_{obj}^c is by finding at least three 2D-3D correspondences between points in the object coordinate system $P_{obj,i}$ and their 2D projections p_i in the image plane. This is known as the Perspective-3-Point (P3P) problem which was first solved by the mathematician Grunert [1] in 1841. However, with just three point correspondences, we obtain up to four geometrically feasible solutions. A fourth non-colinear correspondence can be used to remove this ambiguity. In practice, more 2D-3D correspondences are usually estimated to solve the general Perspective-n-Point (PnP) problem and achieve robustness against noise and

outliers, often in conjunction with a voting scheme like RANSAC [6]. We search for R and t that minimize the reprojection error

$$\text{PnP}(p, P, K) = \underset{R,t}{\text{argmin}} \frac{1}{n} \sum_{1}^{n} ||p_i - K[R|t]P_i||_2^2 \tag{2.5}$$

A popular variant is the Efficient PnP (EPnP) [23] by Lepetit, et al. where the $n \geq 4$ points are represented by a weighted linear combination of four virtual control points resulting in $\mathcal{O}(n)$ time complexity.

For 6-DoF object pose estimation a 3D object model is often used to define 3D reference points such as the eight corners of an object aligned 3D bounding box [102] or normalized object coordinates (NOCS) [155]. The major challenge in the PnP framework is to robustly predict these correspondences in monocular images and several approaches will be discussed in Chapter 3.

2.2.2 Kabsch Algorithm

If an RGB-D image is available, we can unproject pixels into 3D camera coordinates P_c and the relation 2.4 simplifies to

$$P_c = H_{obj}^c P_{obj} \tag{2.6}$$

We can find a unique solution for the object pose H_{obj}^c from at least three 3D-3D point correspondences using the Kabsch algorithm [5]. After centering the two 3D point sets, we compute their covariance matrix $H = P^T Q \mid P, Q \in \mathbb{R}^{nx3}$ from which the optimal rotation to align the points can be determined by $R = (H^T H)^{\frac{1}{2}} H^{-1}$. However, since H is not guaranteed to have an inverse we commonly perform a singular value decomposition (SVD) $H = U\Sigma V^T$. Then the optimal rotation matrix R follows as

$$R = V \begin{Bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & d \end{Bmatrix} U^T \qquad \text{where} \quad d = sign(\det(VU^T)) \tag{2.7}$$

Outliers can again be removed using e.g. RANSAC [6].

2.2.3 Explicit 6-DoF Pose Representations

Estimating correspondence points is just one way to retrieve a 6-DoF object pose H^c_{obj}. We will discuss other methods such as template matching [29, 25, 32], point-pair features (PPFs) [128, 26], pose regression [160, 58] and classification [95] in the related work, Chapter 3.

An explicit 6-DoF pose can be represented in various forms and its choice is especially crucial for pose regression methods [160]. In equation 2.6 and 2.4 we used homogeneous coordinates to depict a 6-DoF transformation which contain twelve parameters: nine for the rotation matrix R and three for the 3D translation vector t. However, for direct regression methods it can be beneficial to re-parameterize this representation. For example, instead of directly regressing the 3D translation vector, we can decouple the translation into the projection of the object centroid to the image plane and its distance in Z direction, thus simplifying the regression task [95, 107].

There are a variety of representations to depict 3D rotations such as quaternions, axis-angles, Euler angles and rotation matrices. For example, three Euler angles α, β, γ can describe any valid 3D rotation. They are applied subsequently in a predefined order since they are non-commutative. A common convention is to start with a rotation of γ around the original x-axis, followed by a rotation of β around the *resulting* y-axis and finally a rotation α around the *resulting* z axis. Each Euler angle can be converted to a rotation matrix that is left multiplied to the target vector as

$$
\begin{aligned}
R &= R_z(\alpha)\, R_y(\beta)\, R_x(\gamma) \\
&= \begin{bmatrix} \cos\alpha & -\sin\alpha & 0 \\ \sin\alpha & \cos\alpha & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \cos\beta & 0 & \sin\beta \\ 0 & 1 & 0 \\ -\sin\beta & 0 & \cos\beta \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\gamma & -\sin\gamma \\ 0 & \sin\gamma & \cos\gamma \end{bmatrix} \\
&= \begin{bmatrix} \cos\alpha\cos\beta & \cos\alpha\sin\beta\sin\gamma - \sin\alpha\cos\gamma & \cos\alpha\sin\beta\cos\gamma + \sin\alpha\sin\gamma \\ \sin\alpha\cos\beta & \sin\alpha\sin\beta\sin\gamma + \cos\alpha\cos\gamma & \sin\alpha\sin\beta\cos\gamma - \cos\alpha\sin\gamma \\ -\sin\beta & \cos\beta\sin\gamma & \cos\beta\cos\gamma \end{bmatrix}
\end{aligned}
\tag{2.8}
$$

The individual and combined rotation matrices are orthogonal with a determinant of 1 which preserves the right-handed orientation. All feasible rotations in $\mathbb{R}^3$ form the special orthogonal group SO(3).

Euler angles are rather intuitive and only require 3 dimensions, but they also have several drawbacks. They are always discontinuous at certain points and need to be constrained to certain ranges for uniqueness, e.g. $\alpha, \gamma \in [-\pi, \pi]$ and $\beta \in [-\frac{\pi}{2}, \frac{\pi}{2}]$. Even then we can find multiple Euler angles describing the same 3D rotation at so-called Gimbal locks, where we lose a degree of freedom and can describe the same 3D rotation using arbitrary combinations of the remaining two angles. The ambiguities and discontinuities make Euler angles unsuitable for regression tasks.

Axis-angle representations are also 3-dimensional combining a unit vector e indicating the rotation axis with a rotation angle θ encoded as vector $r = \theta e$. They do not suffer from Gimbal locks but need to be transformed to other representations for composition, are non-unique $(-\theta)(-e) = \theta e$ and the angle also has a discontinuity.

Unit Quaternions have four dimensions $(q_1, q_2, q_3, q_4) \in \mathbb{R}^4$ and are given by $q = q_1 + q_2 i + q_3 j + q_4 k$ where $i^2 = j^2 = k^2 = ijk = -1$. They behave similar to the axis-angle representation, i.e. cause no Gimbal locks, but in contrast can be composed by simple vector multiplications. A 3D vector $v \in \mathbb{R}^3$ can be rotated by $v' = qvq^{-1}$. Still they are non-unique, $q = -q$ represent the same 3D rotation, and have a discontinuity. In fact, it has been shown [160] that any $\leq 4D$ representation has discontinuities which impedes their regression.

Rotation matrices are continuous representations of 3D rotations containing nine entries consisting of three basis vectors $r_j \in \mathbb{R}^3 \mid j \in \mathbb{Z} : j \in [1, 3]$. The basis vectors r_j are orthonormal to each other so that the third basis vector can be derived from the cross product of the first two basis vectors $r_3 = r_1 \times r_2$ resulting in a 6D representation. The dimension can be even further reduced to 5D by stereographic projection while still being continuous since basis vectors are constrained to unit length $||r_j|| = 1$. Interestingly, empirical results [160] suggest that the higher dimensional 6D representation is a favorable regression target for neural network trainings. The authors hypothesize that the stereographic projection to 5D causes distortions in the gradients. However, increasing the dimensions of regression targets has been demonstrated to be beneficial for other, even larger representations such as the full 9D rotation matrix projected onto SO(3) using SVD/Procrustes orthonormalization [174, 189]. In [189] it is

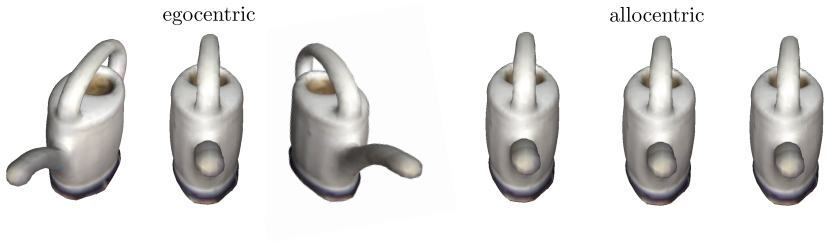


Figure 2.2: Egocentric vs. allocentric viewpoints. Shown are cropped views of an object translated along the camera X axis at constant 3D rotation.

experimentally shown that losses on higher dimensional regression targets behave more linearly wrt. rotation erros which could explain the better performance using gradient based methods. In our work on 6-DoF grasping [198] we build upon the 6D representation combined with a Gram Schmidt orthogonalization.

Egocentric vs. Allocentric: To reduce the combinatorial 6-DoF search space it is favorable to estimate 3D rotation and 3D translation independently. However, from the egocentric camera viewpoint, an object translation parallel to the image plane also results in a change of appearance, as shown in Fig. 2.2 on the left. Therefore, the egocentric representation prevents estimating the 3D rotation from an image crop alone because the appearance depends on the translation as well. To alleviate this issue, one can estimate object rotation in the allocentric parameterization [117] which is agnostic to 3D translations, as shown in Fig. 2.2 on the right, and convert it back to egocentric camera coordinates subsequently.

2.3 6-DoF Robotic Grasp Estimation

In 6-DoF object pose estimation, we determine the current pose of a known, rigid object that is observed in a sensor image. In contrast, 6-DoF robotic grasp estimation seeks a virtual 6-DoF gripper pose from which a robotic grasp of a potentially unknown target object is stable. Commonly, not only a single grasp is estimated but a full distribution of stable 6-DoF grasps so that the subsequent planner can pick a non-colliding, reachable and efficient grasp for a specified target object.

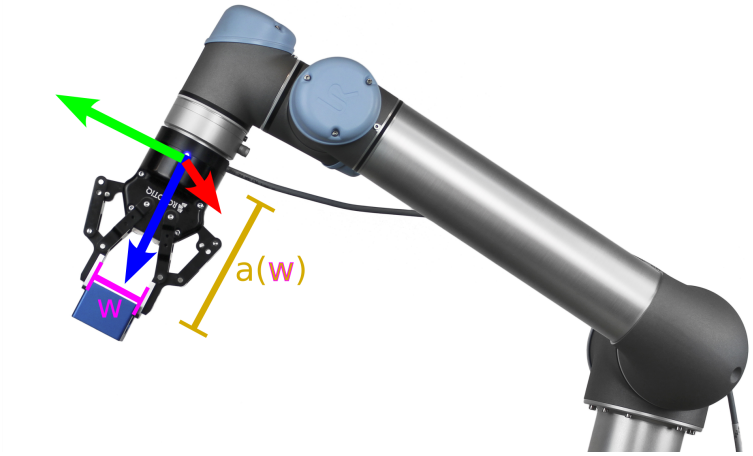


Figure 2.3: A parallel-jaw robotic grasp can be parameterized by a 6-DoF end-effector pose, the gripper width w and the gripper depth $a(w)$. Depending on the gripper model there is an analytic dependence of the gripper depth $a(w)$ on the gripper width w.

2.3.1 6-DoF Parallel Jaw Grasp Representations

A 6-DoF robotic grasp pose of parallel jaw grippers can be described similarly to a 6-DoF object pose, i.e. the representations introduced in Section 2.2.3 still apply. In addition to the 6-DoF object pose of the gripper origin, a grasp is characterized by its grasp width w as shown in Fig. 2.3. For certain gripper models such as the RobotiQ grippers[1], the grasp depth a is not constant but depends on the grasp width w.

2.3.2 Robotic Grasp Execution

To execute an estimated 6-DoF grasp, we first need to perform a hand–eye calibration between an in-hand or external camera and the robot base. Usually, we can retrieve the camera in robot base coordinates H_c^{base} using an extrinsic calibration method, e.g. by Tsai et al. [9]. Estimated grasp poses in camera coordinates H_g^c are transformed to grasp poses in robot base coordinates H_g^{base} by

$$H_g^{base} = H_c^{base} H_g^c \tag{2.9}$$

A robotic motion planner [48] then plans a non-colliding end-effector trajectory to the grasp pose H_g^{base} where the gripper is closed. The grasp approach and retreat directions are along the z-axis of the gripper.

[1] https://robotiq.com/products/2f85-140-adaptive-robot-gripper

3 Related Work

This chapter introduces and categorizes existing research in 6-DoF object and grasp pose estimation with a focus on learning-based techniques that are most relevant to this dissertation. We do not claim to list every published method in this increasingly large field but instead aim to give a comprehensible overview of the main foundational directions.

3.1 6-DoF Object Pose Estimation

6-DoF object pose estimation has drawn attention from the early days in Computer Vision and is still a very active and growing area of research. We discuss the lines of work that evolved with different sensory, algorithmic and computational advances. While we distinguish between classical and learning-based methods, it is notable that many learning-based methods are based on or inspired by classical ideas which are briefly introduced in the following.

3.1.1 Classical Low-level Feature Matching

For many years, 6-DoF object pose estimation was primarily performed based on matching low-level features such as corners, edges, templates, feature points or pairs. These low-level features were extracted from grayscale, color and later also depth images as well as 3D object models using classical Computer Vision techniques [15, 36].

3.1.1.1 Edge-based Matching

The earliest works in 6-DoF object pose estimation go back to the 1960s where Roberts [2] fitted 3D wire-frame models to edges in grayscale images. At the time such methods were limited by the amount of available computation. Lowe [8]

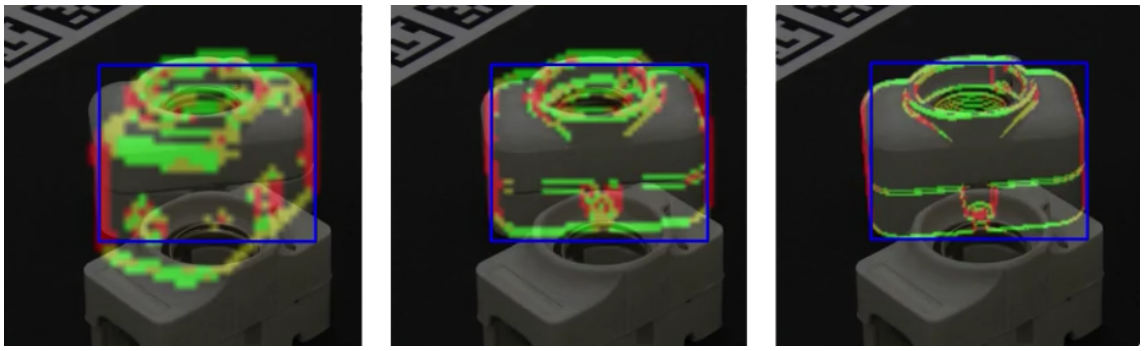


Figure 3.1: Our GPU implementation of edge-based pose estimation based on Ulrich et al. [25] combined with CNN-based 2D object detection [77]. From left to right 3D model edges are matched with 2D image edges at hierarchically finer resolutions while reducing the large 6-DoF search space.

extended edge-based recognition and pose estimmation to 3D object models and filtered corresponding edges for viewpoint invariance. In 2009, Ulrich et al. [25] proposed an edge-based pose estimation method that matches 3D edges extracted from object models with 2D edges extracted at multiple levels of an image pyramid. Based on these ideas we implemented a GPU accelerated version of the same approach combined with learning based object detection [77] as shown in Fig. 3.1. Especially for texture-less, industrial objects their approach demonstrates better accuracy and robustness over feature point methods described in the next Sec. 3.1.1.2. However, depending on the object, edges are not always present or observable from any viewpoint, and it is typically difficult to distinguish whether edges in 2D images originate from geometry or texture. Furthermore, while GPU acceleration, hierarchical resolutions and 2D object detection reduce the runtime, online rendering still bottlenecks the efficiency of such approaches, particularly if the full 6-DoF pose space is searched.

3.1.1.2 Local Feature Point Matching

In the 2000s, keypoint detection and description techniques from color or grayscale images such as SIFT [13] or SURF [19] revolutionized the image matching field. These local features points are typically invariant to changes in illumination and affine image transformations. Consequently, feature point matching also attracted interest in the object pose estimation domain where 2D-3D correspondences to a reference 3D model are sought. Lepetit [17] estimated box and face poses based on feature points using an iterative POSIT [12] + RANSAC [6] solver. Today,

2D-3D correspondence problems are often solved with Perspective-n-Point (PnP) + RANSAC algorithms [23] that do not require initial pose estimates, see Sec. 2.2.1 for an introduction. Classical keypoint matching approaches excel on objects with non-repetitive texture where the local regions can be repeatedly detected and distinctively described despite viewpoint changes. However, for texture-less and symmetric objects [93] their robustness is often compromised.

3.1.1.3 Template-based Matching

Template-based methods [29, 38] were introduced to tackle objects that do not necessarily contain clear edges or rich texture. In an offline stage, template views are recorded or rendered from a 3D model and low-level features such as color histograms or gradients are extracted. At test time, the templates with corresponding object poses are compared against the image at different image positions and scales to detect an object and determine its pose. Multi-modal template matching [32, 38] from RGB-D data was introduced to improve robustness and accuracy. Template-based approaches excel on texture-less objects and can be implemented efficiently on parallel hardware [38]. They can have difficulties with strong occlusions, high-frequency textures and distinguishing similar instances.

3.1.1.4 Depth-based Pose Estimation and Refinement

With the rise of consumer depth sensors [34], pure depth-based methods for object pose estimation and refinement were introduced. Point cloud registration techniques such as Iterative closest point (ICP) [11] refine object poses given a close initial estimate. In contrast to the Kabsch algorithm introduced in Section 2.2.2 which assumes correspondences between point sets to be given, ICP iteratively estimates the point correspondences, e.g. by nearest neighbor in the simplest form. Different ICP variants [43] such as point-to-plane ICP exhibit increased robustness against noise and occlusions and therefore still enjoy popularity in modern computer vision pipelines. However, since ICP performs local pose optimizations, it relies on accurate initialization without which the optimization can get stuck in local minima.

For absolute depth-based pose estimation, 3D features such as e.g. SHOT [54] can be extracted from the observed point cloud to establish 3D-3D correspondences

with the object model. Kabsch [5] + RANSAC solvers [35] subsequently yield the 6-DoF object pose as formulated in Section 2.2.2.

Point pair features (PPFs) Alternatively, Point pair features (PPFs) [26] proposed in 2010 by Drost et al. are still widely used in computer vision applications [110, 223] as of 2023. A point pair feature depicts the relation of two 3D points based on their distance and based on the orientation of their surface normals. Crucially, these distinctive relations are invariant under rigid body transformations which makes them an excellent candidate for 6-DoF object pose estimation. At training time, PPFs are sampled from the 3D object model, then binned and stored in a hash-table. At test time, PPFs are sampled from the observed point cloud and similar point pairs are retrieved from the hash table to vote for an object pose. Several works improve upon the original PPFs, e.g. by novel voting and sampling schemes [73] or view-dependent verifications [128]. According to experiments in [116] point pair features mostly outperform SHOT [54] features for depth-based pose estimation except when point clouds are low-resolution and noisy. In the Benchmark for 6D Object Pose Estimation (BOP challenge) [137] that captures the state-of-the-art in the field, PPF-based methods combined with ICP had the most accurate results in 2019 [128] and still competitive results in 2020 [172].

However, several limitations of purely depth-based approaches were also exposed. First of all, existing time-of-flight (ToF) or structured light depth sensors are often sensitive to sunlight as well as specular or light absorbing object materials that produce missing or wrong measurements which consequentially leads to wrong pose estimates. Furthermore, purely geometric approaches usually rely on the computationally expensive evaluation of many pose hypotheses and do not take into account high level features. Thus, they have troubles detecting and distinguishing different objects in cluttered environments. An effective way to reduce the search space and improve the robustness against clutter is to combine them with 2D segmentation networks as shown by Koenig et al. [172].

3.1.2 Learning-based Approaches

Learning-based methods have been proposed to tackle the previously described limitations of classical pose estimation and to increase the robustness and efficiency under challenging test conditions. In an early attempt, Jurie et al. [16] improved template tracking by replacing pre-computed Jacobians with a matrix learned by linear regression on data pairs. Later, random forest classifiers were deployed by Tejani et al. to perform patch-based voting for 6-DoF poses [55]. Brachmann et al. [45] also used random forests to predict 3D object coordinates for 6-DoF pose estimation.

More recently, Convolutional Neural Networks (CNNs) are predominantly trained to perform 6-DoF object pose estimation. When trained on sufficient, representative data, CNNs can extract high-level features from 2D images which allows detecting a wide variety of objects in cluttered environments [77, 120]. While the training of these networks can be resource and time intensive, the inference time is usually on par or lower than classical methods and scales better with the number of target objects [94]. To further estimate the full 6-DoF pose, different strategies have been developed which we will categorize in the following.

3.1.2.1 Direct 6D Pose Regression or Classification

The most straightforward approach is to let a neural network directly regress or classify the 6D pose from a 2D input image. PoseNet [58] performs such a direct 6-DoF camera pose regression on images but the reported results are subpar to classical approaches. Their training and test data come from a similar, limited subset of SE(3) and they tackle camera pose estimation which is usually easier than object pose estimation since features in the whole image can be utilized. When training object pose estimation on the full, non-euclidean 6-DoF target space in cluttered scenes, the direct regression approach hardly converges on the training set and does not generalize well to unseen regions of SE(3) as shown by Zhao et al. [184].

Expanding 2D Object Detection and Segmentation Networks To overcome the curse of dimensionality, existing 2D object detectors were extended with additional branches that directly regress or classify an independent representation of 3D rotation. Kehl et al. [95] adapted the 2D Single Shot Multibox Detector

(SSD) [77] by adding a branch that classifies discretized 3D rotations. The 3D translation is then computed based on the size of the 2D bounding box, the 3D model size and camera intrinsics. Xiang et al. [107] proposed PoseCNN that instead predicts object masks and regresses pixel-wise 2D unit vectors pointing towards the projected object centers as well as their distance to the camera. The 3D translation is determined by Hough voting for the 2D object centers from the predicted unit vectors combined with an average over the predicted pixel-wise distances from the camera. The 3D orientation is derived by cropping a region around the instance mask and regressing a quaternion representation from it.

Regressing Representations of 3D Rotations Do et al. [134] instead predict 3D orientations parameterized by a Lie algebra representation. Zhou et al. [160] further compared multiple different output representations of 3D rotations, such as quaternions, rotation matrices and Euler angles, and showed that they have a major influence on the ability of neural networks to regress them. They find that the discontinuities that are present in all four or less dimensional representations impede neural network training. Moreover, they find that higher dimensional targets such as the 6D representation, consisting of two of the three unit vectors of a rotation matrix, perform best. Levinson et al. [174] confirm this relationship and find that regressing the full 9D rotation matrix followed by an SVD orthogonalization does improve results further. An introduction to the different representations and their impact on performance can be found in Section 2.2.3.

Classification While regressing rotations seems like a natural choice, Li et al. [118] and also Mahendran et al. [96] have found that a mix of 3D rotation classification and subsequent regression of the remaining pose delta can perform better. The motivation is that direct regression in large output spaces such as SO(3) cannot easily deal with multi-modal targets originating from symmetries or imbalanced rotation distributions in the training set and consequently often result in predictions that are biased towards the mean. In contrast, classification allows reweighing or resampling the bins depending on the data distribution and losses such as binary cross-entropy can optimize for multiple possible rotations jointly. On the other hand, classification of 3D object orientations requires a discretization of SO(3). Even rather coarse intervals of $\sim 5^o$ lead to over 50.000 possible classes. Since each class appears only sparsely in the training data, this hinders

convergence. In SSD6D [95] the 3D orientation is learned by separately classifying a discretized viewpoint and in-plane rotation, thus reducing the complexity to $\mathcal{O}(n^2)$. However, for non-canonical views, e.g. if an object is seen from above, a change of viewpoint can be nearly equivalent to a change of in-plane rotation which yields ambiguous class combinations. Another general disadvantage of one-hot classification is that the relation between similar orientations is ignored.

Single Stage vs. Multi Stage Most of the previously described methods are single stage, i.e. they process the whole image or point cloud in a single forward pass to predict the 3D translation and 3D rotation of an object wrt. the camera. While single stage methods [126, 127] can be fast, it has been shown for example in the BOP challenges [223, 168] that two stage pipelines which perform 2D object detection or segmentation followed by 6D pose estimation on the resulting crops reach superior performance. Intuitively, detection benefits from viewpoint invariant features whereas pose estimation requires viewpoint variant features which might favor the separation of feature extractors. Furthermore, the pose network can focus computations on the relevant parts of the scene at higher local resolutions and the sequential estimation reduces the combinatorial search space. As explained in Section 2.2.3, this separation of translation and rotation prediction is not possible if poses are given from an egocentric viewpoint since the appearance of objects inside crops depends on both object translation and rotation. Therefore, Kundu et al. [117] proposed to instead regress allocentric poses which resolves the dependency between 3D rotation and 3D translation. Alternatively, we [124] instead train wrt. canonical viewpoints at the optical center and correct 3D rotation estimates depending on the independently estimated translation, as described in Section 4.1.3.6.

3.1.2.2 Correspondence-based 6D Pose Estimation

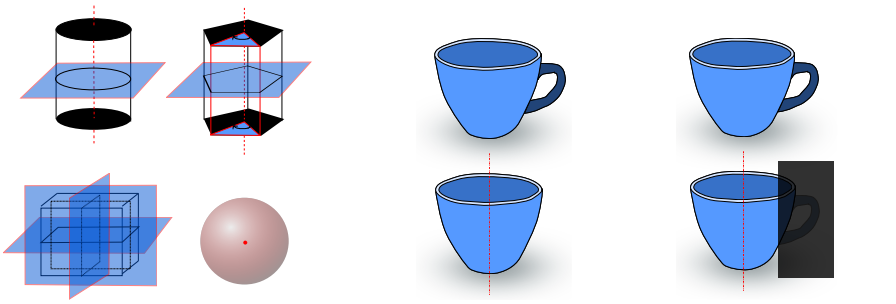
Another line of work estimates sparse or dense 2D-3D correspondences between object views in 2D images and the corresponding 3D model. Variants of Perspective-N-Point (PnP) together with RANSAC are applied to estimate the 6-DoF pose [45, 102, 166, 159].

Sparse Correspondences Several works such as BB8 from Rad and Lepetit [102] predict the locations of the eight 3D object bounding box corners in the image

plane. To enhance the stability of the sparse regression, BB8 [102] predicts the probability of each pixel within a 2D object crop to correspond to a specific 3D keypoint. However, the 3D bounding box corners can be far away from the object which can complicate the regression for neural networks. Therefore, DOPE [127] and PVNet [150] regress 2D vector fields pointing towards the corner keypoint locations which are then determined via voting. They show that this improves the prediction of occluded and further keypoints. Tremblay et al. [127] and Tekin et al. [126] propose single-stage networks for 3D bounding box regression that additionally predict a 3D centroid keypoint. Instead of 3D bounding box regression other approaches attempt to define sparse 3D keypoints on the surface of the 3D model using heuristics such as farthest point algorithms [150] or learn the most visually consistent 3D keypoint positions [185]. However, due to the low number of 2D-3D matches, processing sparse correspondences with PnP / RANSAC can be prone to outliers. The sparse keypoint predictions can also suffer from occlusions and pose ambiguities.

Dense Correspondences Dense correspondences aim to predict the (normalized) 3D object coordinate location for each pixel in the image plane that belongs to the target object. This results in a variable and usually higher number of 2D-3D matches than sparse methods that can be used to optimize a 6-DoF object pose with more effective outlier filtering. As only the visible part and not the occluded part of the object is matched, the correspondences are often more stable. However, especially under occlusions, dense correspondences are also closer together than e.g. sparse bounding box corners and therefore require more precise matches to achieve the same 6-DoF pose accuracy.

An early work from Brachmann et al. [45] uses random forests for dense 3D object coordinate prediction. For each RGB-D input pixel the 3D location of the corresponding 3D model point in object coordinates is predicted. The trees simply split according to differences in RGB and depth. Later, encoder-decoder CNNs [155, 148, 141] were deployed to predict normalized 3D object coordinates (NOCS) [155] from RGB input crops only. Formulating the NOCS regression as a discrete classification problem was found to lead to faster convergence and superior quality of correspondences. DPOD [159] and DPODv2 [197] discretize the continuous normalized object coordinate space $[0, 1]^3$ into 256 classes. Hodan et al. [166] propose to discretize the 3D model surface into 64 fragments using farthest point sampling. To obtain precise 3D locations, EPOS [166] additionally



(a) Object symmetries **(b)** Self-occlusion induced symmetries **(c)** Occlusion induced symmetries

Figure 3.2: Causes of pose ambiguities

predicts local 3D fragment coordinates for each of the fragments. Another major advantage of classification is that multiple possible fragments can be predicted for each pixel which improves performance for symmetric objects.

Recent approaches such as GDRNet [200, 170] predict dense correspondences as an intermediate network representation from which a light-weight network head directly regresses a representation of 3D object rotation and translation. This end-to-end procedure is shown to perform on par or better than PnP / RANSAC methods while being significantly faster. In the BOP challenge 2022 [223] a method based on GDRNet called GDRNPP [209] displayed excellent performance while having an inference time of around 0.2 seconds per scene. We analyze the BOP challenge 2022 results in detail in Section 4.6.2.

3.1.2.3 Handling Object Pose Ambiguities

The mapping from an object view to an object pose can be ambiguous, i.e. the same object view might explain multiple different object poses. This is an issue for any unimodal learning-based algorithm that discriminates object views based on fixed SO(3) representations.

Symmetries, as shown in Figure 3.2, are a major cause of pose ambiguities. If not addressed, identical training images can have different orientation labels assigned which can significantly disturb the learning process itself. For example, when regressing the orientation of a line-symmetric object, the training often leads to average predictions which fail to explain either of the two potential ground truth poses corresponding to an object view. Likewise, the PnP / RANSAC pose optimization expects one-to-one 2D-3D correspondences and can fail if correspondences between the object view and the 3D model are ambiguous.

In order to cope with ambiguous object poses, most approaches in literature are manually adapted [67, 39, 95, 102]. The strategies reach from ignoring poses around a specified symmetry axis [67, 39] over adapting the training viewpoints according to the object symmetries [95] to the training of extra CNNs to predict specific symmetries [102]. These depict tedious, manual ways to filter out global object symmetries (Fig. 3.2a) in advance. But even the definition of object symmetries is not always straightforward in practice where small differences in geometry and texture can break them. 3D reconstructed models usually have imperfections which makes the automatic detection of symmetries from 3D models difficult. Xiang et al. [107] proposes a loss that is invariant to global object symmetries by computing 3D distances between the closest surface points of the 3D model in the predicted and ground truth pose. However, this approach ignores texture and is insensitive to small symmetry breaking geometrical features.

Ambiguities due to self-occlusions (Fig. 3.2b) and occlusions (Fig. 3.2c) are viewpoint dependent and cannot be covered by simply computing global object symmetries. For instance, many objects in the T-LESS dataset [93] contain small symmetry breaking details that are not visible from all viewpoints. To tackle this issue, Manhardt et al. [145] regress multiple quaternions for each 2D object detection but compute the loss only against the closest ground truth quaternion to account for potential symmetries. Hodan et al. [166] train many-to-many 2D-3D correspondences between object pixels and surface fragments. To find a set of consistent one-to-one 2D-3D correspondences among them they deploy EPnP [23] with GC-RANSAC [108] that takes into account spatial coherence, i.e. it is assumed that nearby (in 2D and 3D) correspondences belong to the same pose hypothesis. It is shown that this improves the robustness under ambiguous correspondences.

3.1.2.4 Latent Embeddings for 6D Pose Estimation

The pose ambiguities described in the last Section 3.1.2.3 motivate a third perspective on 6D pose estimation which we will focus on in this thesis. The goal is to learn latent embedding spaces that do not necessarily depend on potentially ambiguous explicit poses but encode object appearance from different viewpoints of the target object.

Wohlhart et al. [67] introduced a CNN-based descriptor learning approach using a triplet loss that minimizes/maximizes the Euclidean distance between

similar/dissimilar object orientations. In addition, the distance between different objects is maximized. Although mixing in synthetic data, the training also relies on pose-annotated sensor data. The approach is not immune against symmetries since the descriptors are supervised using explicit 3D orientations. Thus, the loss can be dominated by symmetric object views that appear the same but have opposite orientations which can still produce incorrect average pose predictions.

Balntas et al. [85] extended this work by enforcing proportionality between descriptor and pose distances. They acknowledge the problem of object symmetries by weighting the pose distance loss with the depth difference of the object at the considered poses. This heuristic increases the accuracy on symmetric objects with respect to [67].

Our work is also based on learning descriptors, but in contrast we train our Augmented Autoencoders (AAEs) such that the learning process itself is independent of any fixed SO(3) representation. The loss is solely based on the appearance of the reconstructed object views and thus symmetrical ambiguities are inherently regarded. Thus, unlike [85, 67] we abstain from the use of real pose-annotated data during training and instead train completely self-supervised. The assignment of explicit 3D orientations to the descriptors only happens after the training. Our Multi-Path Encoders [180] extend this approach to encode multiple objects at the same time in the same latent space using one general encoder and multiple object-specific decoders. These view-centric representations were also shown to work well for 6D object tracking [132].

Kehl et al. [75] train an Autoencoder architecture on random RGB-D scene patches from the LineMOD dataset [32]. At test time, descriptors from scene and object patches are compared to vote for the 6D pose. Since the approach requires the evaluation of many patches, it takes about 670ms per prediction. Furthermore, processing local patches independently means to ignore holistic relations between object features which is crucial if few texture exists. Instead we train on holistic object views and explicitly learn domain invariance.

The advantage of the implicit approach is that symmetric views share the same target representation during trainig. Corresponding explicit poses are automatically grouped and can be retrieved at test time.

3.1.2.5 Simulation to Reality Transfer

CNNs have revolutionized 2D object detection from RGB images [63, 77, 120]. But, in comparison to 2D bounding box annotation, the effort of labeling real images with full 6D object poses is magnitudes higher, requires expert knowledge and a complex setup [93]. Although there are ways to label object poses semi-automatically [122], it remains a cumbersome, error-prone process.

Nevertheless, the majority of learning-based pose estimation methods, namely [126, 67, 69, 102, 107], use real labeled images that you only obtain within pose-annotated datasets.

In consequence, [95, 67, 127, 159] have proposed to train on synthetic images rendered from a 3D model, yielding a great data source with pose labels free of charge. However, naive training on synthetic data does not typically generalize to real test images. Therefore, a main challenge is to bridge the domain gap that separates simulated views from real camera recordings.

There exist three major strategies to generalize from synthetic to real data:

Photo-Realistic Rendering The works of [78, 66, 99, 81] have shown that photo-realistic renderings of object views and backgrounds can in some cases benefit the generalization performance for tasks like object detection and viewpoint estimation. It is especially suitable in simple environments and performs well if jointly trained with a relatively small amount of real annotated images. However, photo-realistic modeling is often imperfect and requires computational effort. Hodan et al. [138] have shown promising results for 2D Object Detection trained on physically-based renderings. Tremblay et al. [127] use mixed training data consisting of photo-realistic rendering and domain randomized samples.

Domain Adaptation Domain Adaptation (DA) [87] refers to leveraging training data from a source domain to a target domain of which a small portion of labeled data (supervised DA) or unlabeled data (unsupervised DA) is available. Generative Adversarial Networks (GANs) have been deployed for unsupervised DA by generating realistic from synthetic images to train classifiers [104], 3D pose estimators [86] and grasping algorithms [109]. While constituting a promising approach, GANs often yield fragile training results. Supervised DA can lower the need for real annotated data, but does not abstain from it.

Domain Randomization Domain Randomization (DR) builds upon the hypothesis that by training a model on rendered views in a variety of semi-realistic settings (augmented with random lighting conditions, backgrounds, saturation, etc.), it will also generalize to real images. Tobin et al. [106] demonstrated the potential of the DR paradigm for 3D shape detection using CNNs. Hinterstoisser et al. [91] showed that by training only the head network of FasterRCNN [63] with randomized synthetic views of a textured 3D model, it also generalizes well to real images. It must be noted, that their rendering is almost photo-realistic as the textured 3D models have very high quality. Kehl et al. [95] pioneered an end-to-end CNN, called 'SSD6D', for 6D object detection that uses a moderate DR strategy to utilize synthetic training data. The authors render views of textured 3D object reconstructions at random poses on top of MS COCO background images [51] while varying brightness and contrast. This lets the network generalize to real images and enables 6D detection at 10Hz. Alghonaim [187] benchmarks the different factors of domain randomization and photorealism on the task of object detection and pose estimation and finds that they are complementary in improving sim2real transfer.

Since photo-realistic renderings are costly, we first follow a less expensive DR strategy [95, 126] as presented in Section 4.1: 3D models are rendered in poses randomly sampled from SO(3) using OpenGL with randomized Phong illumination [4] and superimposed onto real images from data sets such as MS COCO [51] or Pascal VOC [27]. Later in Section 4.5, we show how to scale procedural, photorealistic data generation with our BlenderProc [133, 220] framework and combine it with domain randomization to minimize the sim2real gap.

3.1.2.6 Generalization to Unseen Objects

Most learning-based methods predict the pose of one [148] or few instances [127, 159, 126, 107] and have to be retrained for every newly encountered object. However, in domains like service and industrial robotics or augmented reality, it would be beneficial to have a general feature extractor that could produce pose-sensitive features for untrained objects such that testing on a novel object becomes immediately possible.

When trained on few instances, current pose networks like [127, 126] concurrently classify objects which potentially hinders their ability to generalize to untrained objects. Wohlhart et al. [67] and Balntas et al. [85] were the first to report

qualitative results of deep pose descriptors applied to untrained objects. However, their descriptors are discriminated by both, orientation and object class. So if an untrained object has similar appearance from any viewpoint to one of the trained objects, the corresponding descriptor will get corrupted. Unlike [67, 85], our multi-path training strategy does not attempt to separate different object instances in the encoding space and instead allows them to share the same latent features.

Category-level pose estimation [123, 203] can be used to generalize to novel objects from a given category. It assumes that all instances within a category have similar shape and are aligned in a joint coordinate frame. However, these assumptions often do not hold in practice where semantic and geometric similarities oftentimes do not coincide. Assigning aligned coordinate frames can be ambiguous because symmetries of instances can vary within a category. Therefore, in this work, we will not explicitly enforce the alignment within semantic categories and instead leave this decision to the self-supervised, appearance-based training.

CNNs trained on large datasets are frequently used to extract low-level features for downstream tasks, e.g. image retrieval [71] or clustering [47]. A naive baseline to predict the 3D orientation of unknown objects would be to compare feature maps of a network trained on a large dataset like ImageNet or COCO. Unsurprisingly this baseline does not work very well because (1) early features are sensitive to translation while the later layers lost geometric information (2) features strongly differ for synthetic and real object views (3) the dimensionality of feature maps is too high to finely discretize SO(3) while reduction techniques like PCA destroy too much information.

The pose refinement methods [121, 119, 173] iteratively predict rotation and translation residuals between an estimated rendered and a real target view of an object. They were shown to generalize to untrained objects of the same category, and to a less degree even generalize to objects of new categories. These approaches predict an accurate, relative transformation between two object views in a local neighborhood. In contrast, our proposed MP-Encoder method is able to yield both, local relative and global 3D orientation estimates. More recently, Labbe et al. introduced MegaPose [221] that scales the render&compare approach to a large training set for better generalization.

3.2 Vision-based 6-DoF Grasp Generation

As a fundamental problem in robotics, grasping has been studied for decades [21, 14, 49, 192]. We review related literature in the context of data-driven methods as these are the main focus in my PhD thesis.

3.2.1 End-to-end Policy Learning

One line of work for grasping and manipulation of objects employs an end-to-end policy that learns to generate actions from raw input pixel values [76, 115]. This results in a monolithic model that concurrently reasons about perception, planning, grasping, and controlling the robot. A large group of these works learn from interactions of the robot with the environment through reinforcement learning. These approaches have mostly shown promise in bin picking, in (quasi-) planar grasping and in small, insensible workspaces that do not require complicated motion planning in the robot configuration space. Few works [130] have demonstrated iterative 6-DoF grasping approaches with a monolithic policy by combining imitation learning and reinforcement learning. A common drawback of these methods is the limited generalization to novel environments, because the perception and control are learned indirectly at the same time. In addition, these methods are not easily steerable towards grasping a specific object as the reward function encourages grasping any object. In contrast, we will learn to generate diverse 6-DoF grasps on novel objects and scenes for specifiable target objects while just using simulated training data. Additionally, Contact-GraspNet (Sec. 5.1) can be integrated with other perception and motion planning algorithms.

3.2.2 3D Scene Reconstruction

A complete 3D scene reconstruction enables traditional grasp planning. However, learned single-view reconstructions are often ambiguous, coarse and require class-conditioning [158, 129, 186]. Multiple views for 3D scanning are beneficial [190] but not always obtainable in constrained environments like households, take additional time and typically assume a static scene.

3.2.3 Discriminative Methods

Discriminative methods for grasping train a classifier that evaluates the quality of existing grasps [57, 143, 97]. They use different sampling strategies to generate potential candidates. For planar grasping, cross entropy is widely used since it can converge to the final grasp location by iteratively evaluating the quality of grasps in different locations [97]. However, the cross-entropy method does not work well in the higher dimensional 6-DoF grasp space. To overcome the sampling complexity issue, grasp locations are often sampled using geometric heuristics [100, 143].

3.2.4 Generative Methods

Learning-based generative grasp methods aim to overcome the limitations of geometric heuristics and generate meaningful 6-DoF grasps often from experience in a physics simulator [147, 176]. The main challenge is the large, multi-modal search space of 6-DoF grasps. Instead of sampling some potential candidates using heuristics and ranking them, some methods directly predict a per-point graspability score and approach direction in SO(3) space [178, 165, 177]. One problem with predicting approach directions is that they cannot easily capture high curvature areas such as mug rims or handles and also can not represent grasps encompassing hollow structures. Furthermore, successful approach directions are quite ambiguous to learn as multiple ones are possible for a single contact. Therefore, in this work we are aiming to generate stable grasps for unknown objects with full surface contact. We design losses that further improve the convergence by accounting for the discontinuities, imbalance and multi modality of the grasp distribution. Unlike other methods [177], our proposed method is independent of category labels and has no assumption of grasps being always perpendicular to a surface. Instead, we learn a grasp semantic purely from a wide variety of grasp annotated training shapes [194].

4 Contributions: 6D Object Pose Estimation

This chapter presents the publications and additional results on Augmented Autoencoders [124, 125, 181] and Multi-Path Encoders [180]. Furthermore, the motivation and my contributions towards BlenderProc [133, 163], the Benchmark on 6D Object Pose Estimation (BOP) [168, 137] and the combination of object pose estimation with 3D tracking [214] are discussed. All of these efforts have the common goal to enable scalable 6-DoF pose estimation of rigid objects with known geometry from RGB and/or depth images.

4.1 Augmented Autoencoders: Implicit 3D Orientation Learning for 6D Object Detection (ECCV 2018 Best Paper Award, IJCV 2019)

Martin Sundermeyer[1,2], Maximilian Durner[1,2], Zoltan-Csaba Marton[1], Rudolph Triebel [1,2]

[1]German Aerospace Center (DLR), [2]Technical University of Munich (TUM)

4.1.1 Motivation

One of the most important components of modern computer vision systems for applications such as autonomous robotic manipulation and augmented reality is a reliable and fast 6D object detection module. Although, there are very encouraging recent results from [107, 95, 93, 67, 128, 73, 127], a general, easily applicable, robust and fast solution is not available, yet. The reasons for this are manifold. First and foremost, current solutions are often not robust enough against typical challenges such as object occlusions, different kinds of background

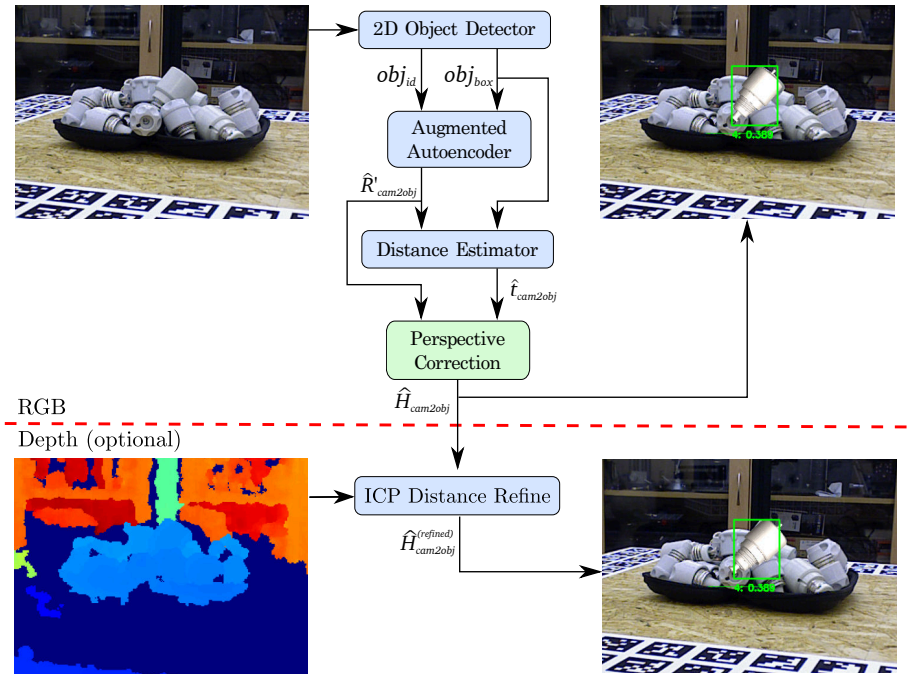


Figure 4.1: Our full 6D Object Detection pipeline: after detecting an object (2D Object Detector), the object is quadratically cropped and forwarded into the proposed Augmented Autoencoder. In the next step, the bounding box scale ratio at the estimated 3D orientation $\hat{R}'_{obj2cam}$ is used to compute the 3D translation $\hat{t}_{obj2cam}$. The resulting euclidean transformation $\hat{H}'_{obj2cam} \in \mathbb{R}^{4x4}$ already shows promising results as presented in [124], however it still lacks of accuracy given a translation in the image plane towards the borders. Therefore, the pipeline is extended by the Perspective Correction block which addresses this problem and results in more accurate 6D pose estimates $\hat{H}_{obj2cam}$ for objects which are not located in the image center. Additionally, given depth data, the result can be further refined ($\hat{H}^{(refined)}_{obj2cam}$) by applying an Iterative Closest Point post-processing (bottom).

clutter, and dynamic changes of the environment. Second, existing methods often require certain object properties such as enough textural surface structure or an asymmetric shape to avoid confusions. And finally, current systems are not efficient in terms of run-time and in the amount and type of annotated training data they require.

Therefore, we propose a novel approach that directly addresses these issues. Concretely, our method operates on single RGB images, which significantly increases the usability as no depth information is required. We note though that depth maps may be incorporated optionally to refine the estimation. As a first step, we build upon 2D Object Detectors ([77, 120]) which provide object bounding boxes and identifiers. On the resulting scene crops, we employ our novel 3D

orientation estimation algorithm, which is based on a previously trained deep network architecture. While deep networks are also used in existing approaches, our approach differs in that we do not explicitly learn from 3D pose annotations during training. Instead, we *implicitly* learn representations from rendered 3D model views. This is accomplished by training a generalized version of the Denoising Autoencoder from [31], that we call *'Augmented Autoencoder (AAE)'*, using a novel Domain Randomization strategy. Our approach has several advantages: First, since the training is independent from concrete representations of object orientations within $SO(3)$ (e.g. quaternions), we can handle ambiguous poses caused by symmetric views because we avoid one-to-many mappings from images to orientations. Second, we learn representations that specifically encode 3D orientations while achieving robustness against occlusion, cluttered backgrounds and generalizing to different environments and test sensors. Finally, the AAE does not require any real pose-annotated training data. Instead, it is trained to encode 3D model views in a self-supervised way, overcoming the need of a large pose-annotated dataset. A schematic overview of the approach based on [124] is shown in Fig 4.1.

Contributions: I proposed and implemented the approach. Maximilian Durner helped with 2D detection results and supervision. Zoltan Csaba Marton and Prof. Rudolph Triebel advised me throughout the process with their domain knowledge and paper writing.

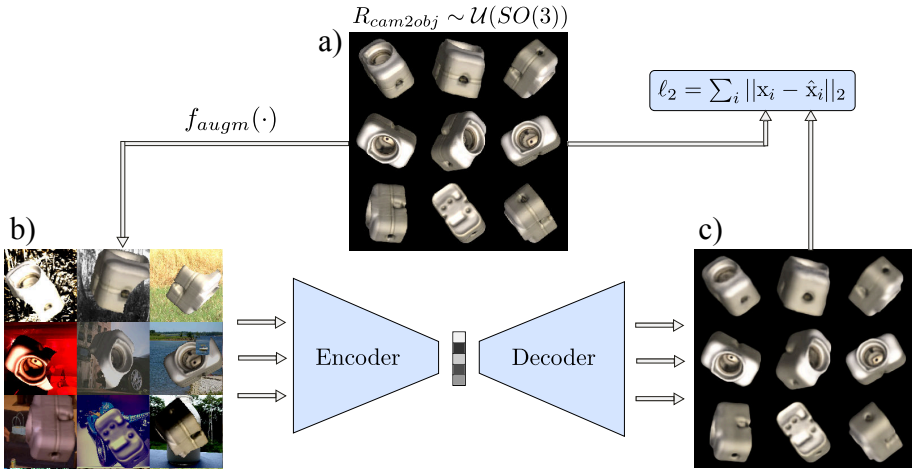


Figure 4.2: Training process for the AAE; a) reconstruction target batch $\boldsymbol{x}$ of uniformly sampled SO(3) object views; b) geometric and color augmented input; c) reconstruction $\hat{\boldsymbol{x}}$ after 40000 iterations

4.1.2 Abstract

We propose a real-time RGB-based pipeline for object detection and 6D pose estimation. Our novel 3D orientation estimation is based on a variant of the Denoising Autoencoder that is trained on simulated views of a 3D model using Domain Randomization.

This so-called Augmented Autoencoder has several advantages over existing methods: It does not require real, pose-annotated training data, generalizes to various test sensors and inherently handles object and view symmetries. Instead of learning an explicit mapping from input images to object poses, it provides an implicit representation of object orientations defined by samples in a latent space. At the time of publication this pipeline achieved state-of-the-art performance on the T-LESS dataset, both, in the RGB and RGB-D domain, and competitive performance with other synthetically trained methods on the LineMOD dataset.

Our code is available here [1]

4.1.3 Method

In the following, we mainly focus on the novel 3D orientation estimation technique based on the AAE.

[1] https://github.com/DLR-RM/AugmentedAutoencoder

4.1.3.1 Autoencoders

The original Autoencoder (AE), introduced by [7], is a dimensionality reduction technique for high dimensional data such as images, audio or depth. It consists of an Encoder Φ and a Decoder Ψ, both arbitrary learnable function approximators which are usually neural networks. The training objective is to reconstruct the input $x \in \mathbb{R}^{\mathcal{D}}$ after passing through a low-dimensional bottleneck, referred to as the latent representation $z \in \mathbb{R}^{n}$ with $n << \mathcal{D}$:

$$\hat{x} = (\Psi \circ \Phi)(x) = \Psi(z) \tag{4.1}$$

The per-sample loss is simply a sum over the pixel-wise L2 distance

$$\ell_2 = \sum_{i \in \mathcal{D}} \parallel x_i - \hat{x}_i \parallel_2 \tag{4.2}$$

The resulting latent space can, for example, be used for unsupervised clustering.

Denoising Autoencoders introduced by [31] have a modified training procedure. Here, artificial random noise is applied to the input images $x \in \mathbb{R}^{\mathcal{D}}$ while the reconstruction target stays clean. The trained model can be used to reconstruct denoised test images. But how is the latent representation affected?

Hypothesis 1: *The Denoising AE produces latent representations which are invariant to noise because this facilitates the reconstruction of de-noised images.*

We will demonstrate that this training strategy actually enforces invariance not only against noise but against a variety of different input augmentations. Finally, it allows us to bridge the domain gap between simulated and real data.

4.1.3.2 Augmented Autoencoder

The motivation behind the AAE is to control what the latent representation encodes and which properties are ignored. We apply random augmentations $f_{augm}(.)$ to the input images $x \in \mathbb{R}^{\mathcal{D}}$ against which the encoding should become invariant. The reconstruction target remains eq. (4.2) but eq. (4.1) becomes

$$\hat{x} = (\Psi \circ \Phi \circ f_{augm})(x) = (\Psi \circ \Phi)(x') = \Psi(z') \tag{4.3}$$

To make evident that **Hypothesis 1** holds for geometric transformations, we learn latent representations of binary images depicting a 2D square at different scales, in-plane translations and rotations. Our goal is to encode only the in-plane

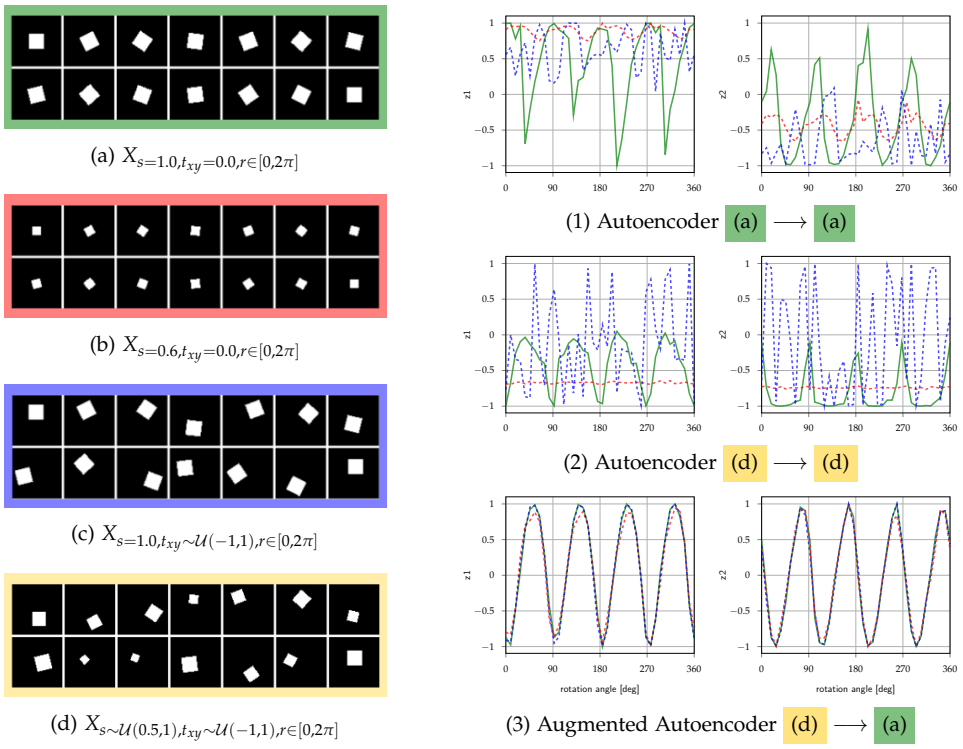


Figure 4.3: Experiment on the dsprites dataset of [98]. Left: 64x64 squares from four distributions (a,b,c and d) distinguished by scale (s) and translation (t_{xy}) that are used for training and testing. Right: Normalized latent dimensions z_1 and z_2 for all rotations (r) of the distribution (a), (b) or (c) after training ordinary AEs (1),(2) and an AAE (3) to reconstruct squares of the same orientation.

rotations $r \in [0, 2\pi]$ in a two dimensional latent space $z \in \mathbb{R}^2$ independent of scale or translation. Fig. 4.3 depicts the results after training a CNN-based AE architecture similar to the model in Fig. 4.4. It can be observed that the AEs trained on reconstructing squares at fixed scale and translation (1) or random scale and translation (2) do not clearly encode rotation alone, but are also sensitive to other latent factors. Instead, the encoding of the AAE (3) becomes invariant to translation and scale such that all squares with coinciding orientation are mapped to the same code. Furthermore, the latent representation is much smoother and the latent dimensions imitate a shifted sine and cosine function with frequency $f = \frac{4}{2\pi}$ respectively. The reason is that the square has two perpendicular axes of symmetry, i.e. after rotating $\frac{\pi}{2}$ the square appears the same. This property of representing the orientation based on the appearance of an object rather than on a fixed parametrization is valuable to avoid ambiguities due to symmetries when teaching 3D object orientations.

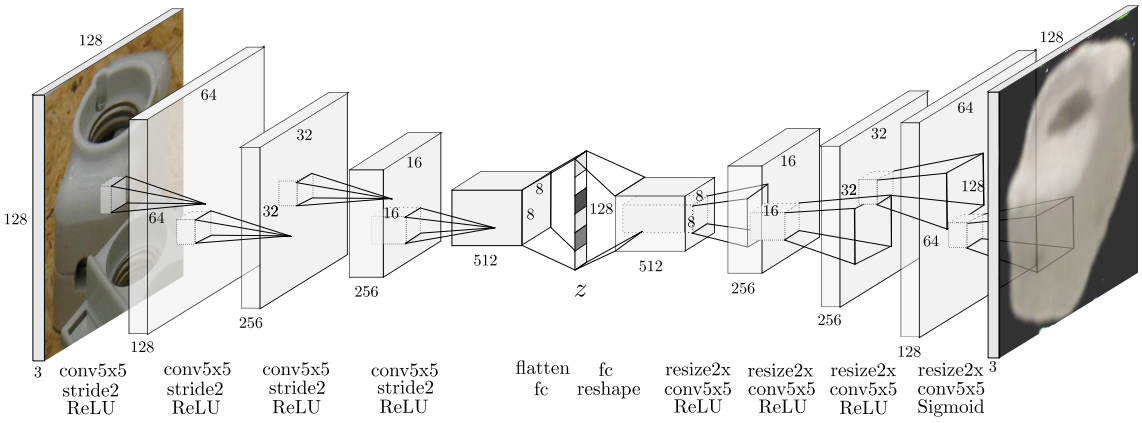


Figure 4.4: Autoencoder CNN architecture with occluded test input, "resize2x" depicts nearest-neighbor upsampling

4.1.3.3 Learning 3D Orientation from Synthetic Object Views

Our toy problem showed that we can explicitly learn representations of object in-plane rotations using a geometric augmentation technique. Applying the same geometric input augmentations we can encode the whole SO(3) space of views from a 3D object model (CAD or 3D reconstruction) while being robust against inaccurate object detections. However, the encoder would still be unable to relate image crops from real RGB sensors because (1) the 3D model and the real object differ, (2) simulated and real lighting conditions differ, (3) the network can't distinguish the object from background clutter and foreground occlusions. Instead of trying to imitate every detail of specific real sensor recordings in simulation we propose a Domain Randomization (DR) technique within the AAE framework to make the encodings invariant to insignificant environment and sensor variations. The goal is that the trained encoder treats the differences to real camera images as just another irrelevant variation. Therefore, while keeping reconstruction targets clean, we randomly apply additional augmentations to the input training views: (1) rendering with random light positions and randomized diffuse and specular reflection (simple Phong model [4] in OpenGL), (2) inserting random background images from the Pascal VOC dataset [37], (3) varying image contrast, brightness, Gaussian blur and color distortions, (4) applying occlusions using random object masks or black squares. Fig. 4.2 depicts an exemplary training process for synthetic views of object 5 from T-LESS [93].

Table 4.1: Augmentation Parameters of AAE; Scale and translation is in relation to image shape and occlusion is in proportion of the object mask

	50% chance (30% per channel)	light (random position) & geometric	
add	$\mathcal{U}(-0.1, 0.1)$	ambient	0.4
contrast	$\mathcal{U}(0.4, 2.3)$	diffuse	$\mathcal{U}(0.7, 0.9)$
multiply	$\mathcal{U}(0.6, 1.4)$	specular	$\mathcal{U}(0.2, 0.4)$
invert		scale	$\mathcal{U}(0.8, 1.2)$
gaussian blur	$\sigma \sim \mathcal{U}(0.0, 1.2)$	translation	$\mathcal{U}(-0.15, 0.15)$
		occlusion	$\in [0, 0.25]$

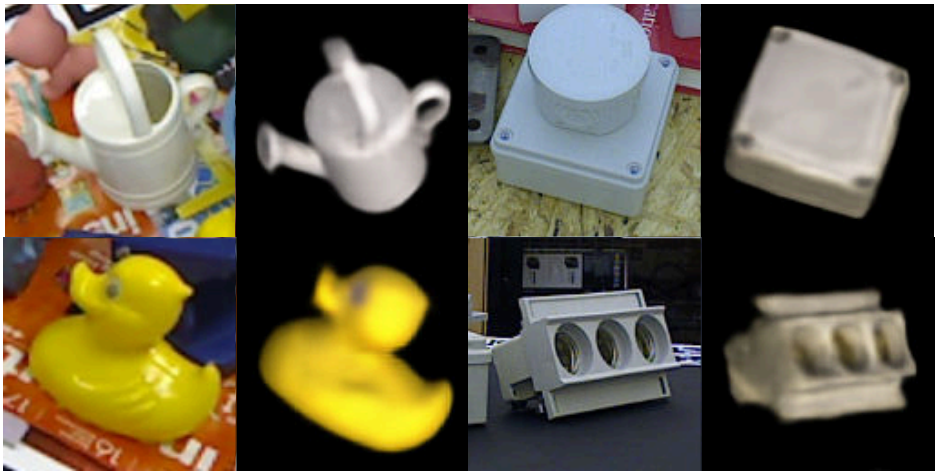


Figure 4.5: AAE decoder reconstruction of LineMOD (left) and T-LESS (right) scene crops

4.1.3.4 Network Architecture and Training Details

The convolutional Autoencoder architecture that is used in our experiments is depicted in Fig. 4.4. We use a bootstrapped pixel-wise L2 loss, first introduced by [84]. Only the pixels with the largest reconstruction errors contribute to the loss. Thereby, finer details are reconstructed and the training does not converge to local minima like reconstructing black images for all views. In our experiments, we choose a bootstrap factor of $k = 4$ per image, meaning that $\frac{1}{4}$ of all pixels contribute to the loss. Using OpenGL, we render 20000 views of each object uniformly at random 3D orientations and constant distance along the camera axis (700mm). The resulting images are quadratically cropped using the longer side of the bounding box and resized (nearest neighbor) to $128 \times 128 \times 3$ as shown in Fig. 4.2. All geometric and color input augmentations besides the rendering with random lighting are applied online during training at uniform random strength, parameters are found in Tab. 4.1. We use the Adam [50] optimizer with a learning rate of 2×10^{-4}, Xavier initialization [28], a batch size = 64 and 40000 iterations

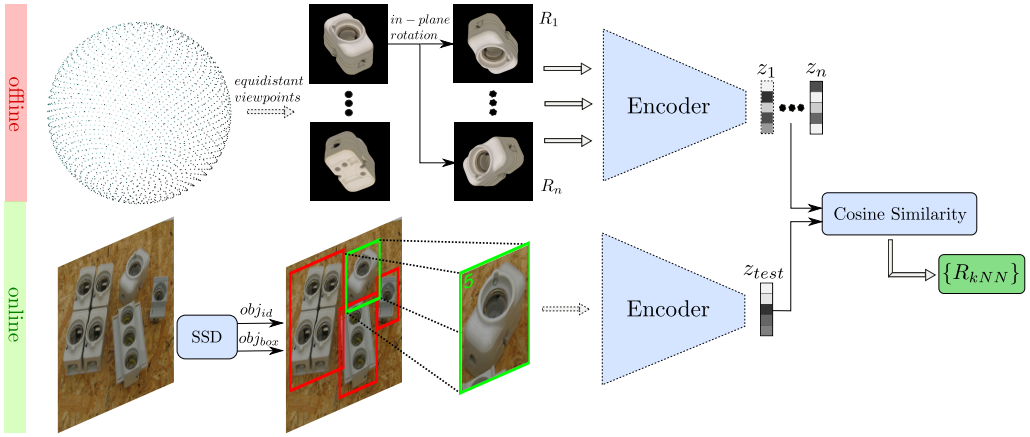


Figure 4.6: Top: creating a codebook from the encodings of discrete synthetic object views; bottom: object detection and 3D orientation estimation using the nearest neighbor(s) with highest cosine similarity from the codebook

which takes $\sim$ 4 hours on a single Nvidia Geforce GTX 1080.

4.1.3.5 Codebook Creation and Test Procedure

After training, the AAE is able to extract a 3D object from real scene crops of many different camera sensors (Fig. 4.5). The clarity and orientation of the decoder reconstruction is an indicator of the encoding quality. To determine 3D object orientations from test scene crops we create a codebook (Fig. 4.6 (top)):

1) Render clean, synthetic object views at nearly equidistant viewpoints from a full view-sphere (based on a refined icosahedron [20])
2) Rotate each view in-plane at fixed intervals to cover the whole SO(3)
3) Create a codebook by generating latent codes $z \in \mathbb{R}^{128}$ for all resulting images and assigning their corresponding rotation $R_{cam2obj} \in \mathbb{R}^{3x3}$

At test time, the considered object(s) are first detected in an RGB scene. The image is quadratically cropped using the longer side of the bounding box multiplied with a padding factor of 1.2 and resized to match the encoder input size. The padding accounts for imprecise bounding boxes. After encoding we compute the cosine similarity between the test code $z_{test} \in \mathbb{R}^{128}$ and all codes $z_i \in \mathbb{R}^{128}$ from the codebook:

$$cos_i = \frac{\boldsymbol{z}_i \, \boldsymbol{z}_{test}}{\|\boldsymbol{z}_i\| \|\boldsymbol{z}_{test}\|} \tag{4.4}$$

The highest similarities are determined in a k-Nearest-Neighbor (kNN) search and the corresponding rotation matrices $\{R_{kNN}\}$ from the codebook are returned

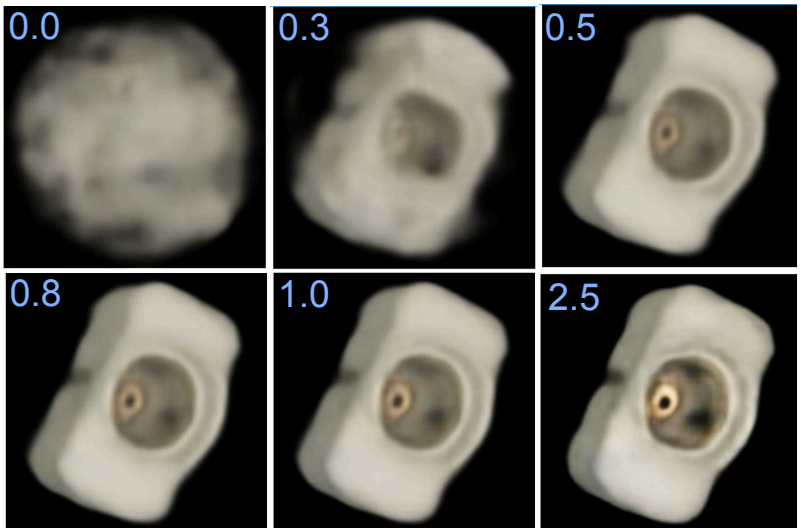


Figure 4.7: AAE decoder reconstruction of a test code $z_{test} \in \mathbb{R}^{128}$ scaled by a factor $s \in [0, 2.5]$

Table 4.2: Augmentation Parameters for Object Detectors, top five are applied in random order; bottom part describes phong lighting from random light positions

	chance (per ch.)	SIXD train	Rendered 3D models
add	0.5 (0.15)	$\mathcal{U}(-0.08, 0.08)$	$\mathcal{U}(-0.1, 0.1)$
contrast norm.	0.5 (0.15)	$\mathcal{U}(0.5, 2.2)$	$\mathcal{U}(0.5, 2.2)$
multiply	0.5 (0.25)	$\mathcal{U}(0.6, 1.4)$	$\mathcal{U}(0.5, 1.5)$
gaussian blur	0.2	$\sigma \sim \mathcal{U}(0.5, 1.0)$	$\sigma = 0.4$
gaussian noise	0.1 (0.1)	$\sigma = 0.04$	-
ambient	1.0	-	0.4
diffuse	1.0	-	$\mathcal{U}(0.7, 0.9)$
specular	1.0	-	$\mathcal{U}(0.2, 0.4)$

as estimates of the 3D object orientation. For the quantitative evaluation we use $k = 1$, however the next neighbors can yield valuable information on ambiguous views and could for example be used in particle filter based tracking. We use cosine similarity because (1) it can be very efficiently computed on a single GPU even for large codebooks. In our experiments we have 2562 equidistant viewpoints $\times$ 36 in-plane rotation = 92232 total entries. (2) We observed that, presumably due to the circular nature of rotations, scaling a latent test code does not change the object orientation of the decoder reconstruction (Fig. 4.7).

4.1.3.6 Extending to 6D Object Detection

Training the 2D Object Detector.

We finetune the 2D Object Detectors using the object views on black background

which are provided in the training datasets of LineMOD and T-LESS. In LineMOD we additionally render domain randomized views of the provided 3D models and freeze the backbone like in [91]. Multiple object views are sequentially copied into an empty scene at random translation, scale and in-plane rotation. Bounding box annotations are adapted accordingly. If an object view is more than 40% occluded, we re-sample it. Then, as for the AAE, the black background is replaced with Pascal VOC images. The randomization schemes and parameters can be found in Table 4.2. In T-LESS we train SSD [77] with VGG16 backbone and RetinaNet [120] with ResNet50 backbone which is slower but more accurate, on LineMOD we only train RetinaNet. For T-LESS we generate 60000 training samples from the provided training dataset and for LineMOD we generate 60000 samples from the training dataset plus 60000 samples from 3D model renderings with randomized lighting conditions (see Table 4.2). The RetinaNet achieves 0.73mAP@0.5IoU on T-LESS and 0.62mAP@0.5IoU on LineMOD. On Occluded LineMOD, the detectors trained on the simplistic renderings failed to achieve good detection performance. However, recent work of [138] quantitatively investigated the training of 2D detectors on synthetic data and they reached decent detection performance on Occluded LineMOD by fine-tuning FasterRCNN on photo-realistic synthetic images showing the feasibility of a purely synthetic pipeline.

Projective Distance Estimation

We estimate the full 3D translation t_{real} from camera to object center, similar to [95]. Therefore, we save the 2D bounding box for each synthetic object view in the codebook and compute its diagonal length $\|bb_{syn,i}\|$. At test time, we compute the ratio between the detected bounding box diagonal $\|bb_{real}\|$ and the corresponding codebook diagonal $\|bb_{syn,\text{argmax}(cos_i)}\|$, i.e. at similar orientation. The pinhole camera model yields the distance estimate $\hat{t}_{real,z}$

$$\hat{t}_{real,z} = t_{syn,z} \times \frac{\|bb_{syn,\text{argmax}(cos_i)}\|}{\|bb_{real}\|} \times \frac{f_{real}}{f_{syn}} \tag{4.5}$$

with synthetic rendering distance $t_{syn,z}$ and focal lengths f_{real}, f_{syn} of the real sensor and synthetic views. It follows that

$$\boldsymbol{\Delta\hat{t}} = \hat{t}_{real,z}\boldsymbol{K_{real}^{-1}bb_{real,c}} - t_{syn,z}\boldsymbol{K_{syn}^{-1}bb_{syn,c}} \tag{4.6}$$

$$\boldsymbol{\hat{t}_{real}} = \boldsymbol{t_{syn}} + \boldsymbol{\Delta\hat{t}} \tag{4.7}$$

where $\boldsymbol{\Delta\hat{t}}$ is the estimated vector from the synthetic to the real object center, $\boldsymbol{K_{real}}, \boldsymbol{K_{syn}}$ are the camera matrices, $\boldsymbol{bb_{real,c}}, \boldsymbol{bb_{syn,c}}$ are the bounding box centers in homogeneous coordinates and $\boldsymbol{\hat{t}_{real}}, \boldsymbol{t_{syn}} = (0, 0, t_{syn,z})$ are the translation vectors from camera to object centers. In contrast to [95], we can predict the 3D translation for different test intrinsics.

Perspective Correction

While the codebook is created by encoding centered object views, the test image crops typically do not originate from the image center. Naturally, the appearance of the object view changes when translating the object in the image plane at constant object orientation. This causes a noticeable error in the rotation estimate from the codebook towards the image borders. However, this error can be corrected by determining the object rotation that approximately preserves the appearance of the object when translating it to our estimate $\boldsymbol{\hat{t}_{real}}$.

$$\begin{pmatrix} \alpha_x \\ \alpha_y \end{pmatrix} = \begin{pmatrix} -\arctan(\hat{t}_{real,y} / \hat{t}_{real,z}) \\ \arctan(\hat{t}_{real,x} / \sqrt{\hat{t}^2_{real,z} + \hat{t}^2_{real,y}}) \end{pmatrix} \tag{4.8}$$

$$\boldsymbol{\hat{R}_{obj2cam}} = \boldsymbol{R_y}(\alpha_y)\boldsymbol{R_x}(\alpha_x)\boldsymbol{\hat{R}'_{obj2cam}} \tag{4.9}$$

where α_x, α_y describe the angles around the camera axes and $\boldsymbol{R_y}(\alpha_y)\boldsymbol{R_x}(\alpha_x)$ the corresponding rotation matrices to correct the initial rotation estimate $\boldsymbol{\hat{R}'_{obj2cam}}$ from object to camera. The perspective corrections give a notable boost in accuracy as reported in Table 4.7. If strong perspective distortions are expected at test time, the training images x' could also be recorded at random distances as opposed to constant distance. However, in the benchmarks, perspective distortions are minimal and consequently random online image-plane scaling of x' is sufficient.

ICP Refinement

Optionally, the estimate is refined on depth data using a point-to-plane Iterative Closest Point (ICP) approach with adaptive thresholding of correspondences based on [10, 11] taking an average of $\sim 320ms$. The refinement is first applied in direction of the vector pointing from camera to the object where most of the RGB-based pose estimation errors stem from and then on the full 6D pose.

Inference Time

The Single Shot Multibox Detector (SSD) with VGG16 base and 31 classes plus

Table 4.3: Inference time of the RGB pipeline using SSD on CPUs **or** Nvidia 1070 GPU

	4 CPUs	GPU
SSD	-	**∼17ms**
Encoder	∼100ms	**∼5ms**
Cosine Similarity	2.5ms	**1.3ms**
Nearest Neighbor	**0.3ms**	3.2ms
Projective Distance	**0.4ms**	-
	∼24ms	

Table 4.4: Single object pose estimation runtime w/o refinement

Method	fps
[128]	0.2
[69]	2
[75]	2
[102]	4
[95]	12
OURS	13 (RetinaNet)
	42 (SSD)
[126]	50

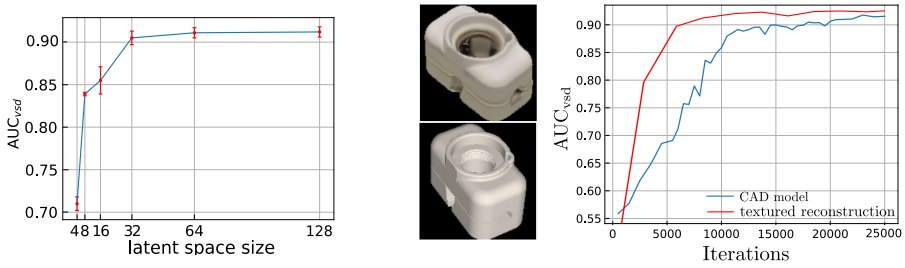
Table 4.5: Ablation study on color augmentations for different test sensors. Object 5 tested on all T-LESS [93] scenes. Standard deviation of three runs in brackets.

Train RGB	**Test RGB**	dyn. light	add	contrast	multiply	invert	**AUC_{vsd}**
3D Reconstruction (synthetic)	Primesense (real)	✓					0.472 (± 0.013)
		✓	✓				0.611 (± 0.030)
		✓	✓	✓			0.825 (± 0.015)
		✓	✓	✓	✓		0.876 (± 0.019)
		✓	✓	✓	✓	✓	**0.877** (± 0.005)
			✓	✓	✓		0.861 (± 0.014)
Primesense (real)	Primesense (real)		✓	✓	✓		0.890 (± 0.003)
3D Reconstruction (synthetic)	Kinect (real)	✓					0.461 (± 0.022)
		✓	✓				0.580 (± 0.014)
		✓	✓	✓			0.701 (± 0.046)
		✓	✓	✓	✓		0.855 (± 0.016)
		✓	✓	✓	✓	✓	0.897 (± 0.008)
			✓	✓	✓		**0.903** (± 0.016)
Kinect (real)	Kinect (real)		✓	✓	✓		0.917 (± 0.007)

the AAE (Fig. 4.4) with a codebook size of 92232×128 yield the average inference times depicted in Table 4.3. We conclude that the RGB-based pipeline is real-time capable at ∼42Hz on a Nvidia GTX 1080. This enables augmented reality and robotic applications and leaves room for tracking algorithms. Multiple encoders (15MB) and corresponding codebooks (45MB each) fit into the GPU memory, making multi-object pose estimation feasible.

4.1.4 Evaluation

We evaluate the AAE and the whole 6D detection pipeline on the T-LESS [93] and LineMOD [32] datasets.



(a) Effect of latent space size, standard deviation in red

(b) Training on CAD model (bottom) vs. textured 3D reconstruction (top)

Figure 4.8: Testing object 5 on all 504 Kinect RGB views of scene 2 in T-LESS

4.1.4.1 Test Conditions

Few RGB-based pose estimation approaches (e.g. [95, 25]) only rely on 3D model information. Most methods like [67, 85, 69] make use of real pose annotated data and often even train and test on the same scenes (e.g. at slightly different viewpoints, as in the official LineMOD benchmark). It is common practice to ignore in-plane rotations or to only consider object poses that appear in the dataset [102, 67] which also limits applicability. Symmetric object views are often individually treated [102, 85] or ignored [67]. The SIXD challenge [92] is an attempt to make fair comparisons between 6D localization algorithms by prohibiting the use of test scene pixels. We follow these strict evaluation guidelines, but treat the harder problem of 6D detection where it is unknown which of the considered objects are present in the scene. This is especially difficult in the T-LESS dataset since objects are very similar. We train the AAEs on the reconstructed 3D models, except for object 19-23 where we train on the CAD models because the pins are missing in the reconstructed plugs.

We noticed, that the geometry of some 3D reconstruction in T-LESS is slightly inaccurate which badly influences the RGB-based distance estimation (Sec. 4.1.3.6) since the synthetic bounding box diagonals are wrong. Therefore, in a second training run we only train on the 30 CAD models.

4.1.4.2 Metrics

Hodan et al. [74] introduced the Visible Surface Discrepancy (err_{vsd}), an ambiguity-invariant pose error function that is determined by the distance between the estimated and ground truth visible object depth surfaces. As in the SIXD challenge, we report the recall of correct 6D object poses at $err_{vsd} < 0.3$ with

Table 4.6: T-LESS: Object recall for $err_{vsd} < 0.3$ on all Primesense test scenes (SIXD/BOP benchmark from [114]). $RGB^{\dagger}$ depicts training with 3D reconstructions, except objects 19-23 $\longrightarrow$ CAD models; $RGB^{\ddagger}$ depicts training on untextured CAD models only

	AAE			**AAE**					**AAE**	
	SSD	RetinaNet		RetinaNet	[69]	[75]	[128]	[26]	w/ GT 2D BBs	
Data	$RGB^{\dagger}$	$RGB^{\dagger}$	$RGB^{\ddagger}$	$RGB^{\dagger}$+Depth(ICP)	RGB-D	RGB-D +ICP	Depth +ICP	Depth +edge	$RGB^{\dagger}$	+Depth(ICP)
1	5.65	9.48	12.67	67.95	8	7	43	53	12.67	85.98
2	5.46	13.24	16.01	70.62	10	10	47	44	11.47	86.27
3	7.05	12.78	22.84	78.39	21	18	69	61	13.32	90.80
4	4.61	6.66	6.70	57.00	4	24	63	67	12.88	84.20
5	36.45	36.19	38.93	77.18	46	23	69	71	67.37	90.14
6	23.15	20.64	28.26	72.75	19	10	67	73	54.21	90.58
7	15.97	17.41	26.56	83.39	52	0	77	75	38.10	86.94
8	10.86	21.72	18.01	78.08	22	2	79	89	24.83	91.79
9	19.59	39.98	33.36	88.64	12	11	90	92	49.06	91.09
10	10.47	13.37	33.15	84.47	7	17	68	72	15.67	84.67
11	4.35	7.78	17.94	56.01	3	5	69	64	16.64	77.01
12	7.80	9.54	18.38	63.23	3	1	82	81	33.57	79.32
13	3.30	4.56	16.20	43.55	0	0	56	53	15.29	64.38
14	2.85	5.36	10.58	25.58	0	9	47	46	50.14	71.37
15	7.90	27.11	40.50	69.81	0	12	52	55	52.01	73.90
16	13.06	22.04	35.67	84.55	5	56	81	85	36.71	87.58
17	41.70	66.33	50.47	74.29	3	52	83	88	81.44	78.88
18	47.17	14.91	33.63	83.12	54	22	80	78	55.48	85.64
19	15.95	23.03	23.03	58.13	38	35	55	55	53.07	82.71
20	2.17	5.35	5.35	26.73	1	5	47	47	38.97	70.87
21	19.77	19.82	19.82	53.48	39	26	63	55	53.45	86.83
22	11.01	20.25	20.25	60.49	19	27	70	56	49.95	84.20
23	7.98	19.15	19.15	62.69	61	71	85	84	36.74	76.40
24	4.74	4.54	27.94	62.99	1	36	70	59	11.75	84.38
25	21.91	19.07	51.01	73.33	16	28	48	47	37.73	87.53
26	10.04	12.92	33.00	67.00	27	51	55	69	29.82	90.26
27	7.42	22.37	33.61	82.16	17	34	60	61	23.30	84.43
28	21.78	24.00	30.88	83.51	13	54	69	80	43.97	89.84
29	15.33	27.66	35.57	74.45	6	86	65	84	57.82	88.58
30	34.63	30.53	44.33	93.65	5	69	84	89	72.81	95.01
Mean	14.67	19.26	**26.79**	**68.57**	17.84	24.60	66.51	67.50	38.34	**84.05**
Time(s)	**0.024**	0.077	0.077	**0.4**	13.5	1.8	4.7	21.5	**0.006**	0.33

tolerance $\tau = 20mm$ and $> 10\%$ object visibility. Although the Average Distance of Model Points (ADD) metric introduced by [40] cannot handle pose ambiguities, we also present it for the LineMOD dataset following the official protocol in [40]. For objects with symmetric views (eggbox, glue), [40] adapts the metric by calculating the average distance to the *closest* model point. Manhardt et al. [121] has noticed inaccurate intrinsics and sensor registration errors between RGB and D in the LineMOD dataset. Thus, purely synthetic RGB-based approaches, although visually correct, suffer from false pose rejections. The focus of our

Table 4.7: Effect of Perspective Corrections on T-LESS

Method	$RGB^{\dagger}$
w/o correction	18.35
w/ correction	**19.26 (+0.91)**

experiments lies on the T-LESS dataset.

In our ablation studies we also report the AUC_{vsd}, which represents the area under the 'err_{vsd} vs. recall' curve:

$$AUC_{vsd} = \int_0^1 recall(err_{vsd})\, derr_{vsd} \tag{4.10}$$

4.1.4.3 Ablation Studies

To assess the AAE alone, in this subsubsection we only predict the 3D orientation of Object 5 from the T-LESS dataset on Primesense and Kinect RGB scene crops. Table 4.5 shows the influence of different input augmentations. It can be seen that the effect of different color augmentations is cumulative. For textureless objects, even the inversion of color channels seems to be beneficial since it prevents overfitting to synthetic color information. Furthermore, training with real object recordings provided in T-LESS with random Pascal VOC background and augmentations yields only slightly better performance than the training with synthetic data. Fig. 4.8a depicts the effect of different latent space sizes on the 3D pose estimation accuracy. Performance starts to saturate at $dim = 64$.

4.1.4.4 Discussion of 6D Object Detection Results

Our RGB-only 6D Object Detection pipeline consists of 2D detection, 3D orientation estimation, projective distance estimation and perspective error correction. Although the results are visually appealing, to reach the performance of state-of-the-art depth-based methods we also need to refine our estimates using a depth-based ICP. Table 4.6 presents our 6D detection evaluation on all scenes of the T-LESS dataset, which contains a high amount of pose ambiguities. Our pipeline outperforms all 15 reported T-LESS results on the 2018 BOP benchmark from [114] in a fraction of the runtime. Table 4.6 shows an extract of competing methods. Our RGB-only results can compete with the RGB-D learning-based approaches of [69] and [75]. Previous state-of-the-art approaches from [128, 26]

Table 4.8: LineMOD: Object recall (ADD [40] metric) of methods that use different amounts of training and test data, results taken from [126]

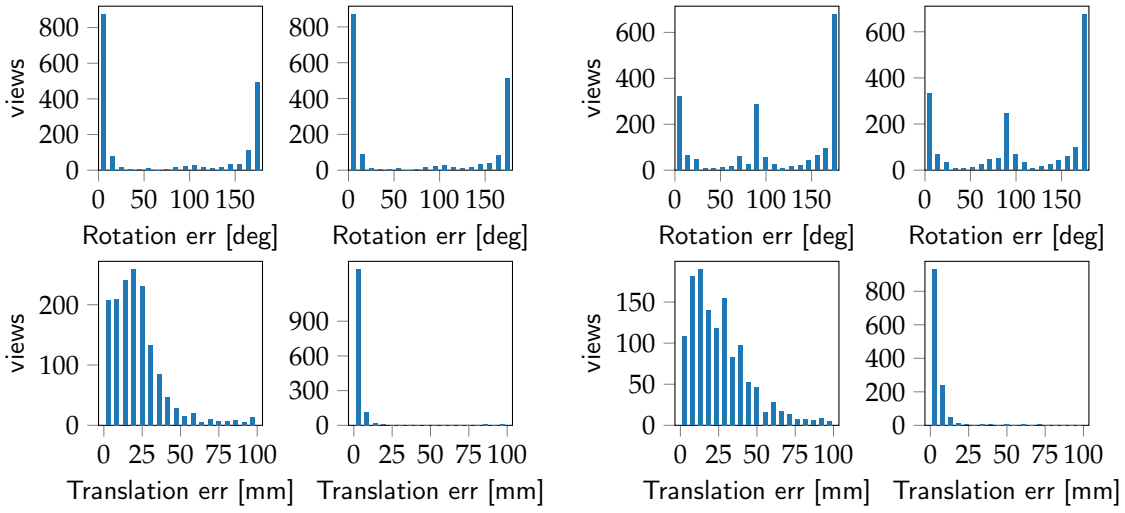
Test data	RGB								+Depth (ICP)	
Train data	RGB w/o real pose labels		RGB with real pose labels						–	
Object	[95]	OURS	[69]		[102]		[126]	[107]	OURS	[95]
			+refine		+refine			+DeepIm		
Ape	0.00	**4.18**	-	33.2	27.9	40.4	21.62	- **77.0**	24.35	**65**
Benchvise	0.18	**22.85**	-	64.8	62.0	91.8	81.80	- **97.5**	**89.13**	80
Cam	0.41	**32.91**	-	38.4	40.1	55.7	36.57	- **93.5**	**82.10**	78
Can	1.35	**37.03**	-	62.9	48.1	64.1	68.80	- **96.5**	70.82	**86**
Cat	0.51	**18.68**	-	42.7	45.2	62.6	41.82	- **82.1**	**72.18**	70
Driller	2.58	**24.81**	-	61.9	58.6	74.4	63.51	- **95.0**	44.87	**73**
Duck	0.00	**5.86**	-	30.2	32.8	44.3	27.23	- **77.7**	54.63	**66**
Eggbox	8.90	**81.00**	-	49.9	40.0	57.8	69.58	- **97.1**	96.62	**100**
Glue	0.00	**46.17**	-	31.2	27.0	41.2	80.02	- **99.4**	94.18	**100**
Holepuncher	0.30	**18.20**	-	52.8	42.4	**67.2**	42.63	- 52.8	**51.25**	49
Iron	8.86	**35.05**	-	80.0	67.0	84.7	74.97	- **98.3**	**77.86**	**78**
Lamp	8.2	**61.15**	-	67.0	39.9	76.5	71.11	- **97.5**	**86.31**	73
Phone	0.18	**36.27**	-	38.1	35.2	54.0	47.74	- **87.7**	**86.24**	79
Mean	2.42	**32.63**	32.3	50.2	43.6	62.7	55.95	62.7 **88.6**	71.58	**79**

perform a time consuming refinement search through multiple pose hypotheses while we only perform the ICP on a single pose hypothesis. That being said, the codebook is well suited to generate multiple hypotheses using $k > 1$ nearest neighbors. The right part of Table 4.6 shows results with ground truth bounding boxes yielding an upper bound on the pose estimation performance.

The results in Table 4.6 show that our domain randomization strategy allows to generalize from 3D reconstructions as well as untextured CAD models as long as the considered objects are not significantly textured. Instead of a performance drop we report an increased $err_{vsd} < 0.3$ recall due to the more accurate geometry of the model which results in correct bounding box diagonals and thus a better projective distance estimation in the RGB-domain.

In Table 4.8 we also compare our pipeline against state-of-the-art methods on the LineMOD dataset. Here, our synthetically trained pipeline does not reach the performance of approaches that use real pose annotated training data.

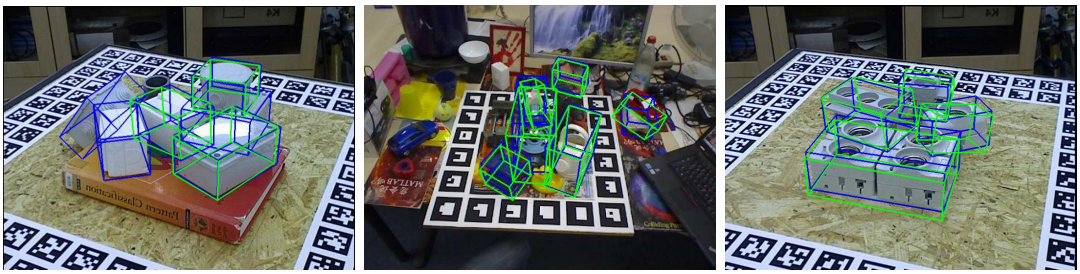
There are multiple issues: (1) As described in Sec 4.1.4.1 the real training and test set are strongly correlated and approaches using the real training set can over-fit to it; (2) the models provided in LineMOD are quite bad which affects both, the detection and pose estimation performance of synthetically trained approaches; (3) the advantage of not suffering from pose-ambiguities does not matter much in LineMOD where most object views are pose-ambiguity free; (4) We train and



(a) Object 5, one view-dependent symmetry

(b) Object 28, two view-dependent symmetries

Figure 4.9: Rotation and translation error histograms on all T-LESS test scenes with our RGB-based (left columns) and ICP-refined (right columns) 6D Object Detection



(a) **(b)** **(c)**

Figure 4.10: Failure cases; Blue: True poses; Green: Predictions; (a) Failed detections due to occlusions and object ambiguity, (b) failed AAE predictions of Glue (middle) and Eggbox (right) due to strong occlusion, (c) inaccurate predictions due to occlusion

test poses from the whole SO(3) as opposed to only a limited range in which the test poses lie. SSD6D also trains only on synthetic views of the 3D models and we outperform their approach by a large margin in the RGB-only domain before ICP refinement.

4.1.4.5 Failure Cases

Figure 4.10 shows qualitative failure cases, mostly stemming from missed detections and strong occlusions. A weak point is the dependence on the bounding box

size at test time to predict the object distance. Specifically, under sever occlusions the predicted bounding box tends to shrink such that it does not encompass the occluded parts of the detected object even if it is trained to do so. If the usage of depth data is clear in advance other methods for directly using depth-based methods for distance estimation might be better suited. Furthermore, on strongly textured objects, the AAEs should not be trained without rendering the texture since otherwise the texture might not be distinguishable from shape at test time. The sim2real transfer on strongly reflective objects like satellites can be challenging and might require physically-based renderings. Some objects, like long, thin pens can fail because their tight object crops at training and test time appear very near from some views and very far from other views, thus hindering the learning of proper pose representations. As the object size is unknown during test time, we cannot simply crop a constantly sized area.

4.1.4.6 Rotation and Translation Histograms

To investigate the effect of ICP and to obtain an intuition about the pose errors, we plot the rotation and translation error histograms of two T-LESS objects (Fig. 4.9). We can see the view-dependent symmetry axes of both objects in the rotation errors histograms. We also observe that the translation error is strongly improved through the depth-based ICP while the rotation estimates from the AAE are hardly refined. Especially when objects are partly occluded, the bounding boxes can become inaccurate and the projective distance estimation (Sec. 4.1.3.6) fails to produce very accurate distance predictions. Still, our global and fast 6D Object Detection provides sufficient accuracy for an iterative local refinement method to reliably converge.

4.1.5 Conclusion

We have proposed a new self-supervised training strategy for Autoencoder architectures that enables robust 3D object orientation estimation on various RGB sensors while training only on synthetic views of a 3D model. By demanding the Autoencoder to revert geometric and color input augmentations, we learn representations that (1) specifically encode 3D object orientations, (2) are invariant to a significant domain gap between synthetic and real RGB images, (3) inherently regard pose ambiguities from symmetric object views. Around this approach, we

created a real-time (42 fps), RGB-based pipeline for 6D object detection which is especially suitable when pose-annotated RGB sensor data is not available.

4.2 Multi-path Learning for Object Pose Estimation Across Domains (CVPR 2020)

Martin Sundermeyer[1,2], Maximilian Durner[1,2], En Yen Puang[1], Zoltan-Csaba Marton[1], Narunas Vaskevicius[3], Kai O. Arras[3], Rudolph Triebel [1,2]

[1]German Aerospace Center (DLR), [2]Technical University of Munich (TUM), [3]Robert Bosch GmbH

4.2.1 Motivation

Learning-based 6D pose estimation methods such as Augmented Autoencoder (AAEs) [124] that were introduced in the previous Section 4.1 are computationally efficient and can show higher robustness towards sensor noise, clutter and environment changes [159, 154] than traditional template or feature based methods [137]. On the other hand, the need for pose-annotated data as well as lengthy training phases make them less flexible. To tackle the lack of annotations, several methods have recently shifted to train on synthetic data rendered from 3D models [95, 124, 127]. However, it is still common to train individual models for every newly encountered instance. This practice hardly scales since training time and inference GPU memory grow linearly with the number of target objects. Attempts to train on many objects at once often result in deteriorated performance [159]. Furthermore, due to object discriminative training, most of these approaches are not suited to generalize to untrained objects. Since the real world consists of large amounts of object categories and instances, we propose a more adaptive and scalable approach in this paper.

Inspired by AAEs [124] that extract pose representative features on an instance level, we propose a single-encoder-multi-decoder network for jointly estimating the 3D object orientation of multiple objects. While the encoder and latent space are shared among all objects, the decoders are trained to reconstruct views of specific instances (top of Fig. 4.11). This *multi-path learning* strategy

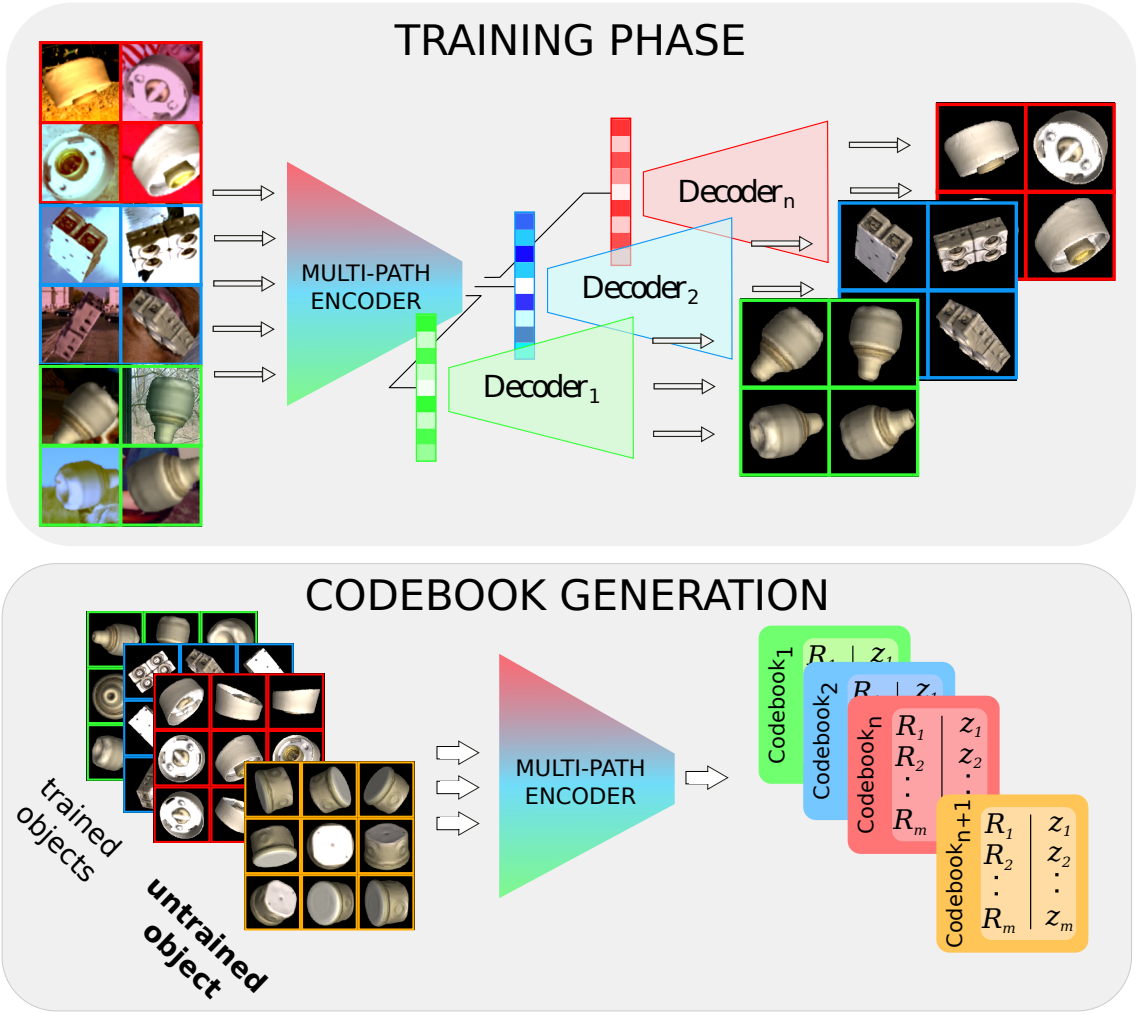


Figure 4.11: Training (top) and setup phase (bottom) of the MP-Encoder. During training one encoder is shared among all objects, while each decoder reconstruct views of a single object. This turns the encoder into a viewpoint-sensitive feature extractor, that generates expressive encodings for multiple trained and even untrained objects.

allows similarly shaped objects to share the same encoding space. After training, we can generate instance-specific codebooks containing encodings of synthetic object views from all over SO(3). So each entry contains a shape and viewpoint dependent descriptor mapped to an explicit 3D object orientation that can be retrieved at test time. (bottom of Fig. 4.11)

As we show experimentally, the learned encodings generalize well to the real domain and are expressive enough to relate views from all trained and even untrained objects in a viewpoint sensitive way. For a large number of objects the performance of the Multi-Path Encoder (MP-Encoder) does not deteriorate compared to separately trained encoders for each object and even slightly improves encoding quality which leads to new state-of-the-art results in 6-DoF object pose estimation on the T-LESS dataset.

Motivated by this, we also introduce an iterative render-inference scheme based on the learned encodings, which enables relative pose estimation on untrained objects. It resembles the online creation of a local codebook which also helps avoiding SO(3) discretization errors. We apply this method to iteratively refine poses of untrained instances from ModelNet40 and outperform DeepIM [119], a state-of-the-art approach for RGB-based 6D pose refinement.

Contributions: I proposed and implemented the approach in the context of a collaboration with the Robert Bosch GmbH. Maximilian Durner helped with 2D detection results and supervision. En Yen Puang implemented the ICP. Dr. Zoltan Csaba Marton and Prof. Rudolph Triebel advised me throughout the process with their domain knowledge and paper writing. Dr. Narunas Vaskevicius helped proof reading and with the rebuttal.

4.2.2 Abstract

We introduce a scalable approach for object pose estimation trained on simulated RGB views of multiple 3D models together. We learn an encoding of object views that does not only describe an implicit orientation of all objects seen during training, but can also relate views of untrained objects. Our single-encoder-multi-decoder network is trained using a technique we denote "multi-path learning": While the encoder is shared by all objects, each decoder only reconstructs views of a single object. Consequently, views of different instances do

not have to be separated in the latent space and can share common features. The resulting encoder generalizes well from synthetic to real data and across various instances, categories, model types and datasets. We systematically investigate the learned encodings, their generalization, and iterative refinement strategies on the ModelNet40 and T-LESS dataset. Despite training jointly on multiple objects, our 6D Object Detection pipeline achieves state-of-the-art results on T-LESS at much lower runtimes than competing approaches. [2]

4.2.3 Method

We will first briefly describe the AAE (Sec. 4.1). Building upon these results, we then propose a novel Multi-Path Encoder-Decoder architecture and training strategy (Sec. 4.2.3.2). Next, we will investigate the encoder on its ability to extract pose-sensitive features and examine generalization to novel instances (Sec. 4.2.3.3). Different application scenarios depending on the test conditions are discussed (Sec. 4.2.3.4). Finally, an iterative render-inference optimization for pose refinements is presented (Sec. 4.2.3.5).

4.2.3.1 Implicit Object Pose Representations

Sundermeyer et al. [124] have shown that implicit pose representations can be learned in a self-supervised way using an encoder-decoder architecture. This so-called AAE allows to encode 3D orientations from arbitrary object views, generalizes from synthetic training data to various test sensors, and inherently handles symmetric object views.

The AAE is trained to reconstruct rendered views of a single object. To encode 3D orientation exclusively, the inputs are randomly translated and scaled while the reconstruction target stays untouched. To encode object views from real images, the background of input views are randomized, occlusions at various locations are generated, and various lighting and color augmentations are produced. As a result of this *domain randomization*, the network learns to represent the objects' orientation of real object views implicitly using the latent code $\mathbf{z}$.

Concretely, an input sample $\mathbf{x} \in \mathbb{R}^d$ is randomly augmented by $f(.)$ and mapped by an encoder Y onto a latent code $\mathbf{z} \in \mathbb{R}^m$ where $m \ll d$. The decoder $\Lambda : \mathbb{R}^m \rightarrow$

[2] Code can be found here: https://github.com/DLR-RM/AugmentedAutoencoder/tree/multipath

$\mathbb{R}^d$ is trained to map the code back to the target $\mathbf{x}$.

$$\hat{\mathbf{x}} = \Lambda(\mathrm{Y}(f(\mathbf{x}))) = \Lambda(\mathrm{Y}(x')) = \Lambda(\mathbf{z}) \tag{4.11}$$

Both Y and Λ are neural networks, and their weight parameters are trained such that the ℓ_2-loss is minimized, i.e.

$$\ell_2(\mathcal{B}) = \sum_{i \in \mathcal{B}} \|\mathbf{x}_i - \hat{\mathbf{x}}_i\|_2 = \sum_{i \in \mathcal{B}} \|\mathbf{x}_i - \Lambda(\mathrm{Y}(f(\mathbf{x}_i)))\|_2 \tag{4.12}$$

where $\mathcal{B}$ contains the indices of input samples of a given batch. After training, the decoder is discarded and latent encodings of object views from all over SO(3) are saved in a codebook together with their corresponding orientations assigned. At test time, a real object crop is encoded and the nearest code(s) in the codebook according to cosine similarity yield the predicted orientation. The rotation can further be corrected for the translational offset as described in [152].

A downside of this formulation is that a new network must be trained for every new object instance. When naively training the original AAE jointly on several objects, they need to be separated in the latent space so that the decoder is able to reconstruct the correct object. Even when conditioning the decoder on the object by concatenating a one-hot vector to the encoding, it can only reconstruct few instances and it diminishes the ability of the encoder to encode object orientations.

4.2.3.2 Multi-Path Encoder-Decoder

We propose a simple but effective architecture which, in combination with our multi-path learning strategy, enables the 3D orientation estimation of multiple objects (see Fig. 4.11). Our architecture consists of a single encoder Y, an encoding $\mathbf{z} \in \mathbb{R}^{128}$, and n decoders Λ_j with $j = 1, ..., n$ where n is the number of different object instances. The convolutional encoder is fed with the same augmented inputs as an AAE but with heterogeneous batches $\tilde{\mathcal{B}}$ containing multiple objects. The resulting codes are split and each decoder only receives codes that correspond to a single object instance. The multi-path loss function can be written as:

$$\begin{aligned} \ell_m(\tilde{\mathcal{B}}) &= \sum_{j=1}^{b} \sum_{k=1}^{n} \mathbb{I}(s_j = k) \|\mathbf{x}_j - \Lambda_k(\mathrm{Y}(f(\mathbf{x}_j)))\|_2 \\ &= \sum_{j=1}^{b} \sum_{k=1}^{n} \mathbb{I}(s_j = k) \|\mathbf{x}_j - \Lambda_k(\mathbf{z}_j)\|_2 \end{aligned} \tag{4.13}$$

where $\mathbb{I}$ is the indicator function used to select the decoder Λ_k that corresponds to the instance s_j. Note that in this setting only the encoder Υ receives information from the entire mini-batch, while the decoders Λ_j backpropagate a loss ℓ_j from a sub-batch. Since the decoders are only used to learn an efficient encoding, they can be discarded after training, leaving behind a compact encoder model.

In contrast to other approaches [67, 85], where objects are explicitly separated in the descriptor space, our encoder can learn an interleaved encoding where general features can be shared across multiple instances. We consider this ability as the main qualification to obtain encodings that can represent object orientations from novel, untrained instances, even when they belong to untrained categories.

4.2.3.3 Principal Component Analysis of Encodings

In order to gain insights into the characteristics of the latent space from trained and untrained objects, we perform an experiment on the ModelNet40 [68] dataset. We first train the multi-path encoder-decoder on 80 CAD instances that originate from the *car* class. A second model is trained on 10 instances from 8 different classes, namely *airplane, bed, bench, car, chair, piano, sink, toilet*. Training details can be found in the appendix.

After training, we generate 72 viewpoints along a full revolution of azimuth, elevation and in-plane rotation. We record different objects from these viewpoints and feed them into the encoder. From the encoding space $z_i \in \mathbb{R}^{128}$ of all objects we compute the first three principal components and project all encodings onto these directions. The interpolated results can be seen in Fig. 4.12. The top row shows the encodings of a car instance from the training set. The other rows show instances which are not in the training set of neither model, but different sofa and toilet instances were used to train the second model.

First, it is noticeable that the encodings vary smoothly along each of the rotation axes, even when views of untrained objects are evaluated. It can be seen that the views, while starting at the same point, end up in different sub-manifolds that capture the shape of the encoded objects and their symmetries. For example, the car instance produces closed trajectories with two revolutions because the car shape looks similar at opposite viewpoints.

Furthermore, it can be seen that the car in the training and test set are encoded similarly by each of the models. This indicates that the encoder as well as

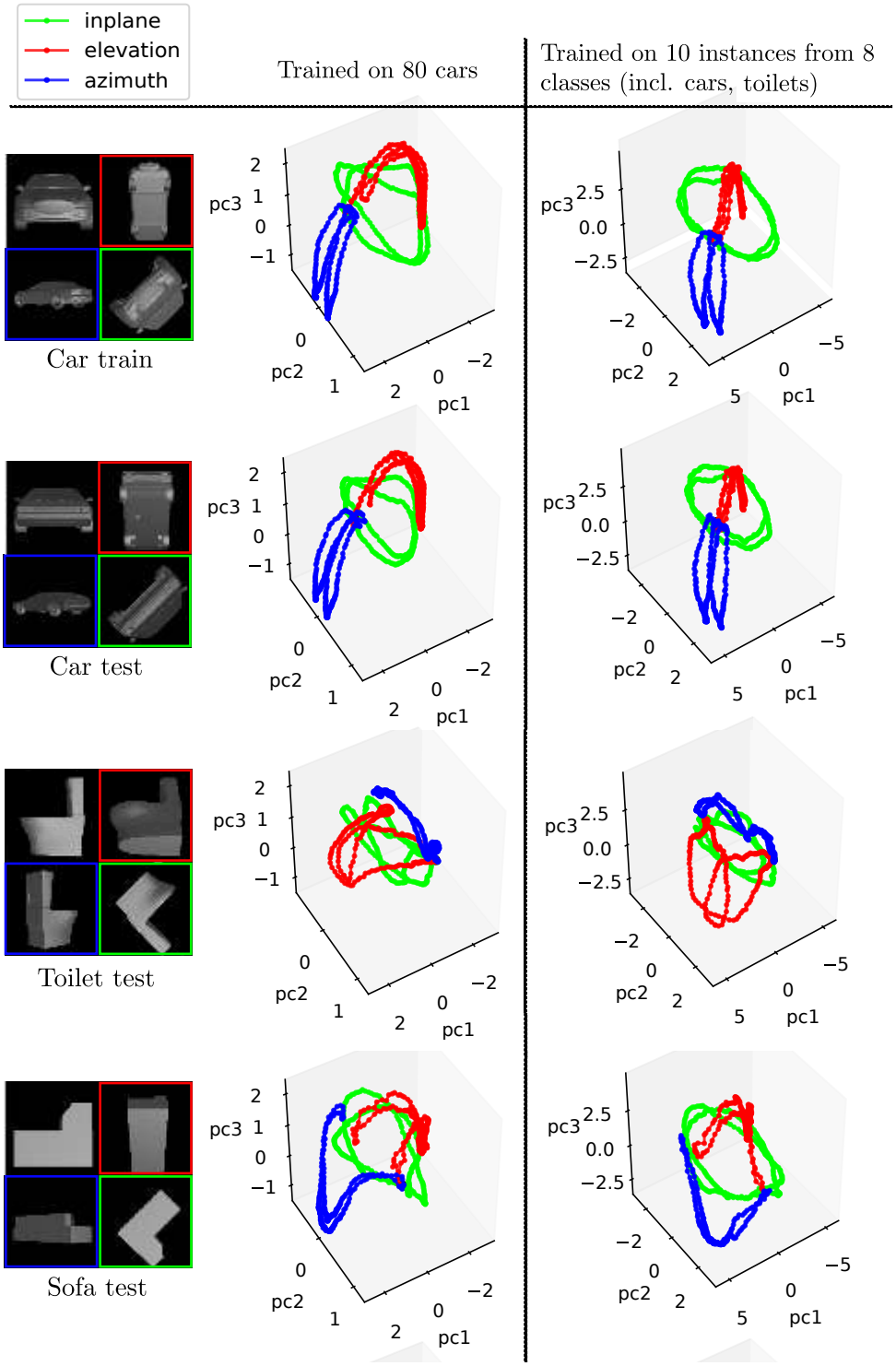


Figure 4.12: Principal Component Analysis of the learned encodings. Depicted are the encodings of views around elevation (red), azimuth (blue), and in-plane (green). Middle column: Encoder trained only on cars; Right column: Encoder trained on objects from 8 categories

Table 4.9: Different application scenarios for global and iterative pose estimation and the handling of untrained objects

	Scenario	
Prerequisites	I	II
Shape category trained	✗	✓
3D Test Model available	✓	✗
3D Test Model aligned to Train Model	✗	✗
Available Methods	⇓	⇓
Reuse codebook from a trained instance	✗	✓
Create a new full / sparse codebook	✓	✗
Iterative relative pose refinement	✓	✗

the codebook could be reused when predicting orientations of novel cars, even without access to their 3D models.

The remaining encodings of untrained objects can still be clearly separated and thus a meaningful pose descriptor could be extracted without retraining the encoder but simply by creating a codebook from their 3D model.

Apart from an offset, both models learn quite similar encodings, even for the sofa category which the first model has not seen during training. This implies that low-level features are extracted that generalize across various shapes and categories. However, to fulfill the reconstruction task the features still need to be robust against lighting and color augmentations as well as translations that are applied to the inputs. In the next section, we will explore different approaches to utilize these properties.

4.2.3.4 Object Pose Estimation Across Domains

After training, the MP-Encoder can create codebooks for all n training instances for pose estimation. (see Sec. 4.2.3.1)

On top of that, it can deal with scenarios inlcuding untrained objects which are depicted in Table 4.9. Here, the trained encoder model is used as a fixed pose-sensitive feature extractor. Available methods depend on the characteristics and given information on the considered test objects.

If a 3D model of the untrained object is available, it is usually preferable to create

Algorithm 1: Iterative 6D Pose Refinement

Input: encoder Y, init pose $\mathbf{q_{init}}, \mathbf{t_{init}}$, target view $\mathbf{x}_*$

$\mathbf{z}_* \leftarrow \mathrm{Y}(\mathbf{x}_*)$

$\mathbf{q_{est}} \leftarrow \mathbf{q_{init}}$

$\mathbf{t_{est}} \leftarrow \mathbf{t_{init}}$

for $k = 0 \dots 2$ **do**

  for $j = 0 \dots 3$ **do**

    for $i = 0 \dots 40 - 10k$ **do**

      $\alpha \sim \mathcal{N}(0, \frac{\sigma^2}{j+1})$

      $\mathbf{v} \sim \mathcal{N}_3(\mathbf{0}, I)$

      $\Delta\mathbf{q} \leftarrow quat(\frac{\mathbf{v}}{\|\mathbf{v}\|}, \alpha)$

      $\mathbf{q_i} \leftarrow \Delta\mathbf{q}\,\mathbf{q_{est}}$

      $\mathbf{x}_i \leftarrow render(\mathbf{q_i}, \mathbf{t_{est}})$

      $\mathbf{z}_i \leftarrow \mathrm{Y}(\mathbf{x}_i)$

    end

    $k \leftarrow \arg\max_i \frac{\mathbf{z}_i\,\mathbf{z}_*}{\|\mathbf{z}_i\|\|\mathbf{z}_*\|}$

    $\mathbf{q_{est}} \leftarrow \mathbf{q_k}$

  end

  $\mathbf{x_{est}} = render(\mathbf{q_{est}}, \mathbf{t_{est}})$

  $\Delta\mathbf{t} = multiScaleEdgeMatching(\mathbf{x}_*, \mathbf{x_{est}})$

  $\mathbf{t_{est}} = \mathbf{t_{est}} + \Delta\mathbf{t}$

end

Result: $\mathbf{q_{est}}, \mathbf{t_{est}}$

a new codebook and estimate the 3D orientation from it (I). If no 3D model is available, but we have codebooks of trained instances from a category with coinciding shapes, we could also reuse these codebooks directly for the test instance (II). However, as we do not explicitly align models of a category in a single coordinate frame, the extracted representation will be more dependent on the shape of an object than on any semantics.

Given a 3D model, we can also use our method for iterative pose refinement at test time. This allows us to refine results from sparser codebooks or to perform local, relative pose estimation directly without any codebook.

Table 4.10: Pose estimation performance of the MP-Encoder trained on objects 1-18 on the complete T-LESS primesense test set (RGB-only) compared to 30 single object AAEs [124] trained on all objects. We measure recall under the err_{vsd} metric. The right column shows the performance of a model trained only on 30 instances of the ModelNet40 dataset. We use ground truth bounding boxes (top) and ground truth masks (bottom) in this experiment. It can be observed that a single MP-Encoder can reach similar performance on unknown objects if segmentation masks are given.

	Mean VSD recall	30 separate AAE encoders [124]	Single Multi-Path Encoder trained on T-Less Objects 1-18	Single Multi-Path Encoder trained on 30 Instances ModelNet40
+gt bbox	Objects 1-18	**35.60**	**35.25**	27.64
	Objects 19-30	**42.45**	33.17	34.57
	Total	**38.34**	34.42	30.41
+gt mask	Objects 1-18	38.98	**43.17**	35.61
	Objects 19-30	**45.33**	43.33	42.59
	Total	41.52	**43.24**	38.40

4.2.3.5 Iterative Refinement of Latent Codes

Our method for iterative pose refinement is outlined in Alg. 1. We start with an initial pose estimate and a target view. Next, we render a batch of 3D model views at small rotational perturbations from the initial pose and insert the whole batch into the encoder. The code with the highest cosine similarity to the target view encoding determines the new rotation estimate. We sample again with a smaller perturbation radius. The inner loop of this random optimization scheme, consisting of rendering and inference, can be efficiently parallelized.

Rotation and translation are alternately optimized three times in our experiments. As the MP-Encoder is trained to be invariant against translations, we can first optimize for the more challenging out-of-plane rotation neglecting translation. Afterwards, the rotation is usually roughly aligned to the target and we use a simple edge-based, multi-scale template matching method based on OpenCV [15] to determine the translational offset. The advantage of sampling in SO(3) vs. sampling in the latent space is that (1) the SO(3) space has lower dimensionality and (2) we only search valid orientations.

Apart from relative pose estimation with a novel 3D model without a codebook, the refinement also allows to trade-off between set-up time, inference time,

memory resources and accuracy. The full 92232×128 codebook creation takes $\sim$ 5 minutes per object on a modern GPU and 45MB space. Inference only takes $\sim$ 6ms on a modern GPU while the 6D pose refinement in the presented configuration takes approximately 1 second. To further speed this up, the random search could be replaced by more sophisticated black-box optimization tools such as CMA-ES [22]. Depending on the application, i.e. number of objects, available resources and constraints on the set-up time, global initialization with sparse codebooks combined with local pose refinement could be a viable option.

4.2.3.6 6-DoF Object Detection Pipeline

Our full RGB-based 6DoF object detection pipeline consists of a MaskRCNN with ResNet50 backbone [90], the MP-Encoder for 3D orientation estimation and a projective distance estimate [124] for translation estimation. To compare against other approaches that use depth data, we also report results with a point-to-plane ICP [10, 11] refinement step. Especially in the presence of severe occlusions the RGB-based projective distance estimate does not produce distances accurate enough to fulfill the strict VSD metric. On the other hand, an accurate initialization is crucial for ICP refinement to work well.

For the MaskRCNN we generate training data by pasting object recordings from the T-LESS training set at random translation, scale and in-plane rotation on random background images. Thereby, we produce 80000 images with MS COCO [51] background, 40000 images with black background and 40000 images with random texture backgrounds [46]. We apply several standard online color augmentations like hue, brightness, blur and contrast. Our MaskRCNN reaches an mAP@0.5 performance of 0.68 (bbox detection) and 0.67 (segmentation). Both, MaskRCNN and the MP-Encoder can process multiple objects at once and thus achieve a runtime that stays nearly constant for a large number of objects.

4.2.4 Evaluation

We focus our analysis on two benchmarks: ModelNet40 [68] and the T-LESS dataset [93].

Table 4.11: Evaluation of our full 6D Object Detection pipeline with MaskRCNN + Multi-Path Encoder + optional ICP. We report the mean VSD recall following the SIXD challenge / BOP benchmark [114] on the T-LESS Primesense test set. See the appendix for object-wise results. A single MP-encoder model outperforms the result of 30 instance-specific AAEs.

	Template matching	Point Pair Features (PPF) based			Learning-based				
	Hodan-15 Depth	Vidal-18 Depth +ICP	Drost-10 Depth	Drost-10-edge Depth + RGB	Brachmann-16 RGB-D	Kehl-16 RGB-D + ICP	**OURS** RGB + ICP	Sundermeyer-18 RGB only	**OURS**
Average	63.18	66.51	56.81	67.5	17.84	24.6	**69.53**	19.26	**20.53**
Time (s)	13.5	4.7	2.3	21.5	4.4	1.8	**0.8**	**0.1**	0.2

4.2.4.1 Metrics

In ModelNet40 we use the absolute angular error

$$e_R = \arccos\left(tr(\hat{R}^T R - I) \,/\, 2\right) \tag{4.14}$$

as well as the ADD metric [40] at an average distance threshold of $0.1\times$ object diameter for the model points $\mathcal{M}$

$$ADD = \frac{1}{m} \sum_{x \in \mathcal{M}} ||(Rx + t) - (\hat{R}x + \hat{t})|| \tag{4.15}$$

and the 2D projection metric at an average 5px threshold

$$Proj2D = \frac{1}{m} \sum_{x \in \mathcal{M}} ||K(Rx + t) - K(\hat{R}x + \hat{t})|| \tag{4.16}$$

In the T-LESS experiments we use the err_{vsd} [74] since here the above metrics are quite meaningless as a result of various object and view symmetries. The err_{vsd} is an ambiguity-invariant pose error metric that is computed from the distance between the estimated and ground truth visible object surfaces. As in the SIXD challenge 2017 [92] and the BOP benchmark 2018 [114], we report the recall of 6D object poses at $err_{vsd} < 0.3$ with tolerance $\tau = 20mm$ and $> 10\%$ object visibility.

4.2.4.2 Generalization Capabilities of the MP-Encoder

We first investigate the joint encoding performance and generalization capabilities of the MP-Encoder in isolation, i.e. with ground truth detections and masks. Therefore, we compare one MP-Encoder against 30 separately trained AAE models on all scenes of the T-LESS Primesense test set (Table 4.10). Equivalent encoder and decoder architectures are used. Furthermore, the MP-Encoder is only

Figure 4.13: Left: Qualitative 6D Object Detection results of the RGB-based pipeline on T-LESS Primesense scenes. Only 18 of 30 objects are used to train the Multi-Path encoder; Middle: ICP-refined results; Right: Relative pose refinement of instances from the untrained categories guitar and bathtub. Red is the initial pose, green depicts the refined pose.

trained on the first 18 3D object reconstructions of the T-LESS dataset to show the generalization capabilities on untrained objects 19-30. On objects 1-18 which the MP-Encoder has seen during training, the results using ground truth bounding boxes are close to the AAE results. Looking at the next row, the performance on the untrained objects 19-30 is significantly worse compared to the single AAEs. We suppose that the MP-Encoder has difficulties to extract unknown objects from the background. This hypothesis is strongly supported by the results with ground truth masks (bottom) where the MP-Encoder even outperforms the AAEs. Even for the untrained objects 19-30 the performance gap to the AAEs that were trained on these objects is quite small. One possible explanation is that a feature extractor trained on multiple objects learns more versatile and thus more robust features.

The last column of Table 4.10 depicts the surprisingly good generalization from ModelNet40 to the T-LESS dataset. Here, the MP-Encoder is specifically trained on 30 texture-free CAD models from the 8 categories *airplane, bed, bench, car, chair, piano, sink, toilet*. Codebooks are created for all T-LESS objects and it is tested on the same real sensor recordings. These results underline that with the multi-path training combined with an input randomization strategy, we can learn to extract orientation variant features that generalize well across greatly differing domains.

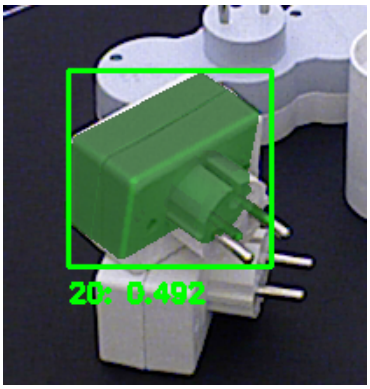


Figure 4.14: Failure case: MaskRCNN predicts the wrong class 20 instead of 19 (the object below). Since the shape is quite similar (except scale) the codebook of object 20 still gives a reasonable pose estimate.

4.2.4.3 6D Object Detection Results

Next, we evaluate our full 6D pose estimation pipeline on the T-LESS dataset which is a particularly challenging 6D object detection benchmark containing texture-less, symmetric objects as well as clutter and severe occlusions. Table 4.11 presents results using the strict vsd metric (Sec. 4.2.3.6). We achieve state-of-the-art results on T-LESS at much lower runtimes than previous methods, both in the RGB domain and also in the depth domain when initializing an ICP with our RGB-based pose estimate. Although the gains are marginal, the results are significant because we only train a single encoder on the whole dataset and still obtain state-of-the-art results. This makes our method scalable, particularly considering that no real pose-annotated data is used for training. Fig. 4.13 (left) shows qualitative examples of the full 6D Object Detection pipeline. Previously, pose estimation of texture-less objects has been dominated by pure depth-based geometric matching approaches that often possess high run-times and do not scale very well with the number of considered objects. Fig. 4.14 shows a failure case which underlines that strict instance-wise pose estimation may not be suitable for real world applications where objects often differ only in details.

4.2.4.4 Iterative Refinement on Untrained Objects

In our final experiment, we evaluate the MP-Encoder on the task of iterative refinement of poses on untrained instances from seen and unseen categories from ModelNet40.

Table 4.12: Relative Pose Refinement from up to 45^o and $\Delta t = (10, 10, 50)[mm]$ perturbations on untrained instances of seen (top) and unseen (bottom) object categories of the ModelNet40 dataset. We measure the recall at the $5cm, 5^o$ threshold, the ADD at 0.1d (object diameter) metric and the Proj2D at 5px threshold.

	metric	(5^o, $5cm$)				6D Pose (ADD)				Proj2D (5px)			
	method	DeepIm [119]		Ours		DeepIm [119]		Ours		DeepIm [119]		Ours	
		init	refined	init	refined	init	refined	init	refined	init	refined	init	refined
novel instance	airplane	0.8	68.9	0.9	**96.9**	25.7	94.7	33.4	**97.9**	0.4	87.3	0.1	**97.4**
	car	1.0	81.5	0.4	**96.4**	10.8	90.7	13.4	**98.5**	0.2	83.9	0.1	**94.0**
	chair	1.0	87.6	0.3	**96.4**	14.6	97.4	16.3	**98.3**	1.5	88.6	0.0	**94.6**
	Mean	0.9	79.3	0.5	**96.6**	17.0	94.3	21.0	**98.2**	0.7	86.6	0.1	**95.3**
novel category	bathtub	0.9	71.6	0.7	**85.5**	11.9	88.6	15.4	**91.5**	0.2	73.4	0.1	**80.6**
	bookshelf	1.2	39.2	0.7	**81.9**	9.2	76.4	13.7	**85.1**	0.1	51.3	0.0	**76.3**
	guitar	1.2	50.4	0.5	**69.2**	9.6	69.6	13.1	**80.5**	0.2	77.1	0.3	**80.1**
	range_hood	1.0	69.8	0.5	**91.0**	11.2	89.6	14.1	**95.0**	0.0	70.6	0.0	**83.9**
	sofa	1.2	82.7	0.6	**91.3**	9.0	89.5	12.2	**95.8**	0.1	**94.2**	0.0	86.5
	wardrobe	1.4	62.7	0.7	**88.7**	12.5	79.4	14.8	**92.1**	0.2	70.0	0.0	**81.1**
	tv_stand	1.2	73.6	0.6	**85.9**	8.8	**92.1**	10.5	90.9	0.2	76.6	0.1	**82.5**
	Mean	1.2	64.3	0.6	**84.8**	10.3	83.6	13.4	**90.1**	0.1	73.3	0.1	**81.6**

We follow the evaluation protocol of DeepIm [119] where the relative pose between two object views of an untrained instance is sought. The target view is rendered at constant translation $t = (0, 0, 500)$ (mm) and random rotation $R \sim SO(3)$. Then we render another object view with that pose perturbed by the angles $\beta_{x/y/z} \sim \mathcal{N}(0, (15^o)^2)$ sampled for each rotation axis and a translational offset $\Delta x \sim \mathcal{N}(0, 10^2)$, $\Delta y \sim \mathcal{N}(0, 10^2)$, $\Delta z \sim \mathcal{N}(0, 50^2)$ (mm). If the total angular perturbation is more than 45^o, it is resampled.

We train category specific MP-Encoders on 80 instances of the airplane, car and chair class and predict the relative pose on novel instances. We also train another MP-Encoder on 80 instances from 8 different categories [3] to obtain a general viewpoint-sensitive feature extractor which we test on instances from novel categories. To retain the relative pose to the target view we use our random optimization scheme described in Alg. 1 with an initial $\sigma = 45^o$.

We compare our approach against DeepIm [119] that also minimizes the relative pose between an initial and a target view through an iterative rendering inference cycle. The results in Table 4.12 demonstrate superior performance of our approach on both, seen and unseen categories. Figure 4.13 on the right shows qualitative results of the refinement on unseen categories.

[3] *airplane, bed, bench, car, chair, piano, sink, toilet*

4.2.5 Conclusion

In this paper we presented a novel approach for estimating poses of multiple trained and untrained objects from a single encoder model. In contrast to other methods, training on multiple objects does not degrade performance and instead leads to state-of-the-art results on the texture-less, symmetric objects of the T-LESS dataset. The same MP-Encoder architecture is also used for iterative pose refinement on untrained objects outperforming previous approaches on ModelNet40.

The ability of this pose estimator to generalize across objects from different categories, datasets and image domains indicates that low-level viewpoint-sensitive features are shared across various domains. Higher-level features for discriminating semantics require viewpoint-invariance and usually generalize less well. Therefore, our results suggest that these two tasks should be learned separately.

We believe that this is a step towards coping with the large number of instances in industrial settings where 3D models are often available and service robotics where categories often appear repeatedly and interactions with novel objects should be immediately possible.

4.3 Benchmark for 6D Object Pose Estimation (BOP)

In the following, I present the motivation, metrics and datasets of the Benchmark for 6D Object Pose Estimation (BOP), a series of public competitions that we organized with the goal to capture and push the state of the art in the field of 6D object pose estimation from an RGB-D image.

4.3.1 Motivation

The field of Computer Vision and Machine Learning has been greatly advanced by the use of benchmarks. One notable example is the ImageNet Challenge [64], which demonstrated the superior performance of Convolutional Neural Networks (CNNs) in object recognition tasks by providing a large training dataset. Until today architectural decisions [72] are validated on ImageNet. Similarly, the COCO benchmark [51] has become popular in the field of 2D object detection and segmentation due to its high-quality mask annotations of common objects in everyday environments. However, a similar benchmark for 6D object pose estimation had yet to be established.

Previously, we have presented two novel approaches for pose estimation, Augmented Autoencoders [181] (Section 4.1) and Multipath-Encoders [180] (Section 4.2), and evaluated them on the LineMOD [32] (Tab. 4.8) and T-LESS [93] (Tab. 4.6, 4.11) datasets. Both methods were trained using synthetic data generated from provided 3D meshes and are capable of estimating orientations in the full SO(3) space.

In contrast, most other methods train on the real pose-annotated images that are provided in popular datasets such LineMOD [32]. The issue is that training images are usually closely related to the respective test images. Especially in LineMOD [32] the training and test images are just slightly varying camera views of the same underlying scenes and cover only a fraction of the full SE(3) space. These constraints greatly simplifies the pose estimation task, but do not fully reflect real world situations where methods need to work outside of the specified test environment and where real, pose-annotated data is usually not available. Other datasets, such as T-LESS, present more realistic challenges such as stronger viewpoint variations and the presence of texture-less and symmetric objects. However, these datasets may not fully capture the complexity of real-world scenarios where other factors, such as lighting, may vary.

To ensure progress in real-world applications, it is crucial to have broad and reliable comparisons between proposed approaches. In the case of 6D object pose estimation, this means evaluating methods on a diverse range of datasets and objects to encompass a broad range of real-world challenges. Furthermore, pose predictions should be evaluated automatically with unified metrics to facilitate comparisons and avoid evaluation errors, varying experimental conditions and cherry-picked metrics. Ideally, only validation sets should be available while test sets remaining private, and parameters should be fixed across different objects and datasets.

The Benchmark for 6D Object Pose Estimation (BOP challenge) [137, 168, 223] aims to address these requirements by providing a unified format for diverse pose-annotated datasets and an online evaluation system where anyone can submit and automatically evaluate their 6D pose predictions. The BOP toolkit [167] supports authors with utility functions to process the datasets and pose estimates. Awards for different training and test settings encourage the development of diverse solutions for various use cases.

4.3.2 Contributions

The BOP challenge [137, 168, 223, 137, 114] is regularly organized in conjunction with the *Recovering 6D Pose Workshop (R6D)* [169] at ECCV / ICCV. In 2019, I joined Tomas Hodan in the organization of the first BOP challenge [137] with current datasets and 6D pose metrics starting with contributions to the evaluation backend, i.e. the BOP toolkit [167]. I also participated in the BOP challenge 2019 myself with the Augmented Autoencoders [124]. For the BOP challenge 2020, I led the extension of BlenderProc to generate physically based scenes for the seven core BOP datasets and rendered 350K photorealistic, synthetic images with annotated poses in BOP format which turned out to be crucial for the success and practicality of learning-based pose estimation as discussed in our BOP 2020 report [168]. In 2022, I was the main organizer of the BOP challenge and R6D Workshop with large support from Tomas Hodan and other organizers. I extended the BOP challenge evaluation to separately measure object detection and segmentation performance with COCO [51] metrics and generated COCO annotations for the BOP datasets. This allowed distinguishing advances in the common object detection/segmentation and pose estimation stages and revealed many new insights discussed next.

		Train. im.		Val im.	Test im.		Test inst.	
Dataset	#Obj	Real	PBR	Real	All	Used	All	Used
LM-O [45]	8	–	50000	–	1214	200	9038	1445
T-LESS [93]	30	37584	50000	–	10080	1000	67308	6423
ITODD [88]	28	–	50000	54	721	721	3041	3041
HB [140]	33	–	50000	4420	13000	300	67542	1630
YCB-V [107]	21	113198	50000	–	20738	900	98547	4123
TUD-L [114]	3	38288	50000	–	23914	600	23914	600
IC-BIN [70]	2	–	50000	–	177	150	2176	1786
LM [38]	15	–	50000	–	18273	3000	18273	3000
RU-APC [80]	14	–	–	–	5964	1380	5964	1380
IC-MI [55]	6	–	–	–	2067	300	5318	800
TYO-L [114]	21	–	–	–	1670	1670	1670	1670
HOPE [217]	28	–	–	50	188	188	3472	2898

Table 4.13: Parameters of the BOP datasets. Most datasets include also training images obtained by OpenGL rendering of the 3D object models on a black background (not shown in the table). PBR training images rendered by BlenderProc [133, 163] are provided for all core datasets since the BOP challenge 2020. If a dataset includes both validation and test images, the ground-truth annotations are public only for the validation images. All test images are real. Column "Test inst./All" shows the number of annotated object instances for which at least 10% of the projected surface area is visible in the test images. Columns "Used" show the number of test images and object instances used in the BOP Challenge 2019, 2020 and 2022.

4.3.3 Datasets

The BOP website[4] currently offers twelve datasets in a unified format, detailed in Tab. 4.13, seven of which were selected as core datasets for the BOP challenge. A method has to be evaluated on all core datasets to proof its robustness and thereby be considered for the main challenge awards. Among those are LineMOD Occluded (LM-O) [45] which presents challenging occlusions and is backwards compatible to methods evaluated in the past. The industrial T-LESS [93] dataset contains texture-less objects with symmetries. HB [140] and YCB-V [107] contain household objects in clutter. IC-BIN [70] presents objects tightly packed in bins and TUD-L contains single objects with severe lighting changes. Finally, ITODD [88] consists of a larger number of mainly metallic, industrial parts recorded by a grayscale camera and a stereo sensor for depth. For an overview, please refer to

[4] https://bop.felk.cvut.cz/datasets/

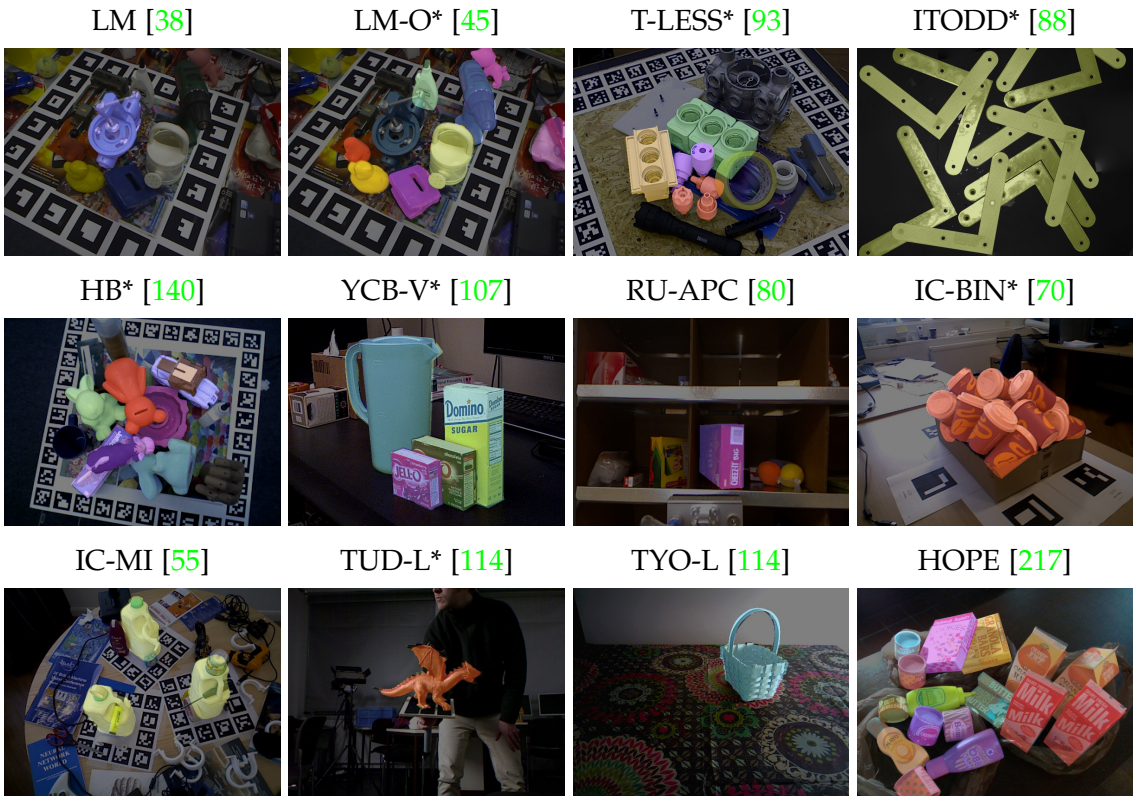


Figure 4.15: An overview of the BOP datasets. The core datasets are marked with a star. Shown are RGB channels of sample test images which were darkened and overlaid with colored 3D object models in the ground-truth 6D poses.

Fig. 4.15.

Each dataset is provided in a unified format and includes 3D object models and training and test RGB-D images annotated with ground-truth 6D object poses. The quality and types of sensor data, 3D meshes and training data vary. The HB and ITODD datasets also include validation images, for which ground-truth poses are publicly available only for the validation images, not for the test images. The object models were created either manually or through the use of KinectFusion-like systems for 3D surface reconstruction [34]. Since BOP 2020, the seven core datasets include a total of 350K photorealistic PBR training images generated and automatically annotated using BlenderProc [133, 163], an open-source physically-based renderer of procedurally generated scenes. A detailed description of the generation process is described in Section 4.5 and an analysis of the importance of PBR data is detailed in Section 4.6. The datasets T-LESS, TUD-L, and YCB-V include real training images, and most datasets include

also training images obtained by OpenGL rendering of the 3D object models on a black background. The test images were captured in scenes of increasing complexity through clutter and occlusions. The datasets can be downloaded from: bop.felk.cvut.cz/datasets.

4.3.4 Evaluation

This section outlines the evaluation methodology which includes the definition of the challenge task, the functions used to measure the error of a 6D pose estimate, and the calculation of the accuracy score used to compare the evaluated methods. The 6D pose localization task has remained the same for the BOP challenges 2019, 2020 and 2022. Therefore, the definitions in this section are based on [168].

4.3.4.1 Task Definition: 6D Pose Localization

All Submissions are evaluated on the task of 6D localization of a **v**arying number of **i**nstances of a **v**arying number of **o**bjects (ViVo) from a single RGB-D image.

Training input: For each object o_i $\forall i \in \{1, \ldots, k\}$, a method is given a 3D mesh model of the object which can include textures or vertex colors and a set of synthetic or real RGB-D images showing instances of the object annotated with 6D poses.

Test input: A method is provided with an image I and a list of instances $L = [(o_1, n_1), \ldots, (o_m, n_m)]$, where n_i is the number of instances of the object o_i present in I.

Test output: The method produces a list $E = [E_1, \ldots, E_m]$, where E_i is a list of n_i pose estimates for instances of the object o_i. Each estimate is given by a 3×3 rotation matrix R, a 3×1 translation vector t, and a confidence score s. The matrix $H^c_{obj} = [R|t]$ defines a rigid transformation from the 3D coordinate system of the object model to the 3D coordinate system of the camera, as described in Section 2.1.1. In the following, we define the 6D pose of an object as $\mathbf{P} = H^c_{obj}$.

4.3.4.2 Pose-Error Metrics

The choice of a suitable pose-error metric is dependent on the application. In BOP, the error of an estimated pose $\hat{\mathbf{P}}$ with respect to the ground-truth pose $\bar{\mathbf{P}}$ of an object model M is measured by three pose-error functions that depend on symmetry-aware 2D or 3D alignment.

VSD (Visible Surface Discrepancy):

$$e_{\text{VSD}}(\hat{D}, \bar{D}, \hat{V}, \bar{V}, \tau) = \text{avg}_{p \in \hat{V} \cup \bar{V}} \begin{cases} 0 & \text{if } p \in \hat{V} \cap \bar{V} \wedge \left|\hat{D}(p) - \bar{D}(p)\right| < \tau \\ 1 & \text{otherwise} \end{cases} \tag{4.17}$$

The symbols $\hat{D}$ and $\bar{D}$ denote distance maps[5] obtained by rendering the object model M in the estimated pose $\hat{\mathbf{P}}$ and the ground-truth pose $\bar{\mathbf{P}}$ respectively. These distance maps are compared with the distance map D_I of the test image I to obtain the visibility masks $\hat{V}$ and $\bar{V}$, i.e. sets of pixels where the model M is visible in the image I. The parameter τ is a misalignment tolerance.

Compared to [114, 74], estimation of the visibility masks has been modified – an object is now considered visible at pixels with no depth measurements. This modification allows evaluating poses of glossy objects from the ITODD dataset [88] whose surface is not always captured in the depth image channel.

VSD treats poses that are indistinguishable in shape (color is not considered) as equivalent by measuring the misalignment of only the visible part of the object surface. See Sec. 2.2 of [114] for details.

MSSD (Maximum Symmetry-Aware Surface Distance):

$$e_{\text{MSSD}}(\hat{\mathbf{P}}, \bar{\mathbf{P}}, S_M, V_M) = \text{min}_{\mathbf{S} \in S_M} \text{max}_{\mathbf{x} \in V_M} \left\|\hat{\mathbf{P}}\mathbf{x} - \bar{\mathbf{P}}\mathbf{S}\mathbf{x}\right\|_2 \tag{4.18}$$

The set S_M contains global symmetry transformations of the object model M, identified as described in Sec. 4.3.4.3, and V_M is a set of the model vertices.

The maximum distance between the model vertices is relevant for robotic manipulation, where the maximum surface deviation strongly indicates the chance of a successful grasp. Moreover, compared to the average distance used in pose-error

[5] A distance map stores at a pixel p the distance from the camera center to a 3D point $\mathbf{x}_p$ that projects to p. It can be readily computed from the depth map which stores at p the Z coordinate of $\mathbf{x}_p$ and which is a typical output of Kinect-like sensors.

functions ADD and ADI [74, 38], the maximum distance is less dependent on the geometry of the object model and the sampling density of its surface.

MSPD (Maximum Symmetry-Aware Projection Distance):

$$e_{\text{MSPD}}(\hat{\mathbf{P}}, \bar{\mathbf{P}}, S_M, V_M) = \min_{\mathbf{S} \in S_M} \max_{\mathbf{x} \in V_M} \left\| \text{proj}(\hat{\mathbf{P}}\mathbf{x}) - \text{proj}(\bar{\mathbf{P}}\mathbf{S}\mathbf{x}) \right\|_2 \tag{4.19}$$

The function $\text{proj}(\cdot)$ is the 2D projection (the result is in pixels) and the meaning of the other symbols is as in MSSD.

Compared to the pose-error function from [45], MSPD considers global object symmetries and replaces the average by the maximum distance to increase robustness against the geometry and sampling of the object model. Since MSPD does not evaluate the alignment along the optical (Z) axis and measures only the perceivable discrepancy, it is relevant for augmented reality applications and suitable for evaluating RGB-only methods, for which estimating the alignment along the optical axis is more challenging.

4.3.4.3 Identifying Global Object Symmetries

The set of global symmetry transformations of an object model M, which is used in the calculation of MSSD and MSPD, is identified in two steps. Firstly, a set of candidate symmetry transformations is defined as $S'_M = \{\mathbf{S} : h(V_M, \mathbf{S}V_M) < \varepsilon\}$, where h is the Hausdorff distance calculated between the vertices V_M of the object model M in the canonical and the transformed pose. The allowed deviation is bounded by $\varepsilon = \max(15\,\text{mm}, 0.1d)$, where d is the diameter of the object model M (the largest distance between any pair of vertices) and the truncation at 15 mm avoids breaking the symmetries by too small details. Secondly, the final set of symmetry transformations S_M is defined as a subset of S'_M which consists of those symmetry transformations that cannot be resolved by the model texture which in ambiguous cases is decided subjectively by us.

The set S_M covers both discrete and continuous global rotational symmetries. The continuous rotational symmetries are discretized such as the vertex which is the furthest from the axis of symmetry travels not more than 1% of the object diameter between two consecutive rotations.

4.3.4.4 Accuracy Score

An estimated pose is considered correct with respect to a pose-error function e, if $e < \theta_e$, where $e \in \{e_{\text{VSD}}, e_{\text{MSSD}}, e_{\text{MSPD}}\}$ and θ_e is a threshold of correctness.

The fraction of annotated object instances for which a correct pose is estimated is referred to as Recall. The Average Recall with respect to a function e, denoted as AR_e, is defined as the average of Recall rates calculated for multiple settings of the threshold θ_e, and also for multiple settings of the misalignment tolerance τ in the case of e_{VSD}. In particular, AR_{VSD} is the average of Recall rates calculated for τ ranging from 5% to 50% of the object diameter with a step of 5%, and for θ_{VSD} ranging from 0.05 to 0.5 with a step of 0.05. AR_{MSSD} is the average of Recall rates calculated for θ_{MSSD} ranging from 5% to 50% of the object diameter with a step of 5%. Finally, AR_{MSPD} is the average of Recall rates calculated for θ_{MSPD} ranging from $5r$ to $50r$ with a step of $5r$, where $r = w/640$ and w is the image width in pixels.

The accuracy of a method on a dataset D is measured by $\text{AR}_D = (\text{AR}_{\text{VSD}} + \text{AR}_{\text{MSSD}} + \text{AR}_{\text{MSPD}})/3$. The overall accuracy on the core datasets is then measured by AR_{Core} defined as the average of the per-dataset AR_D scores. In this way, each dataset is treated as a separate sub-challenge which avoids AR_{Core} being dominated by larger datasets.

4.4 BOP Challenge 2019 (ICCV 2019)

The BOP challenge 2019 [137] was the first challenge based on the current 6D pose metrics described in the previous Section 4.3. Eleven methods were evaluated on all seven core datasets from which six relied on depth-based Point Pair Features (PPFs) [26] and six relied on RGB-based Deep Neural Networks (DNNs) [30] where depth was only used for ICP-based refinements.

The results summarized in Table 4.14 show that methods using the depth image channel, which were mostly based on PPFs [26], clearly outperformed methods relying only on the RGB channels, all of which were based on DNNs. Please refer to Section 3.1.1.4 for an introduction to point pair features. On the other hand, in cluttered scenes these methods need thorough parameter tuning and a large number of point pair features which results in slow runtimes.

#	Method	Type	Data	LM-O	T-LESS	TUD-L	IC-BIN	ITODD	HB	YCB-V	**Avg.**	Time
1	Vidal-Sensors18 [128]	PPF	D	58.2	53.8	87.6	39.3	43.5	70.6	45.0	56.9	3.22
2	Drost-CVPR10-Edges [26]	PPF	RGB-D	51.5	50.0	85.1	36.8	57.0	67.1	37.5	55.0	87.57
3	Drost-CVPR10-3D-Edges [26]	PPF	D	46.9	40.4	85.2	37.3	46.2	62.3	31.6	50.0	80.06
4	Drost-CVPR10-3D-Only [26]	PPF	D	52.7	44.4	77.5	38.8	31.6	61.5	34.4	48.7	7.70
5	Drost-CVPR10-3D-Only-Faster [26]	PPF	D	49.2	40.5	69.6	37.7	27.4	60.3	33.0	45.4	1.38
6	Félix&Neves-ICRA17-IET19 [151, 103]	DNN+PPF	RGB-D	39.4	21.2	85.1	32.3	6.9	52.9	51.0	41.2	55.78
7	Sundermeyer-IJCV19+ICP [152]	DNN	RGB-D	23.7	48.7	61.4	28.1	15.8	50.6	50.5	39.8	0.86
8	Zhigang-CDPN-ICCV19 [142]	DNN	RGB	37.4	12.4	75.7	25.7	7.0	47.0	42.2	35.3	0.51
9	Sundermeyer-IJCV19 [152]	DNN	RGB	14.6	30.4	40.1	21.7	10.1	34.6	37.7	27.0	0.19
10	Pix2Pose-BOP19-ICCV19 [149]	DNN	RGB	7.7	27.5	34.9	21.5	3.2	20.0	29.0	20.5	0.79
11	DPOD (synthetic) [159]	DNN	RGB	16.9	8.1	24.2	13.0	0.0	28.6	22.2	16.1	0.23

Table 4.14: Results of the BOP Challenge 2019. The methods are ranked by the AR_{Core} score (the third column of the upper table) which is the average of the per-dataset AR_D scores (the following seven columns). The scores are defined in Sec. 4.3.4.4. The last column of the upper table shows the average image processing time [s] averaged over the datasets.

4.4.1 Augmented Autoencoder Results

We also trained and submitted pose predictions of our Augmented Autoencoder (AAE) pipeline [181] to the BOP Challenge 2019. The AAEs are trained on the provided 3D models only. We trained RetinaNet [120] for 2D object detection on real or OpenGL rendered object views pasted on random images from PascalVOC [37]. The resulting 2D bounding box predictions are often not accurate enough for distance estimation (Section 4.1.3.6) resulting in reduced VSD and MSSD scores. Therefore we use the depth data to perform a light-weight Iterative Closest Point (ICP) as pose refinement. While this pipeline is not as accurate as the PPF-based methods and takes the 7th place overall with 39.8 AR, it is more efficient which is crucial for Augmented Reality and robotic applications with limited compute capabilities. Consequently, at the BOP challenge 2019 the AAE-based pipeline won the award for the best method with below one second runtime per image.

4.4.2 Discussion

While DNN-based methods can be more efficient, unlike in other vision competitions [51, 41] they were clearly underperforming in the BOP challenge 2019.

We identified the available training data as a potential reason. Just three out of seven BOP challenge datasets provide real training data and only YCB-V and TUD-L provide real training scenes. TUD-L images only contain a single object exposed to severe lighting changes that mainly affect the RGB channel, therefore depth-based PPF methods still perform best on this dataset. However, PPFs ignore the RGB channel which can result in deteriorated detection and pose estimation performance on datasets such as YCB-V where distinguishing many objects from local depth features alone has proven to be difficult. On YCB-V, DNN-based methods such as AAEs [181] with a 2D detector [120] trained on the real training scenes and hybrid methods such as Félix&Neves-ICRA17-IET19 [151, 103] that use RGB-based DNNs for object segmentation in 2D and subsequent PPF-based pose estimation in 3D perform better than pure PPF-based methods.

On the other four BOP datasets without real training data DNNs were trained on synthetic images generated by rendering the 3D object models using OpenGL and pasting them on top of random backgrounds [95, 102, 91, 89, 124]. However, as suggested in [139, 136], the evident domain gap between these "render & paste" training images and real test images presumably limits the potential of DNN-based methods. Real world pose-annotated training datasets are typically not available in realistic applications as they are expensive to collect and limited in their variation and size. Therefore, we need to focus on improving *sim2real* transfer when training on synthetic data generated from 3D meshes. While domain randomization and adaptation techniques can reduce the sim2real gap, the performance is eventually limited by the quality of the synthetic training data. OpenGL rasterizers are fast, but they do not simulate how light rays bounce through the scene, cast shadows and interact with the physically-based materials. The lack of physically plausible object relations, occlusions and clutter and unrealistic backgrounds prevented the success of DNN-based methods in the BOP challenge 2019.

4.5 BlenderProc: Reducing the Reality Gap with Photorealistic Rendering (RSS-W 2020)

In this section, I will motivate and present the open-source project BlenderProc[6], a tool mainly used for realistic synthetic data generation to improve sim2real transfer in Deep Learning applications. This section presents the results published in [163].

4.5.1 Motivation

Today's neural network architectures and optimization schemes result in robust representations if and only if (pre-)trained on large and diverse datasets [53]. For 2D vision tasks we can often label or crowdsource data from the internet and perform transfer learning from public datasets such as ImageNet [64] to improve sample efficiency. However, for 3D vision tasks which are relevant in robotics, the effort of generating large and diverse labeled datasets is magnitudes higher and suitable pretrained models are often unavailable.

Instead of collecting and labeling real world data, it is much more convenient to procedurally generate data in simulation where 3D labels are inherently given. However, although the necessary 3D models are often available, we lack the tools to procedurally assemble them to diverse, plausible scenes, adding realistic lighting and materials and producing photo-realistic images as well as their labels.

As the BOP challenge 2019 (Section 4.4.2) has alluded, Phong-shaded [4] OpenGL images pasted on photographs have limited generalization performance to real test scenes which in turn limits the potential of learning-based pose estimation in general. Other 3D vision domains such as learned 3D scene reconstruction [164], 3D part segmentation and normal estimation have shown similar shortcomings with unrealistic training data. Augmentations and domain adaptation techniques can help to a degree with sim2real transfer [87], but both rely on sufficiently realistic input data. Earlier experiments have demonstrated that photo-realism benefits sim2real transfer, but existing tools are domain or problem specific, non-intuitive to use or closed-source [139].

[6] https://github.com/DLR-RM/BlenderProc

4.5.2 Contributions

In 2019, Maximilian Denninger, Dominik Winkelbauer and me started BlenderProc as a config-based, procedural synthetic data generation pipeline which we built on top of the 3D creation and raytracing suite Blender [111]. While Max and Dominik focused on generating realistic data for single-view 3D reconstruction approaches [164], I focused on object-centric vision tasks such as detection, instance segmentation and pose estimation. I took a major part in integrating rigid body physics, material samplers, object pose samplers, camera intrinsics, aliased and non-aliased depth as well as loaders and writers for common data formats such as COCO [51] and BOP [137]. With the help of Tomas Hodan and my student Dmitry Olefir, we generated the PBR (physically-based rendering) data for the BOP Challenge 2020 & 2022 which will be discussed in the following Section 4.6. I also played a major role in transforming the software from the config-based BlenderProc v1 to the more flexible python API based BlenderProc v2 and together with Wout Boerdijk added loading robots URDF models and computing forward and inverse kinematics which we used to create the Robot Tracking Benchmark [222] described in Section 4.7.1. As of February 2023, BlenderProc has 41 contributors and thousands of users.

4.5.3 Method

Physically-based rendering (PBR) accurately mimics the flow of light in the scene by casting rays and simulating their interaction with 3D geometry and their physically-based materials. This naturally accounts for complex illumination effects such as scattering, refraction and reflection, including diffuse and specular interreflection between the scene elements [79]. The rendered images look realistic and are often difficult to differentiate from real photographs. Rendering techniques based on rasterization, e.g. OpenGL, can approximate the complex effects in an ad hoc way through custom shaders, but the approximations cause physically incorrect artifacts that are difficult to eliminate [60].

BlenderProc makes raytracing capabilities accessible through a flexible Python API that allows procedurally generating 3D scenes and rendering labeled training images from them. A typical run of the pipeline consists of loading a set of objects, sampling object poses with the aid of a physics engine, randomizing object materials, sampling cameras and lights, and rendering modalities such as

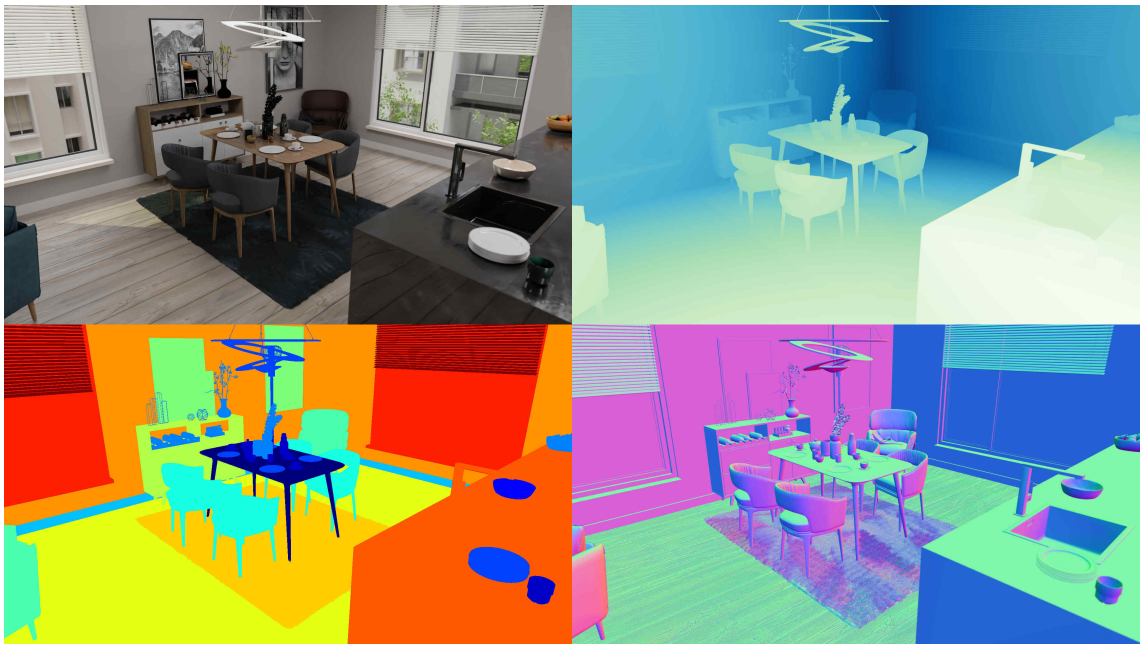


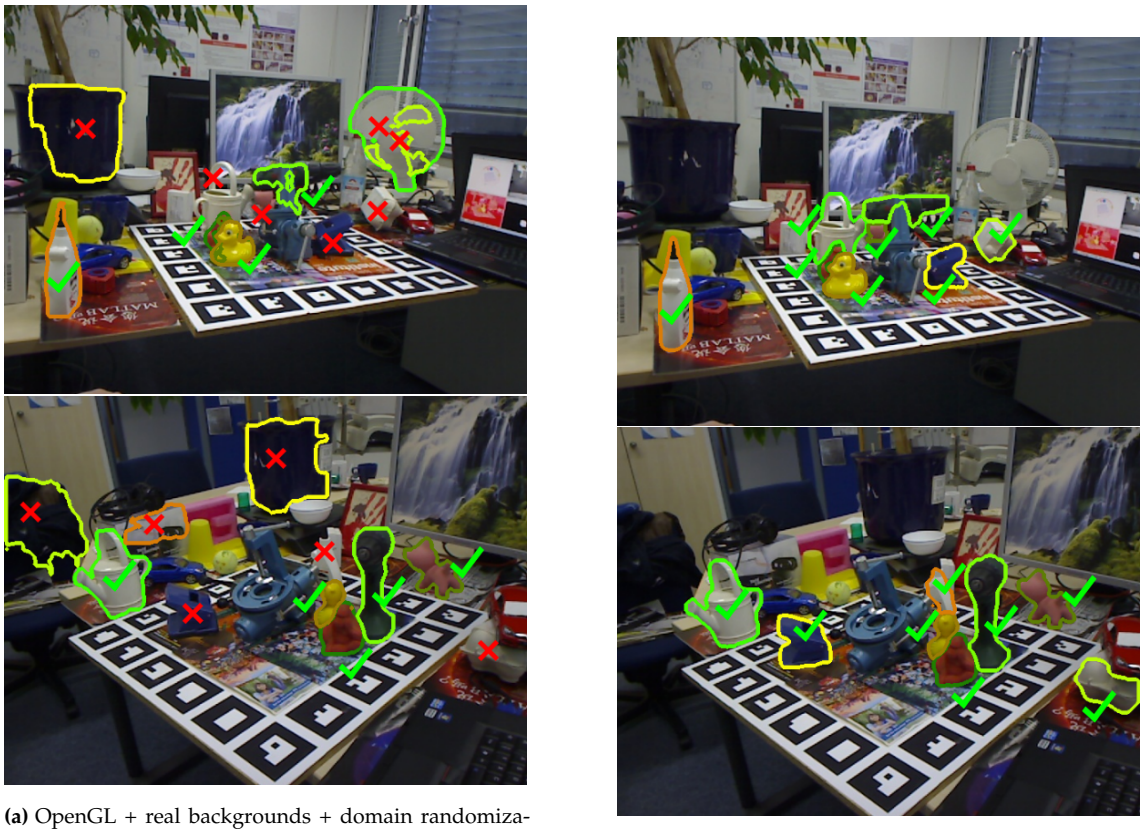
Figure 4.16: BlenderProc renderings of a realistic indoor scene including color, distance, semantic segmentation, and surface normals.

color, depth, surface normals, semantic segmentation or optical flow as shown in Fig. 4.16.

4.5.4 Sim2Real Transfer

We test the sim2real capabilities of data procedurally generated with BlenderProc on the example task of instance segmentation on the LineMOD-Occluded (LM-O) dataset [45]. The dataset provides eight object models with imperfect geometry, texture and without materials acquired with KinectFusion-like techniques. While sim2real transfer benefits from accurately modelled 3D objects placed in realistic 3D environments that imitate test settings [139], the associated modelling effort makes this impractical for most applications. Instead, our goal is to find generic settings that still allow generalizing to various test environments despite imperfect models.

Therefore, we place target and distractor objects into an open cube mapped with randomized PBR textures [161] accounting for collisions. Object material properties like metallicness, roughness and specularity are also randomized. The ceiling emits randomly colored light and a point light source is sampled at random position inside the cube. This *physically-based domain randomization*



(a) OpenGL + real backgrounds + domain randomization (DR).

(b) Training on photorealistic images from BlenderProc.

Figure 4.17: Sim2Real Instance Segmentation on LM-O. Notice that the OpenGL + DR scheme results in missed instances, false positives and failures under strong occlusions. Using the BlenderProc data for training all 8 LM-O instances are segmented well even under challenging conditions.

represents the uncertainties in 3D modelling more realistically than common domain randomization techniques such as color jittering or deformations in the image plane [89]. We sample random camera poses inside a shell within the cube and render 50K synthetic images from these scenes with BlenderProc. An example of the synthetic training data is depicted in Fig. 4.18 on the top right.

To assess sim2real performance, we train a Mask R-CNN [90] with Resnet50 backbone on the BlenderProc data and compare it to training on OpenGL renderings pasted on real backgrounds from COCO [51] combined with strong domain randomization [89], using 50K images as well. Qualitative test results on LM-O are shown in Fig. 4.17.

With default hyperparameters, we achieve a clear improvement from 26.2 (OpenGL) to 33.8 (BlenderProc) Mask mAP50 (+7.6). Note, that the Mask R-CNN models were pretrained on COCO and fine-tuned on the respective datasets. Hinter-

stoisser et al. [91] found that when fine-tuning on the domain-randomized OpenGL data, freezing the complete backbone is necessary to avoid overfitting to the synthetic domain. This is accounted to the domain gap of low-level image statistics between real and OpenGL data. Interestingly, when training on synthetic data generated by BlenderProc we can unfreeze the backbone which even slightly improves results.

Other tasks such as surface normal and depth prediction also showed improved sim2real performance when trained on BlenderProc data, as shown in [163].

These results encouraged the extension of BlenderProc towards generating realistic training data for all seven core datasets of the BOP Challenge 2020 [168] to unfold the potential of learning-based 6D pose estimation. The BOP datasets contain diverse model qualities and test settings which will extend our insights into the factors driving sim2real transfer as described in the next Section 4.6.

4.6 BOP Challenge 2020 & 2022 (ECCV 2020/2022)

The BOP Challenge 2020 & 2022 were organized in conjunction with the 6th/7th International Workshop on Recovering 6D Object Pose at ECCV 2020/2022. They follow the same 6D pose task definitions, evaluation methodology and list of core datasets as the BOP Challenge 2019. However, they differ in the available training data and the evaluated tasks. This section presents the results summarized in the following reports on the BOP challenge 2020 [168] and 2022 [223].

4.6.1 BOP Challenge 2020 (ECCV 2020)

In the BOP challenge 2020 we provided participants with new synthetic training images rendered with BlenderProc to tackle the sim2real gap which limited learning-based methods in 2019 (Sec. 4.4). In the following we explain the procedural training image generation and discuss their effect on detection and pose estimation performance.

4.6.1.1 BlenderProc PBR Training Image Generation

Starting with the BOP Challenge 2020, we provided participants with 50K photorealistic training images for each of the seven core datasets. The images were generated and automatically annotated by BlenderProc [133, 163], an open-source and light-weight physically-based renderer of procedurally generated scenes. Fig. 4.18 depicts examples of the training images.

BlenderProc implements a PBR synthesis approach similar to [139]. However, to improve efficiency, objects are not rendered in 3D models of complete indoor scenes but inside an empty cube, with objects from other BOP datasets serving as distractors. To achieve a rich spectrum of the generated images, a random PBR material from the CC0 Textures library [161] is assigned to the walls of the cube, and light with a random intensity and color is emitted from the room ceiling and from a randomly positioned point source. The number of rays traced per image pixel is set to 50 and the Intel Open Image Denoiser [171] is applied to reduce noise in the rendered image. This setup keeps the computational cost low – the full generation of one 640×480 RGB-D image takes **1–3 seconds** on a standard desktop computer with a modern GPU. A set of 50K images can be therefore rendered on 5 GPU's overnight.

Instead of trying to accurately model the object materials, properties such as specularity, roughness and metallicness are randomized. Such physically plausible domain randomization is important since objects in the challenge as well as in real-world scenarios are typically not modeled perfectly. Realistic object poses are achieved by dropping the 3D object models on the ground plane of the cube using the PyBullet physics engine integrated in Blender [111]. This allows to create dense but shallow piles of objects that introduce various levels of occlusion. Since test images from the LineMOD Occluded (LM-O) dataset show the objects always standing upright, the objects from LM-O are not dropped but instead densely placed on the ground plane in upright poses using automated collision checks.

Each object arrangement is rendered from 25 random camera poses. Instead of fitting all objects within the camera frustum, each camera is pointed at a randomly selected object close to the center, which allows generating more diverse camera poses. The azimuth angles, elevation angles, and distances of the cameras are uniformly sampled from ranges determined by the ground-truth 6D object poses from the test images. In-plane rotation angles are generated randomly.

The generated data (object poses, camera intrinsics, RGB and depth) is saved in the BOP format, allowing to interface with utilities from the BOP toolkit [167]. Open-source code to reproduce or modify the generation process for other models is provided here[7].

4.6.1.2 The Effect of PBR Training Images on BOP 2020

In the BOP challenge 2020, methods based on deep neural networks caught up with methods based on point pair features, which were dominating previously. Out of the 26 evaluated methods, the first-ranked method CosyPose-ECCV20-SYNT+REAL-ICP [173] reaches 69.7 AR and is based on learning-based instance segmentation, pose estimation and pose refinement all of which use RGB images only, with a subsequent multi-hypotheses ICP refinement using the available depth data. Without depth refinement the CosyPose method trained on PBR and real RGB images still reaches the third rank with 63.7 AR and the fifth rank when trained only on the synthetic PBR images, still reaching 57.0 AR. The photorealism of PBR images as well as strong data augmentation were identified as key components of the best performing methods.

[7] github.com/DLR-RM/BlenderProc/blob/master/README_BlenderProc4BOP.md

Figure 4.18: BOP 2020 PBR Training Images. Participants were provided 350K photorealistic training images procedurally generated and rendered using BlenderProc [133, 163].

In 2020, most DNN-based methods were trained either only on the photorealistic (PBR) training images from BlenderProc, or also on real training images which are available in datasets T-LESS, TUD-L, and YCB-V (Tab. 4.13). Two of the CosyPose variants also added the "render & paste" synthetic images provided in the original YCB-V dataset, but these images were later found to have no effect on the accuracy score. Although adding real training images yields higher scores, competitive results can be achieved with PBR images only, as demonstrated by the overall fifth PBR-only variant of the CosyPose method. This is an important result considering that PBR-only training does not require any human effort for capturing and annotating real training images.

The PBR training images yield a noticeable improvement over the "render & paste" synthetic images obtained by OpenGL rendering of the 3D object models on real photographs that were commonly used in the BOP challenge 2019 [137].

As shown in Tab. 4.15, the CosyPose method improved by a significant 57.9% (from 6.1% to 64.0%) on T-LESS, by 19.0% on TUD-L, and by 30.9% on YCB-V when trained on 50K PBR images per dataset vs. 50K "render & paste v1" images per dataset. The "render & paste v1" images used for training CosyPose were obtained by imitating the PBR images, i.e. the 3D object models were rendered in the same poses as in the PBR images and pasted on real backgrounds. The

Method	Detection	Pose estim.	T-LESS	TUD-L	YCB-V
CosyPose	PBR+Real	PBR+Real	72.8	82.3	82.1
	PBR	PBR	64.0	68.5	57.4
	PBR	Render & paste v1	16.1	60.4	44.9
	PBR	Render & paste v2	60.0	58.9	58.5
	Render & paste v1	Render & paste v1	6.1	49.5	26.5
	Render & paste v2	Render & paste v2	45.3	42.4	25.7
AAE	Real	PBR	36.1	49.5	51.5
	Real	Render & paste v0	30.4	40.1	44.6

Table 4.15: The effect of different training images. The table shows the AR_{Core} scores achieved by the CosyPose [173] and Augmented Autoencoder [124] methods when different types of images were used for training their object detection (i.e. Mask R-CNN [90]) and pose estimation stages. The "render & paste v0" images were obtained by OpenGL rendering the 3D object models in random poses pasted on random real photographs. The "render & paste v1" images were obtained similarly but using the same physically plausible object poses as the PBR images. The "render & paste v2" images were obtained similarly, but the CAD models of T-LESS objects were assigned a random surface texture instead of a random gray value, the background of most images was assigned a synthetic texture, and 1M instead of 50K images were generated. The 50K PBR images were obtained by BlenderProc. Interestingly, the increased photorealism brought by the PBR images yields noticeable improvements despite the strong data augmentation applied to the training images.

original Augmented Autoencoder (AAE) entry to BOP (Sec. 4.4.1) was trained on similar images "render & paste v0", but using random object orientations. When training the AAEs on PBR data, we also observe a boost in performance on all three datasets. Similar effects are also shown by the CDPN [142] method shown in Tab. 4.16, indicating the generally better sim2real performance of the PBR data.

As an additional experiment, we have trained the CosyPose method on another variant of the "render & paste" images, generated as in [173] and referred to as "render & paste v2". The main differences compared to the "render & paste v1" variant described in the previous paragraph are: (a) the CAD models of T-LESS objects were assigned a random surface texture instead of a random gray value, (b) the background was assigned a real photograph in 30% images and a synthetic texture in 70% images, and (c) 1M instead of 50K images were generated. As shown in Tab. 4.15, "render & paste v2" images yield a noticeable improvement of 39.2% over "render & paste v1" on T-LESS, but no improvement on TUD-L (-7.1%) and YCB-V (-0.8%). This may suggest that randomizing the surface texture of the texture-less CAD models of T-LESS objects improves the generalization of the network by forcing the network to focus more on shape

than on lower-level patterns, as found in [113]. However, when generating the PBR images, which yield the highest accuracy on T-LESS, the CAD models were assigned a random gray value, as in "render & paste v1", but the effect of randomizing the surface texture may have been achieved by randomizing the PBR material (Sec. 4.6.1.1).

The importance of both the objects and the background being synthetic, as suggested in [136], has not been confirmed in this experiment – "render & paste v1" images with only real backgrounds achieved higher scores than "render & paste v2" images on TUD-L and YCB-V. The benefit of having 1M vs. 50K images is unclear since 50K PBR images were sufficient to achieve high scores.

Both types of "render & paste" images are far inferior compared to the PBR images, which yield an average improvement of 35.9% over "render & paste v1" and 25.5% over "render & paste v2" images for CosyPose(Tab. 4.15). Interestingly, the increased photorealism brought by the PBR images is important despite the strong data augmentation that CosyPose and AAEs apply to the training images. Since object poses in the PBR and "render & paste v1" images are identical, the ray-tracing rendering technique, PBR materials and objects realistically embedded in synthetic environments seem to be the decisive factors for successful "sim2real" transfer [163].

We have also observed that the PBR images are more important for training DNN models for object detection/segmentation (e.g. Mask R-CNN [90]) than for training DNN models for pose estimation from the detected regions (Tab. 4.15). In the case of CosyPose, if the detection model is trained on PBR images and the later two models for pose estimation are trained on the "render & paste v2" instead of the PBR images, the accuracy drops moderately (64.0% to 60.0% on T-LESS, 68.5% to 58.9% on TUD-L) or does not change much (57.4% vs. 58.5% on YCB-V). However, if also the detection model is trained on the "render & paste v1" or "render & paste v2" images, the accuracy drops severely. This shows that photorealism is even more important for distinguishing object instances than distinguishing object viewpoints.

4.6.1.3 Multi-Path Encoder Experiment

In BOP 2020 most pose estimation networks were trained per object or per dataset. To scale 6D pose estimation to a large number of objects, it is necessary to train a single network on many objects at once. The MP-Encoder [180] introduced in

Sec. 4.2 has shown to generalize pose estimation over multiple objects including unseen ones. Given the promising results on 80 ModelNet [68] instances, we attempted for the first time to train a single MP-Encoder on all 108 BOP objects using only RGB data. The 108 decoders were split over 4 GPUs to fit into memory and were discarded after training. The joint MP-Encoder has double the channel size but is otherwise identical to the AAE [181] encoders. The purely RGB-based MP-Encoder reaches 18.6 AR_{Core} which almost equals the 19.6 AR_{Core} of 108 AAEs [181] that were trained separately on every single BOP object. To keep the comparison fair, both AAEs and MP-Encoder were not trained on the newly provided PBR data but the same "render & paste v0" data described in the previous section. While the MP-encoder trained on all BOP objects is not competitive to state-of-the-art approaches, it demonstrates that scaling learning-based object pose estimation to large numbers of objects is in fact possible.

4.6.2 BOP Challenge 2022 (ECCV 2022)

This section presents the results of the BOP Challenge 2022 that are published in [223], compares them with the results from 2019/20, and condensates the main messages for our field.

Starting with the BOP challenge 2022, we additionally evaluate methods on the 2D object detection and 2D object segmentation tasks, using metrics from the COCO challenge [51]. These tasks were introduced to address the design of many recent methods for 6D object pose estimation, which start by detecting/segmenting objects and then estimate the object poses from the predicted regions. Evaluating the detection/segmentation stage and the pose estimation stage separately enables to better understand advances in the two stages and allows researchers to focus on only one of the stages. Participants of the challenge were provided detection and segmentation predictions from Mask R-CNN [90] trained for the first stage of CosyPose [173], the winning method from 2020. These predictions not only serve as baseline results but also as a starting point for participants who want to focus only on the pose estimation stage and, finally, as an opportunity for a direct comparison of pose estimation methods that rely on detections/segmentations.

In total, 49 methods were evaluated on all seven core datasets, see Tab. 4.16 for results and Tab. 4.17 for their settings. Additionally, 9 methods were evaluated on the new object segmentation task (Tab. 4.18) and 8 methods on the new object detection task (Tab. 4.19).

#	Method	LM-O	T-LESS	TUD-L	IC-BIN	ITODD	HB	YCB-V	Avg. Recall	Time
1	GDRNPP-PBRReal-RGBD-MModel [200, 209]	77.5	87.4	96.6	72.2	67.9	92.6	92.1	83.7	6.263
2	GDRNPP-PBR-RGBD-MModel [200, 209]	77.5	85.2	92.9	72.2	67.9	92.6	90.6	82.7	6.264
3	GDRNPP-PBRReal-RGBD-MModel-Fast [200, 209]	79.2	87.2	93.6	70.2	58.8	90.9	83.4	80.5	0.228
4	GDRNPP-PBRReal-RGBD-MModel-OfficialDet [200, 209]	75.8	82.4	96.6	70.8	54.3	89.0	89.6	79.8	6.406
5	Extended_FCOS+PFA-MixPBR-RGBD [206]	79.7	85.0	96.0	67.6	46.9	86.9	88.8	78.7	2.317
6	Extended_FCOS+PFA-MixPBR-RGBD-Fast [206]	79.2	77.9	95.8	67.1	46.0	86.0	88.0	77.1	0.639
7	RCVPose3D-SingleModel-VIVO-PBR [219]	72.9	70.8	96.6	73.3	53.6	86.3	84.3	76.8	1.336
8	ZebraPoseSAT-EffnetB4+ICP(DefaultDet) [215]	75.2	72.7	94.8	65.2	52.7	88.3	86.6	76.5	0.500
9	Extended_FCOS+PFA-PBR-RGBD [206]	79.7	80.2	89.3	67.6	46.9	86.9	82.6	76.2	2.631
10	SurfEmb-PBR-RGBD [205]	76.0	82.8	85.4	65.9	53.8	86.6	79.9	75.8	9.048
11	GDRNPP-PBRReal-RGBD-SModel [200, 209]	75.7	85.6	90.6	68.0	35.6	86.4	81.7	74.8	0.556
12	Coupled Iterative Refinement (CIR) [208]	73.4	77.6	96.8	67.6	38.1	75.7	89.3	74.1	-
13	GDRNPP-PBRReal-RGB-MModel[200, 209]	71.3	78.6	83.1	62.3	44.8	86.9	82.5	72.8	0.229
14	ZebraPoseSAT-EffnetB4 [215]	72.1	80.6	85.0	54.5	41.0	88.2	83.0	72.0	0.250
15	ZebraPoseSAT-EffnetB4(DefaultDet) [215]	70.7	76.8	84.9	59.7	41.7	88.7	81.6	72.0	0.250
16	ZebraPose-SAT [215]	72.1	78.7	86.1	54.9	37.9	84.7	82.8	71.0	-
17	Extended_FCOS+PFA-MixPBR-RGB [206]	74.5	77.8	83.9	60.0	35.3	84.1	80.6	70.9	3.019
18	GDRNPP-PBR-RGB-MModel [200, 209]	71.3	79.6	75.2	62.3	44.8	86.9	71.3	70.2	0.284
19	CosyPose-ECCV20-SYNT+REAL-ICP [173]	71.4	70.1	93.9	64.7	31.3	71.2	86.1	69.8	13.743
20	ZebraPoseSAT-EffnetB4 (PBR_Only) [215]	72.1	72.3	71.7	54.5	41.0	88.2	69.1	67.0	-
21	PFA-cosypose [206, 173]	71.4	73.8	83.7	59.6	24.6	71.2	80.7	66.4	-
22	Extended_FCOS+PFA-PBR-RGB [206]	74.5	71.9	73.2	60.0	35.3	84.1	64.8	66.3	3.497
23	SurfEmb-PBR-RGB [205]	66.3	73.5	71.5	58.8	41.3	79.1	64.7	65.0	8.891
24	Koenig-Hybrid-DL-PointPairs [172]	63.1	65.5	92.0	43.0	48.3	65.1	70.1	63.9	0.633
25	CosyPose-ECCV20-SYNT+REAL-1VIEW [173]	63.3	72.8	82.3	58.3	21.6	65.6	82.1	63.7	0.449
26	CRT-6D	66.0	64.4	78.9	53.7	20.8	60.3	75.2	59.9	0.059
27	Pix2Pose-BOP20_w/ICP-ICCV19 [149]	58.8	51.2	82.0	39.0	35.1	69.5	78.0	59.1	4.844
28	ZTE_PPF	66.3	37.4	90.4	39.6	47.0	73.5	50.2	57.8	0.901
29	CosyPose-ECCV20-PBR-1VIEW [173]	63.3	64.0	68.5	58.3	21.6	65.6	57.4	57.0	0.475
30	Vidal-Sensors18 [128]	58.2	53.8	87.6	39.3	43.5	70.6	45.0	56.9	3.220
31	CDPNv2_BOP20 (RGB-only & ICP) [142]	63.0	46.4	91.3	45.0	18.6	71.2	61.9	56.8	1.462
32	Drost-CVPR10-Edges [26]	51.5	50.0	85.1	36.8	57.0	67.1	37.5	55.0	87.568
33	CDPNv2_BOP20 (PBR-only & ICP) [142]	63.0	43.5	79.1	45.0	18.6	71.2	53.2	53.4	1.491
34	CDPNv2_BOP20 (RGB-only) [142]	62.4	47.8	77.2	47.3	10.2	72.2	53.2	52.9	0.935
35	Drost-CVPR10-3D-Edges [26]	46.9	40.4	85.2	37.3	46.2	62.3	31.6	50.0	80.055
36	Drost-CVPR10-3D-Only [26]	52.7	44.4	77.5	38.8	31.6	61.5	34.4	48.7	7.704
37	CDPN_BOP19 (RGB-only) [142]	56.9	49.0	76.9	32.7	6.7	67.2	45.7	47.9	0.480
38	CDPNv2_BOP20 (PBR-only & RGB-only) [142]	62.4	40.7	58.8	47.3	10.2	72.2	39.0	47.2	0.978
39	leaping from 2D to 6D [175]	52.5	40.3	75.1	34.2	7.7	65.8	54.3	47.1	0.425
40	EPOS-BOP20-PBR [166]	54.7	46.7	55.8	36.3	18.6	58.0	49.9	45.7	1.874
41	Drost-CVPR10-3D-Only-Faster [26]	49.2	40.5	69.6	37.7	27.4	60.3	33.0	45.4	1.383
42	Félix&Neves-ICRA2017-IET2019 [151, 103]	39.4	21.2	85.1	32.3	6.9	52.9	51.0	41.2	55.780
43	Sundermeyer-IJCV19+ICP [152]	23.7	48.7	61.4	28.1	15.8	50.6	50.5	39.8	0.865
44	Zhigang-CDPN-ICCV19 [142]	37.4	12.4	75.7	25.7	7.0	47.0	42.2	35.3	0.513
45	PointVoteNet2 [135]	65.3	0.4	67.3	26.4	0.1	55.6	30.8	35.1	-
46	Pix2Pose-BOP20-ICCV19 [149]	36.3	34.4	42.0	22.6	13.4	44.6	45.7	34.2	1.215
47	Sundermeyer-IJCV19[152]	14.6	30.4	40.1	21.7	10.1	34.6	44.6	28.0	0.196
48	SingleMultiPathEncoder-CVPR20 [180]	21.7	31.0	33.4	17.5	6.7	29.3	28.9	24.1	0.186
49	DPOD (synthetic) [159]	16.9	8.1	24.2	13.0	0.0	28.6	22.2	16.1	0.231

Table 4.16: BOP 2022 results on the seven core datasets. Significant improvements can be observed on all datasets, most notably on the challenging ITODD dataset [88]. However, challenging sensor data, objects and occlusions can still deceive current methods.

#	Method	Year	Type	Models per	Det./seg.	Refinement	Train im.	...type	Test im.
1	GDRNPP-PBRReal-RGBD-MModel [200, 209]	2022	DNN	Object	YOLOX	∼CIR	RGB-D	PBR+real	RGB-D
2	GDRNPP-PBR-RGBD-MModel [200, 209]	2022	DNN	Object	YOLOX	∼CIR	RGB-D	PBR only	RGB-D
3	GDRNPP-PBRReal-RGBD-MModel-Fast [200, 209]	2022	DNN	Object	YOLOX	Depth adjust.	RGB	PBR+real	RGB-D
4	GDRNPP-PBRReal-RGBD-MModel-OfficialDet [200, 209]	2022	DNN	Object	Default (synt+real)	∼CIR	RGB-D	PBR+real	RGB-D
5	Extended_FCOS+PFA-MixPBR-RGBD [206]	2022	DNN	Dataset	Extended FCOS	PFA	RGB	PBR+real	RGB-D
6	Extended_FCOS+PFA-MixPBR-RGBD-Fast [206]	2022	DNN	Dataset	Extended FCOS	PFA	RGB	PBR+real	RGB-D
7	RCVPose3D-SingleModel-VIVO-PBR [219]	2022	DNN	Dataset	RCVPose3D	ICP	RGB-D	PBR+real	RGB-D
8	ZebraPoseSAT-EffnetB4+ICP(DefaultDet) [215]	2022	DNN	Object	Default (synt+real)	ICP	RGB	PBR+real	RGB-D
9	Extended_FCOS+PFA-PBR-RGBD [206]	2022	DNN	Dataset	Extended FCOS	PFA	RGB	PBR only	RGB-D
10	SurfEmb-PBR-RGBD [205]	2022	DNN	Dataset	Default (synt+real)	Custom	RGB-D	PBR only	RGB-D
11	GDRNPP-PBRReal-RGBD-SModel [200, 209]	2022	DNN	Dataset	YOLOX	Depth adjust.	RGB	PBR+real	RGB-D
12	Coupled Iterative Refinement (CIR) [208]	2022	DNN	Object	Default (synt+real)	CIR	RGB-D	PBR+real	RGB-D
13	GDRNPP-PBRReal-RGB-MModel[200, 209]	2022	DNN	Object	YOLOX	-	RGB	PBR+real	RGB
14	ZebraPoseSAT-EffnetB4 [215]	2022	DNN	Object	FCOS	-	RGB	PBR+real	RGB
15	ZebraPoseSAT-EffnetB4(DefaultDet) [215]	2022	DNN	Object	Default (synt+real)	-	RGB	PBR+real	RGB
16	ZebraPose-SAT [215]	2022	DNN	Object	FCOS	-	RGB	PBR+real	RGB
17	Extended_FCOS+PFA-MixPBR-RGB [206]	2022	DNN	Dataset	Extended FCOS	PFA	RGB	PBR+real	RGB
18	GDRNPP-PBR-RGB-MModel [200, 209]	2022	DNN	Object	YOLOX	-	RGB	PBR only	RGB
19	CosyPose-ECCV20-SYNT+REAL-ICP [173]	2020	DNN	Dataset	Default (synt+real)	DeepIm+ICP	RGB	PBR+real	RGB-D
20	ZebraPoseSAT-EffnetB4 (PBR_Only) [215]	2022	DNN	Object	FCOS	-	RGB	PBR only	RGB
21	PFA-cosypose [206, 173]	2022	DNN	Dataset	MaskRCNN	PFA	RGB-D	PBR+real	RGB
22	Extended_FCOS+PFA-PBR-RGB [206]	2022	DNN	Dataset	Extended FCOS	PFA	RGB	PBR only	RGB
23	SurfEmb-PBR-RGB [205]	2022	DNN	Dataset	Default (synt+real)	Custom	RGB	PBR only	RGB
24	Koenig-Hybrid-DL-PointPairs [172]	2020	DNN/PPF	Dataset	Retina/MaskRCNN	ICP	RGB	Synt+real	RGB-D
25	CosyPose-ECCV20-SYNT+REAL-1VIEW [173]	2020	DNN	Dataset	Default (synt+real)	∼DeepIm	RGB	PBR+real	RGB
26	CRT-6D	2022	DNN	Dataset	Default (synt+real)	Custom	RGB	PBR+real	RGB
27	Pix2Pose-BOP20_w/ICP-ICCV19 [149]	2020	DNN	Object	MaskRCNN	ICP	RGB	PBR+real	RGB-D
28	ZTE_PPF	2022	DNN/PPF	Dataset	Default (synt+real)	ICP	RGB	PBR+real	RGB-D
29	CosyPose-ECCV20-PBR-1VIEW [173]	2020	DNN	Dataset	Default (pbr)	∼DeepIm	RGB	PBR only	RGB
30	Vidal-Sensors18 [128]	2019	PPF	-	-	ICP	-	-	D
31	CDPNv2_BOP20 (RGB-only & ICP) [142]	2020	DNN	Object	FCOS	ICP	RGB	Synt+real	RGB-D
32	Drost-CVPR10-Edges [26]	2019	PPF	-	-	ICP	-	-	RGB-D
33	CDPNv2_BOP20 (PBR-only & ICP) [142]	2020	DNN	Object	FCOS	ICP	RGB	PBR only	RGB-D
34	CDPNv2_BOP20 (RGB-only) [142]	2020	DNN	Object	FCOS	-	RGB	Synt+real	RGB
35	Drost-CVPR10-3D-Edges [26]	2019	PPF	-	-	ICP	-	-	D
36	Drost-CVPR10-3D-Only [26]	2019	PPF	-	-	ICP	-	-	D
37	CDPN_BOP19 (RGB-only) [142]	2020	DNN	Object	RetinaNet	-	RGB	Synt+real	RGB
38	CDPNv2_BOP20 (PBR-only & RGB-only) [142]	2020	DNN	Object	FCOS	-	RGB	PBR only	RGB
39	leaping from 2D to 6D [175]	2020	DNN	Object	???	-	RGB	Synt+real	RGB
40	EPOS-BOP20-PBR [166]	2020	DNN	Dataset	-	-	RGB	PBR only	RGB
41	Drost-CVPR10-3D-Only-Faster [26]	2019	PPF	-	-	ICP	-	-	D
42	Félix&Neves-ICRA2017-IET2019 [151, 103]	2019	DNN/PPF	Dataset	MaskRCNN	ICP	RGB-D	Synt+real	RGB-D
43	Sundermeyer-IJCV19+ICP [152]	2019	DNN	Object	RetinaNet	ICP	RGB	Synt+real	RGB-D
44	Zhigang-CDPN-ICCV19 [142]	2019	DNN	Object	RetinaNet	-	RGB	Synt+real	RGB
45	PointVoteNet2 [135]	2020	DNN	Object	-	ICP	RGB-D	PBR only	RGB-D
46	Pix2Pose-BOP20-ICCV19 [149]	2020	DNN	Object	MaskRCNN	-	RGB	PBR+real	RGB
47	Sundermeyer-IJCV19[152]	2019	DNN	Object	RetinaNet	-	RGB	Synt+real	RGB
48	SingleMultiPathEncoder-CVPR20 [180]	2020	DNN	All	MaskRCNN	-	RGB	Synt+real	RGB
49	DPOD (synthetic) [159]	2019	DNN	Dataset	-	-	RGB	Synt	RGB

Table 4.17: BOP 2022 entry setting details. Note that the top 18 results are new submissions all based on DNNs.

In 2020, we provided participants with 350K pre-rendered PBR training images for the seven core datasets as described in Sec. 4.6.1.1. As a result, DNN-based methods achieved noticeably higher accuracy scores when trained on PBR training images than when trained on "render & paste" images. The DNN-based methods finally caught up with the PPF-based methods and the single-view variant of CosyPose [173] reached the best overall performance. However, König and Drost [172], a hybrid method using a DNN for object detection and PPF for pose estimation was still awarded the best method below one second inference time and also performed best on the industrial ITODD dataset [88]. A major goal of the BOP challenge 2022 was therefore to find out whether the gains of DNN-based pose estimation are significant enough to justify their increased deployment complexity. Specifically, are DNN-based methods able to achieve a competitive accuracy also in difficult industrial settings, while training only on synthetic data and while satisfying strict constraints on the inference speed? Do DNN-based methods scale with an increasing number of objects?

4.6.2.1 2D Object Detection and Segmentation Tasks

Training input: At training time, a detection/segmentation method is provided a set of training images showing objects annotated with ground-truth 2D bounding boxes (for the detection task) and binary masks (for the segmentation task). The boxes are *amodal* (covering the whole object silhouette, including the occluded parts) while the masks are *modal* (covering only the visible object part). The method can also use 3D mesh models that are available for the objects (e.g. to synthesize extra training images).

Test input: At test time, the method is given an image showing an arbitrary number of instances of an arbitrary number of objects from a considered dataset. No prior information about the visible object instances is provided.

Test output: The method produces a list of an arbitrary number of amodal 2D bounding boxes (for detection) and modal binary masks (for segmentation) with confidences.

Metrics: Following the evaluation methodology from the COCO 2020 Object Detection Challenge [51], the detection/segmentation accuracy is measured by the

Average Precision (AP). Specifically, for each object we first average the precision at multiple Intersection over Union (IoU) thresholds ($[0.5, 0.55, \ldots, 0.95]$). The AP score is then calculated by averaging over objects from the same dataset and then averaging over the seven core datasets (Sec. 4.3.3).

Analogous to the 6D localization task, only object instances for which at least 10% of the projected surface area is visible need to be detected/segmented. Correct predictions for objects that are visible from less than 10% are filtered out and not counted as false positives. Up to 100 predictions with the highest scores per image are considered.

4.6.2.2 6D Pose Localization Results

In BOP 2020 the winning method CosyPose [173] reached 69.8 Average Recall (AR) by performing RGB-based deep iterative refinement [119] followed by an exhaustive ICP on depth data that strongly contributed to the 13.3s inference time. Omitting the unoptimized ICP on top of CosyPose reduced the time per image to 0.45s while reducing performance to 63.7 AR. Consequently, the best method taking less than 1 second per image was still a hybrid approach (Koenig-Hybrid [172]) consisting of a DNN for instance segmentation, point pair features (PPFs) [26] for pose estimation and an optimized ICP for pose refinement, reaching 63.9 AR at 0.63s per image.

In BOP 2022, 18 of the 23 new submissions outperformed the previous winner CosyPose [173]. All of these submissions use DNNs in their 6D pose localization pipeline. The winning entry GDRNPP [200] is purely learning-based and achieves 83.7 AR which corresponds to a large gain of $+\Delta$13.9 AR compared to CosyPose [173]. An overview of the 6D pose localization results of all methods is shown in Tab. 4.17. The performance gains are most notable on the industrial ITODD dataset [88] where GDRNPP reaches 67.9 AR ($+\Delta$36.6 AR wrt. CosyPose [173]). This result is significant since ITODD reflects harsh real world conditions given only untextured CAD models and noisy, monochrome test images of metallic parts. This scenario was previously better tackled using PPF-based methods such as KoenigHybrid [172] that yielded 48.3 AR.

GDRNPP: There are a total of seven variants of the GDRNPP method – including the top four entries in BOP 2022 – tailored towards different BOP awards. Specifically, these variants are trained and tested on different data domains and

modalities and are combined with different detection and pose refinement methods. These variants are crucial to analyze the relevance of individual factors in existing complex, multi-stage pipelines.

The common ground is the Geometrically-Guided Direct Regression Network (GDR-Net) [200] which takes in an amodal RGB object crop from which it predicts dense 2D-3D correspondence maps, ambiguity-aware surface regions [166] and object masks. Then, instead of applying PnP+RANSAC [166], all maps are concatenated and fed into a small CNN with a fully connected head that regresses a scale-invariant translation [142] and 3D rotation using the allocentric 6D representation [117]. The loss on 3D rotations takes into account symmetric shapes that are pre-computed for the BOP datasets using the bop toolkit [8]. For the BOP challenge 2022, GDR-Net [200] was adapted to predict both visible and amodal masks as intermediate representations, exchanging the ResNet34 backbone with ConvNext [210] and applying stronger domain randomization.

The winning GDRNPP entry trains YOLOX [195] for object detection and GDR-Net for pose estimation on the provided PBR and real RGB data. On top, a multi-hypothesis pose refinement method inspired by Coupled Iterative Refinement (CIR) [208] is trained on PBR and real RGB-D data.

Training on depth: In BOP 2022, pose estimation and refinement methods [208, 206] started benefiting from learning from depth in addition to RGB and outperform traditional depth-based refinements such as ICP. On the flip side, especially the multi-hypothesis refinements can be time-intensive – the CIR approach [208] takes an average of $\sim 6s$ per image.

Efficiency: As an alternative the third placed entry of GDRNPP deploys a fast and simple depth-based adjustment that only refines 3D translation and still achieves 80.5 AR in just 0.23s per image. In comparison, the best method under one second runtime per image in BOP 2020 was KoenigHybrid [172] with 63.9 AR in 0.63s per image.

RGB only: The efficiency gains are also due to the strongly increased performance of the initial RGB-based pose estimates. Without any pose refinement the purely RGB-based GDRNPP variant reaches 72.8 AR, i.e $+\Delta 9.1$ AR compared

[8] https://github.com/thodan/bop_toolkit

to CosyPose [173] that applies RGB-based pose refinement and still $+\Delta 3.0$ AR compared to the overall best method from BOP 2020, i.e. CosyPose [173] with an exhaustive, depth-based ICP.

Sim2real gap: Another highly relevant result of the BOP 2022 challenge is the GDRNPP variant that was only trained on the provided synthetic PBR data rendered with BlenderProc [133, 163]. With 82.7 AR it achieved the second-highest overall performance. On the datasets with real pose-annotated training data, i.e. T-LESS, YCB-V and TUD-L, the synthetically trained variant is only $-\Delta 2.5$ AR behind the winning method that was *additionally* trained with the provided real training data. In the RGB-only setting, the sim2real gap on these three datasets has reduced from $\Delta 15.8$ AR with CosyPose [173] to $\Delta 6.2$ AR with GDRNPP [200]. The experiments in the BOP 2020 paper [168] have demonstrated the importance of training on the provided PBR data over rasterized images on random backgrounds. The results in BOP 2022 confirm this observation and additionally reveal that the sim2real gap monotonically shrinks with increasing overall performance, see e.g. [173, 215, 206, 200] in Tab. 4.17.

Scalability: The advancements in sim2real transfer are crucial for increasing the scope of applications. In addition, real world applications require methods whose computational and memory resources scale gracefully with the amount of target objects. The top four entries of GDRNPP are all trained with at least one pose network per object. This means inference memory and training time rise linearly with the number of objects. When GDRNPP is trained with one pose network per BOP dataset containing 2–33 objects (Tab. 4.13) it achieves only 74.8 AR and is outperformed by methods such as Extended_FCOS+PFA [206] that reaches 78.7 AR with one pose network per dataset. This also begs the question of how results would change if the PFA [206] method had been trained per object.

Detector agnostic results: Almost all submissions use a 2D detection or instance segmentation method to distinguish objects and narrow the search space for a subsequent pose estimation stage. The amodal 2D bounding box estimates are also frequently used for estimating 3D translation [200, 152]. This poses the question of where the performance gains of GDRNPP are actually coming from, better pose estimation or simply better detection? To answer this question we provided participants of the BOP 2022 challenge with a set of default MaskRCNN

#	Method	...based on	Year	Data	...type	AP	Time (s)
1	GDRNPPDet	YOLOX [195]	2022	RGB	PBR+real	77.3	0.081
2	GDRNPPDet	YOLOX [195]	2022	RGB	PBR only	73.8	0.081
3	Extended_FCOS	FCOS [153]	2022	RGB	PBR+real	72.1	0.030
4	Extended_FCOS	FCOS [153]	2022	RGB	PBR only	66.7	0.030
5	DLZDet	DLZDet	2022	RGB	PBR only	65.6	-
6	CosyPose-ECCV2020	MaskRCNN[90]	2020	RGB	PBR+real	60.5	0.054
7	CosyPose-ECCV2020	MaskRCNN[90]	2020	RGB	PBR only	55.7	0.055
8	FCOS-CDPN	FCOS [153]	2022	RGB	PBR only	50.7	0.047

Table 4.18: 2D object detection: Average Precision (AP)

detections and instance segmentations from the previous winning method CosyPose [173] and offered a BOP award for the best pose estimation results on top of them. Among the ten submissions that used the default detections GDRNPP again came out on top with 79.8 AR. Therefore, we can conclude that the pure pose estimation performance of the GDRNPP pipeline is performing the best independent of the used detection or instance segmentation method. However, the performance gap to other methods such as PFA [206] and SurfEmb [205] that were trained per dataset and not per object shrinks in this setting.

4.6.2.3 2D Object Detection Results

Object detection and segmentation performance do have a strong influence on pose estimation results. For example, the YOLOX detector [195] employed by the GDRNPP pipeline increased pose estimation performance by + Δ3.9 AR compared to when using the default detections from the last competition winners CosyPose. To further disentangle the gains of the detection and pose estimation stage, we began measuring object detection and segmentation performance following the COCO metrics in BOP 2022 as described in Sec. 4.6.2.1. As shown in Tab. 4.18 the GDRNPP pipeline reached the best detection performance of 77.3 AP by training a YOLOX [195] detector with ConvNext [210] backbone, strong data augmentation and the ranger optimizer[156] on each dataset. In comparison the MaskRCNN from the previous winners CosyPose resulted in a detection score of 60.5AP ($-\Delta$16.8 AP) which explains the Δ3.9 AR difference in pose estimation performance. With only PBR data the GDRNPP entry achieves 73.8 AP ($-\Delta$3.5 AP) which shows that the sim2real gap in object detection is also narrowing. For all methods the sim2real gap is highest in TUD-L, moderate in YCB-V and lowest in T-LESS. Potential reasons are the lower model texture quality in TUD-L, noisy

#	Method	...based on	Year	Data	...type	AP	Time (s)
1	ZebraPoseSAT	CosyPoseDet+ZebraPose[215]	2022	RGB	PBR+real	58.7	0.080
2	ZebraPoseSAT	CDPNv2Det+ZebraPose[215]	2022	RGB	PBR+real	57.8	0.080
3	ZebraPoseSAT	CosyPoseDet+ZebraPose[215]	2022	RGB	PBR only	53.8	0.080
4	ZebraPoseSAT	CDPNv2Det+ZebraPose[215]	2022	RGB	PBR only	52.3	0.080
5	DLZDet-PBRREAL	DLZDet	2022	RGB	PBR+real	49.6	-
6	DLZDet-PBR+Real	DLZDet	2022	RGB	PBR+real	43.3	-
7	DLZDet-PBR1	DLZDet	2022	RGB	PBR only	42.9	-
8	CosyPose-ECCV20	MaskRCNN[90]	2020	RGB	PBR+real	40.5	0.054
9	CosyPose-ECCV20	MaskRCNN[90]	2020	RGB	PBR only	36.2	0.055

Table 4.19: 2D object segmentation: Average Precision (AP). Note that CosyPoseDet corresponds to the provided default detections and DLZDet is not associated to any publication.

test images in both TUD-L and YCB-V or the specific scene structure and lighting in TUD-L, but these effects need to be further de-correlated. While we still expect and encourage further improvements, the best detection results will again be made public to facilitate focusing on pose estimation approaches and obtaining better comparisons among them.

4.6.2.4 2D Object Segmentation Results

Tab. 4.19 shows the results of the submitted object segmentation methods. We see an improvement from 40.5 AP of the default MaskRCNN segmentations from CosyPose to 58.7 AP (+Δ18.2 AP) of the ZebraPoseSAT method. Interestingly, ZebraPoseSAT simply predicts better masks on top of the detections from CosyPose using the ZebraPose [215] network. This implies that the full potential of object segmentation methods is not yet reflected as ZebraPose on top of GDRNPP's YOLOX detections would likely yield better results. It is also apparent that in the BOP challenge most pose estimation pipelines including CosyPose [173], Extended_FCOS+PFA [206], GDRNPP [200] and ZebraPose [215] rely on 2D detections from which they extract a rectangular scene crop in their first stage. Only in subsequent stages they distinguish the object from background and foreground pixels. An exception is the RCVPose3D [219] method which segments the point cloud in the first stage followed by pose estimation using a 3D point cloud network based on PointNet++. Performing object segmentation in the first stage has the advantage of avoiding overlapping bounding boxes corresponding to intersecting instances as present in ITODD [88] (Fig. 4.15) and iShape [201].

4.6.2.5 Conclusions

The BOP challenge 2022 has seen major breakthroughs in performance, efficiency and sim2real transfer of 6D pose estimation pipelines. These advancements are driven by neural network architectures and training schemes that manage to generalize effectively from the provided PBR training data. The accuracy and speed of these networks now clearly surpass the traditional PPF-based methods which lays the basis for new applications in robotics and Augmented Reality. Variations of the winning method GDRNPP [200, 209] allowed us to analyze the importance of different factors related to training data, domain and efficiency. On top, we separately measured detection and segmentation performance and could thereby determine where the gains in multi-stage pose estimation pipelines originate from. Despite the progress, pose and detection scores have not been saturated on most BOP datasets. Furthermore, new challenges such as transparent objects, multi-view or model free few-shot pose estimation are on the horizon and the pose refinement stage also deserves a closer inspection. Therefore, the evaluation system stays open, and we are looking forward to the improvements on current and future tasks in the next BOP challenge.

Approach	AAEs [181]	ICG pose refinement		ICG tracking (GT)
Data	RGB	Depth	Region+Depth	Region+Depth
ADD-S	72.8	76.9	80.3	94.7
ADD	50.5	57.5	61.2	86.4

Table 4.20: Results using ICG as pose refinement on AAE initialization on every frame of YCB-V. As a comparison we show ICG used as

4.7 Combining 6-DoF Object Pose Estimation and Tracking (CVPR 2022)

Single frame 6D object pose estimation neglects temporal information that is often available in the form of sensor streams in robotic applications. Temporal information allows exploiting object permanency, i.e. the low probability of drastic changes in object pose between subsequent frames. This is especially useful in dynamic scenes if the object or the camera are in motion. By constraining the potential 6D solution space to neighboring poses the pose retrieval problem is strongly simplified. Consequently, low-level features such as point cloud distances and color histograms are often sufficiently expressive for the 6D tracking task. At the same time, those methods are extremely efficient which is necessary to process camera streams at 30 frames per second. At CVPR 2022, we proposed Iterative Corresponding Geometry (ICG) [214], a highly efficient probabilistic tracker that fuses low-level region and depth features and reaches state-of-the-art results on benchmarks such as YCB-V.

The downside of such 6D pose trackers is that they need to be initialized and re-initialized once they lose track of the object pose, occlusion or leaving the field of view. Therefore, in the context of a master thesis [212] we also investigated the challenge of detecting tracking loss and the necessary re-initialization through 6D pose detectors which is an important practical application.

Trackers such as ICG can also be used for 6D pose refinement and we performed experiments where ICG is initialized with AAE [181] pose estimates on the YCB-Video sequences. The results in Tab. 4.20 show that ICG refinement improves upon AAE pose estimates and region and depth information are shown to be complementary. Similarly, in [224] we demonstrate the effectiveness of using ICG as a multi-hypothesis pose refinement in the context of satellite pose estimation.

Figure 4.19: Multi-body objects included in the *RTB* dataset [222].

4.7.1 Robot Tracking Benchmark (RTB)

Previously, we have considered pose estimation and tracking of rigid objects. However, in robotic applications we often face articulated structures such as robotic arms and grippers where large parts of the structure can be occluded or reside outside the image. Here, it is crucial to exploit not only temporal constraints but also lateral and rotational kinematic constraints. In [222] we propose such a framework for multi-body object tracking and a corresponding benchmark based on BlenderProc [133].

In the past, evaluations of multi-body tracking and pose estimation considered only a very limited number of sequences and experiments were typically of a more qualitative nature [33, 42, 44, 61, 62, 65]. The main reason is that real-world data from diverse camera angles with high-quality pose annotations is often hard to obtain even if object configurations can be measured. However, for a reliable evaluation, a sufficient number of realistic sequences with accurate ground truth is essential. We, therefore, introduce the *Robot Tracking Benchmark (RTB)*. It is a novel synthetic dataset, which we developed using the procedural rendering pipeline BlenderProc. Example images of the dataset are shown in Fig. 4.19.

To enable the creation of the dataset, we extended *BlenderProc* to load robot models from *URDF* files. For the generation of articulated motions, forward and

backward kinematics were integrated. Our open-source pipeline produces photo-realistic sequences with *HDRi* lighting[9] and physically-based materials. Perfect ground truth annotations for camera and robot trajectories are provided in the *BOP* format [168]. Many physical effects such as motion blur, rolling shutter, and camera shaking are accurately modeled to reflect real-world conditions. For each frame, we also generate four depth qualities to simulate sensors with different characteristics. While the first quality provides perfect ground truth, the second considers measurements with the distance dependent noise characteristics of the *Azure Kinect* time-of-flight sensor [199]. We also model smoothed depth using random Gaussian shifting, as well as missing measurements at very dark surfaces. Finally, for the third and fourth quality, two stereo RGB images with and without a pattern from a simulated dot projector are rendered. Depth images are then reconstructed using *SGM*[18]. The dataset can be found here[10].

The benchmark features six robotic systems with different kinematics, ranging from simple chain and tree topologies to structures with complex circular dependencies. Example images of the included multi-body objects are shown in Fig. 4.19.

For each robotic system, we provide three difficulty levels *easy*, *medium*, and *hard*. In all sequences, the kinematic system is in motion. However, while for *easy* sequences the camera is mostly static with respect to the robot, *medium* and *hard* sequences feature faster and shakier motions for both the robot and camera. Consequently, motion blur increases, which also reduces the quality of stereo matching. Finally, for each object, difficulty level, and depth image quality, 10 sequences with 150 frames are rendered. In total, this results in 108.000 frames that feature different kinematic structures, motion patterns, depth measurements, scenes, and lighting conditions. In summary, our *RTB* allows to extensively measure, compare, and ablate the performance of multi-body tracking algorithms, which is a basis for further progress in the field.

[9] https://polyhaven.com/hdris
[10] https://zenodo.org/record/7548537

5 Contributions: Unknown Object Manipulation

In the previous Chapter 4 we have covered the task of 6-DoF object pose estimation where the object instance or category is known in advance. Access to 3D models is a valid assumption in the manufacturing context and even in other cases meshes can be often acquired using 3D reconstruction pipelines [83, 204].

However, autonomous robots in the wild also need to be able to manipulate completely unknown objects with no geometrically consistent categories and potentially non-rigid parts. While the sim2real transfer techniques presented in Section 4.2.3.6 allow scaling 6-DoF pose estimation methods to infer fine-grained 3D world models at a factory level, it is hardly possible to generate 3D meshes for every object on earth and train corresponding pose estimation networks.

In contrast, humans are able to locate and pick up objects from a single view and place them elsewhere without precise knowledge about their 3D geometry. They are able to infer the necessary information from visual data, tactile feedback and their prior experience with similar shapes in the physical world.

For robots, grasping arbitrary objects without access to their 3D meshes given a single image from an arbitrary 6-DoF viewpoint is a very challenging task, as discussed in Section 1.2. Therefore, most existing methods are either limited to top-down 2D grasps [97] or can only deal with singulated objects [147].

In the following Section 5.1, the results of my publication on Contact-GraspNet [198] are presented where we learn to predict distributions of 6-DoF parallel-jaw grasps in cluttered scenes of unknown objects from single depth images.

5.1 Contact-GraspNet: Efficient 6-DoF Grasp Generation in Cluttered Scenes (ICRA 2021)

Martin Sundermeyer[1,2,3], Arsalan Mousavian[1], Rudolph Triebel[2,3], Dieter Fox[1,4]

[1]NVIDIA, [2]German Aerospace Center (DLR), [3]Technical University of Munich (TUM), [4]University of Washington (UW)

5.1.1 Motivation

The ability to grasp objects is one of the fundamental capabilities required in most robot manipulation tasks. Grasping involves reasoning about the 3D geometry and physics properties of the object such as mass and friction, and also reasoning about complex contact physics. It is studied in two main directions: Model-based grasping where the 3D model or category of the object is known and model-free grasping where there is no prior knowledge about the object. Model-based grasping circumvents reasoning about the physics of contact and grasp generation by pre-defining a set of grasps in the object frame and transform those grasps according to the 6-DoF object pose [132, 181, 162, 154] or detected keypoints of the objects [146, 112]. The downside of model-based approaches is that they only work on a limited subset of known objects or categories, and any errors in detecting 6-DoF object pose or object keypoints degrade the grasping performance.

Model-free approaches do not make any strong assumptions about the category or shape of the object, and they learn a shared representation for all object shapes and sizes. However, having one shared representation for all objects in addition to the large SE(3) space for the grasp poses makes the learning problem quite challenging. As a result, a large body of work in data-driven grasping constraints the space of possible grasps to planar grasping, where grasps are represented by oriented rectangles around each pixel that define the grasp frame [97, 144, 59]. Such a representation needs the camera to view the scene perpendicularly and thus limits 3D reasoning and applications significantly. A large number of possible grasps and the full kinematic capabilities of the robot are also neglected. To address the limitations of planar grasping, there has been a recent interest in tackling the problem of 6-DoF grasping of unknown objects [100, 147, 176, 179, 165]. In this paper, we tackle 6-DoF grasping of unknown objects in cluttered

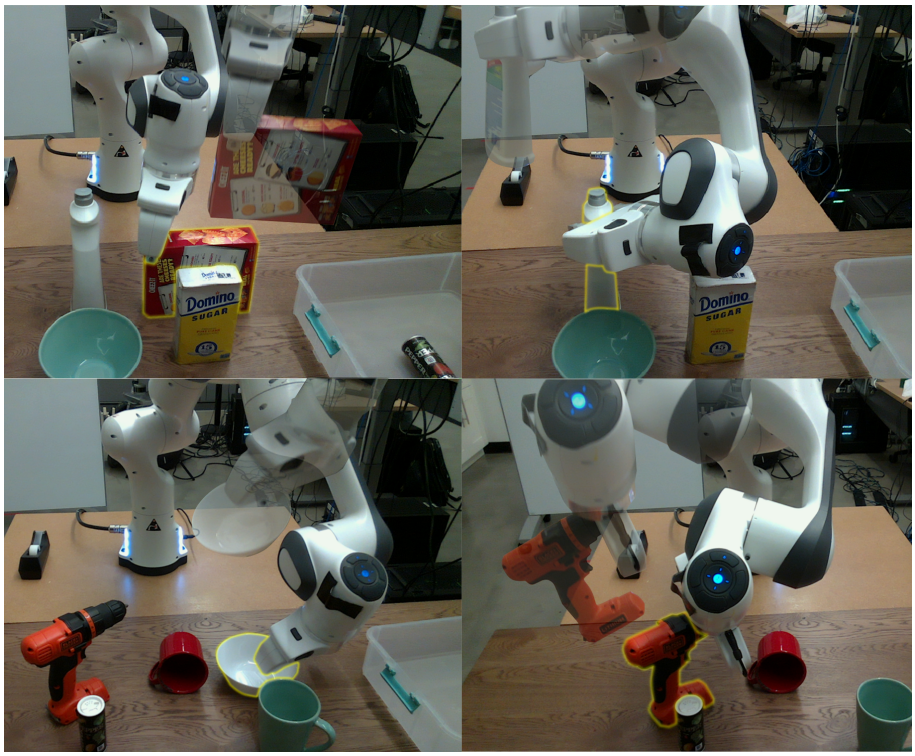


Figure 5.1: Contact-GraspNet efficiently predicts diverse and stable grasps in cluttered scenes while avoiding collisions.

space from a partial point cloud observation of the scene.

Grasping objects from cluttered scenes with structure introduces extra challenges. The target objects must be grasped successfully, while at the same time any collision with other objects must be avoided to prevent damages or transformations into other undesired states. This is particularly important in home robotics and healthcare applications. Additionally, it is crucial to generate a diverse set of grasps for the object due to robot kinematic constraints. Depending on the relative pose between the object and the robot, a different subset of grasps is kinematically feasible.

Our method is closely related to the work of Murali *et al.* [176], where the goal is to generate collision-free diverse grasps for a designated target object from a partial point cloud of the scene, and the objects are segmented using a pre-trained unknown object instance segmentation model [183, 157]. Murali *et al.* [176] use a multi-stage process that synthesizes grasps for the target objects from the segmented object point cloud with no context around it, and then filters out the colliding grasps using another learned model. This leads to three issues: 1) Sensitivity to instance segmentation errors. 2) Grasps are generated just from

the target object point cloud and do not leverage geometric cues in the scene such as table points and surrounding object points. 3) Grasps are predicted in the large, unconstrained 6-DoF pose space. To address these issues, our method instead directly processes a full scene point cloud or a local region around a target object. Therefore, the quality of our generated grasps is not depending on an accurate mask and collisions can be directly taken into account during generation. Instance segmentation can then subsequently be used to filter grasps belonging to a target object. Thus, our main contributions are the following:

- A new end-to-end method for 6-DoF grasping of unknown objects in cluttered real world scenes where we achieve 90% grasp success rate. This is 10% higher than [176] in equal settings.
- A new grasp pose representation that projects 6-DoF grasps to their contact points in an observed point cloud. Our representation has only 4-DoF which facilitates the learning problem significantly.
- Comprehensive ablation studies in a physics simulator to evaluate the effects of different loss functions and training data.

Contributions: I proposed and implemented the approach during a research internship at the NVIDIA AI Robotics Research Lab. Prof. Dieter Fox supervised me and discussed the main ideas in weekly meetings and gave valuable feedback on the paper and video. Arsalan Mousavian helped significantly with his prior experience and works in object grasp estimation, the ACRONYM dataset, paper writing and implementation on the Franka Panda Robot. Prof. Rudolph Triebel made this collaboration possible and contributed to the paper.

5.1.2 Abstract

Grasping unseen objects in unconstrained, cluttered environments is an essential skill for autonomous robotic manipulation. Despite recent progress in full 6-DoF grasp learning, existing approaches often consist of complex sequential pipelines that possess several potential failure points and run-times unsuitable for closed-loop grasping. Therefore, we propose an end-to-end network that efficiently generates a distribution of 6-DoF parallel-jaw grasps directly from a depth recording of a scene. Our novel grasp representation treats 3D points of the recorded point cloud as potential grasp contacts. By rooting the full

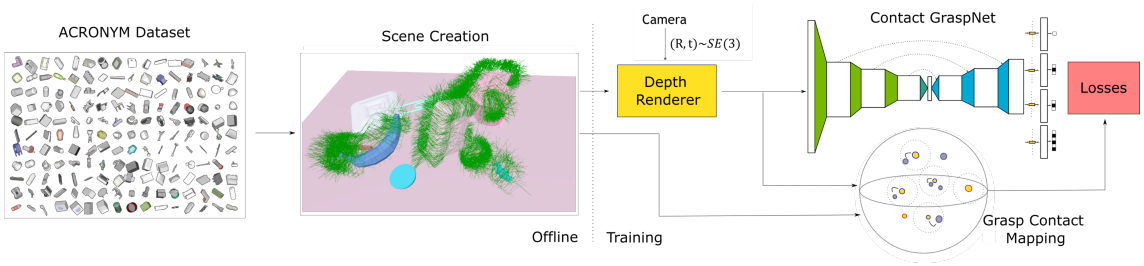


Figure 5.2: Training Data Pipeline. We place object meshes with dense grasp annotations from the ACRONYM dataset [194] at random stable poses in scenes. Grasp poses that produce gripper model collisions are removed. Resulting grasps are mapped to their contacts on the mesh surface. During training, we sample virtual cameras to render point clouds from the scenes. We consider recorded points (yellow) as positive contacts if there exists a mesh contact (blue) in a 5mm radius and associate the grasp transformation belonging to the closest mesh contact to them. These per-point annotations are used to supervise the Contact Grasp Network.

6-DoF grasp pose and width in the observed point cloud, we can reduce the dimensionality of our grasp representation to 4-DoF which greatly facilitates the learning process. Our class-agnostic approach is trained on 17 million simulated grasps and generalizes well to real world sensor data. In a robotic grasping study of unseen objects in structured clutter we achieve over 90% success rate, cutting the failure rate in half compared to a recent state-of-the-art method. Video of the real world experiments and code are available here [1].

5.1.3 Method

We consider the problem of generating 6-DoF grasps from any viewpoint on structured clutter consisting of unknown objects. Our approach takes in a raw depth image, optionally with object masks, and generates 6-DoF grasp proposals together with corresponding grasp widths. Our goal is to predict grasps that are robust, diverse and non-colliding from an only partially observable scene.

From a learning perspective, generating the distribution of successful 6-DoF grasps is quite challenging, because the distribution is multi-modal, discontinuous, imbalanced and ambiguous due to (self-) occlusions. Furthermore, direct regression in high dimensional output spaces like $SE(3)$ has been shown to be difficult in grasping [147] and also in related fields such as object pose estimation [124].

[1] https://research.nvidia.com/publication/2021-03_Contact-GraspNet%3A--Efficient

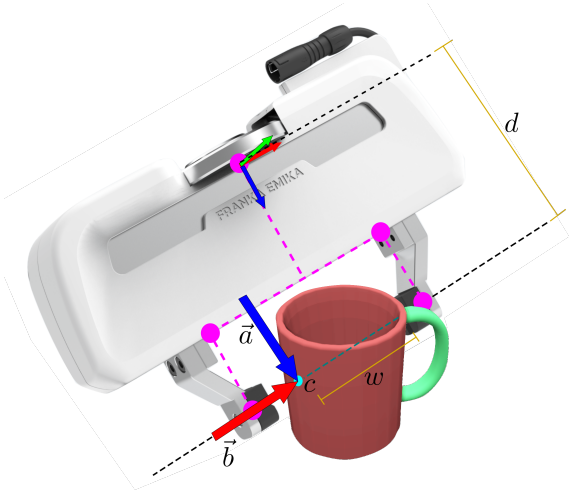


Figure 5.3: Our grasp representation: c depicts an observed contact point. **a** and **b** constitute the 3-DoF rotation, w is the predicted grasp width, d the distance from baseline to base frame. In pink we show the five gripper points **v** that we used in the l_{add-s} loss.

5.1.3.1 Grasp Representation

For these reasons, finding an efficient grasp representation is crucial to solve this task using learning-based methods. This representation should generalize well to unseen objects and handle the high-dimensional output space well.

Contact Grasp Representation: We observe that for most predictable two-finger grasps at least one of the two contacts is visible prior to grasping. In contrast, grasps without any visible contact are often ambiguous or do not preserve the initial object pose after grasping. Therefore, we map a distribution of successful 6-DoF ground truth grasps $g \in G$ to their corresponding contact points $c \in \mathbb{R}^3$. Since visible contact points are bound to lie on surfaces that we can observe with a depth sensor, we can represent their 3D location by nearby points in a recorded point cloud.

Given that we can predict whether observed points are suitable grasp contacts, we can thus reduce the 6-DoF grasp learning problem to estimating the 3-DoF grasp rotation $R_g \in \mathbb{R}^{3\times3}$ and grasp width $w \in \mathbb{R}$ of a parallel-yaw gripper.

Starting from a contact point $\mathbf{c} \in \mathbb{R}^3$, where the gripper baseline intersects the mesh, we depict a 6-DoF grasp pose $g \in G$ defined by $(R_g, t_g) \in SE(3)$ and grasp width $w \in \mathbb{R}$ as

$$\mathbf{t}_g = \mathbf{c} + \frac{w}{2}\mathbf{b} + d\mathbf{a} \tag{5.1}$$

$$R_g = \begin{bmatrix} | & | & | \\ \mathbf{b} & \mathbf{a} \times \mathbf{b} & \mathbf{a} \\ | & | & | \end{bmatrix}, \tag{5.2}$$

where $\mathbf{a} \in \mathbb{R}^3, ||\mathbf{a}|| = 1$ is the approach vector, $\mathbf{b} \in \mathbb{R}^3, ||\mathbf{b}|| = 1$ is the grasp baseline vector, and $d \in \mathbb{R}$ is the constant distance from the gripper baseline to the gripper base. Our grasp representation is depicted in Figure 5.3.

The reduced dimensionality greatly facilitates the learning process compared to methods that estimate grasp poses in unconstrained $SE(3)$ space. It also increases the pose accuracy of predicted grasps as they are bound to the geometry of the observed scene. In contrast to axis-angle representations, our rotation representation has neither ambiguities nor discontinuities. Moreover, at test time we can sample grasp proposals by sampling contact points that cover the whole observable surface of the scene/object and thus represent the modes of the 6-DoF grasp distribution well. While a 3D view on the scene is preferable, even a frontal view on a box produces reasonable grasps due to the radial mapping.

Point Set Networks such as PointNet++ [101] effectively process point clouds and hierarchically aggregate points and their feature representations in local 3D neighborhoods. Their predictions can be directly associated to 3D points in the input point cloud and our proposed grasp representation exploits this ability.

5.1.3.2 Data Generation

To learn the full distribution of stable 6-DoF grasps, diverse and dense grasp pose annotations are required. We used the ACRONYM dataset [194], which consists of 8872 meshes from the Shapenet dataset [56] and 17.7 million simulated grasps under varying friction. An overview of our offline and online training data generation is given in Fig. 5.2.

During training, we render a scene point cloud $\mathcal{P} = \{\mathbf{p}_1, \ldots, \mathbf{p}_n\} \subset \mathbb{R}^3$ and assign a point-wise grasp success

$$\forall i = 1, \ldots, n \quad s_i = \begin{cases} 1 & \min_j ||\mathbf{p}_i - \mathbf{c}_j||_2 < r \\ 0 & otherwise, \end{cases} \tag{5.3}$$

where $\mathbf{c}_j \in \mathcal{P}$ are the mesh contact points of non-colliding ground truth grasps $g_j \in G$ in camera coordinates and $r \in \mathbb{R}$ is their maximum propagation radius. Thus, $\mathcal{P}$ can be split into points $\mathcal{P}^- := \{\mathbf{p}_i | s_i = 0\}$, where no feasible grasp

contact is found within a radius of $r = 5mm$, and $\mathcal{P}^+ := \{\mathbf{p}_i | s_i = 1\}$, containing points suitable for a contact. To the latter ones $\mathbf{p}_i^+ \in \mathcal{P}^+$ we assign the closest grasp as

$$\begin{bmatrix} w_{g,i} \\ R_{g,i} \\ \mathbf{t}_{g,i} \end{bmatrix} = \begin{bmatrix} w_{g,j} \\ R_{g,j} \\ \mathbf{p}_i^+ + \frac{w_j}{2}\mathbf{b}_j + d\mathbf{a}_j \end{bmatrix} \tag{5.4}$$

with

$$j = \arg\min_k || \mathbf{p}_i^+ - \mathbf{c}_k ||_2 \tag{5.5}$$

Given sufficient coverage we can thereby project the ground truth distribution of 6-DoF grasps densely on the recorded point cloud.

5.1.3.3 Network

We employ the set abstraction and feature propagation layers proposed in PointNet++ [101] to build an asymmetric U-shaped network. The network takes n=20000 random points $p \in \mathbb{R}^{20000\times 3}$ as input and predicts grasps for only m=2048 farthest points of the input to make sure the inference fits in GPU memory and predicted grasps have good coverage over the scene. The network has four heads with two 1D-Conv layers each and per-point outputs $s \in \mathbb{R}, \mathbf{z}_1 \in \mathbb{R}^3, \mathbf{z}_2 \in \mathbb{R}^3, \mathbf{o} \in \mathbb{R}^{10}$, from which we form our grasp representation. The predicted grasp width $\hat{\mathbf{w}}_i \in [0, w_{max}]$ is split into 10 equidistant grasp width bins $\hat{\mathbf{o}} \in \mathbb{R}^{10}$ to counteract data imbalance. Then, $\hat{\mathbf{w}}_i$ is represented by the center value of the bin(s) with the highest confidence. The approach direction $\mathbf{a} \in \mathbb{R}^3$ and the baseline direction $\mathbf{b} \in \mathbb{R}^3$ are orthonormal by definition. We inject this property into training by coupling the predictions $\hat{\mathbf{a}}, \hat{\mathbf{b}}$ through an in-network Gram Schmidt orthonormalization

$$\hat{\mathbf{b}} = \frac{\mathbf{z}_1}{||\mathbf{z}_1||} \qquad \hat{\mathbf{a}} = \frac{\mathbf{z}_2 - \langle \hat{\mathbf{b}}, \mathbf{z}_2 \rangle \hat{\mathbf{b}}}{||\mathbf{z}_2||} \tag{5.6}$$

Thus, we perform a projection and only predict $\hat{\mathbf{a}}$ as the component that is orthonormal to $\hat{\mathbf{b}}$. The orthonormalization further reduces the dimensionality of our predicted grasp representation and facilitates the regression of 3D rotations [160].

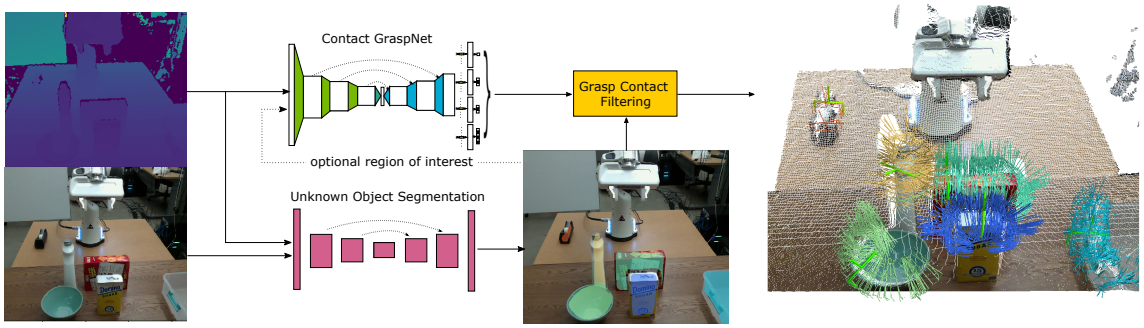


Figure 5.4: Full Inference Pipeline: We segment unknown objects from an RGB-D image using [183]. Our Contact-GraspNet processes the full scene point cloud or a local region of interest around a target object. Predicted 6-DoF grasps are then associated to object segments by filtering their contact points. On the right we show the predicted 6-DoF grasp distribution and, in bold, the most confident grasp per segment.

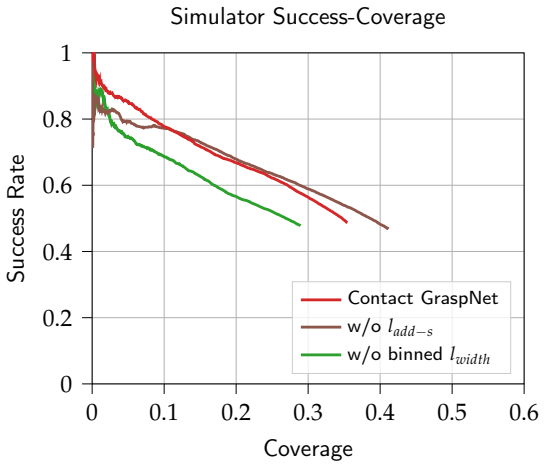


Figure 5.5: Loss Ablations: Without weighted binning in the grasp width loss l_{width} both, success rate and coverage decrease. The l_{add-s} loss leads to increased success rates at high confidence contacts (Coverage $\in [0, 0.1]$) and to slightly decreased success rate in the low-confidence regime. This confidence calibration is important, since it determines which grasp is eventually executed.

5.1.3.4 Target Losses

The contact grasp success predictions $\hat{s} \in \mathbb{R}$ are evaluated at all output points $\mathbf{p}_i \in \mathbb{R}^3 : \forall i \in [0, m]$ using binary cross entropy. We only backpropagate the top-k point predictions with the largest errors $l_{bce,k}$, with k=512, to counteract data imbalance. The other predictions concerning the geometry of grasps are only evaluated at positive contact points $\mathbf{p}_i^+$. Instead of supervising all network

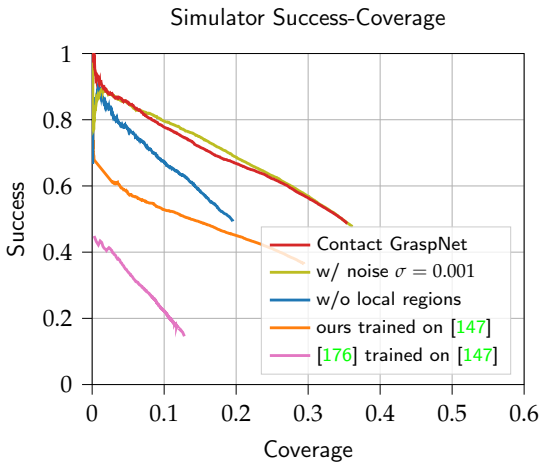


Figure 5.6: Data Ablations: Training with Gaussian noise has similar performance in simulation but helps generalization to noisy sensor data. Predicting grasps directly on full scenes without extracting local regions yields a similar average success rate, but significantly lowers grasp coverage. Training on the small grasp dataset from [147] with 5 categories is not sufficient to generalize to arbitrary objects and shows the importance of ACRONYM [194]

heads in isolation, we propose to combine the predictions to the 6-DoF grasp pose $\hat{g} \in G$ given in Eq. (5.1) and (5.2) already during training. We define five 3D points $\mathbf{v} \in \mathbb{R}^{5\times3}$ representing the 6-DoF gripper pose, as shown in Fig. 5.3, and transform these using all ground truth and predicted grasp poses defined in Eq. (5.4)

$$\mathbf{v}_i^{gt} = \mathbf{v}R_{g,i}^T + \mathbf{t}_{g,i} \qquad \mathbf{v}_i^{pred} = \mathbf{v}\hat{R}_{g,i}^T + \hat{\mathbf{t}}_{g,i} \tag{5.7}$$

We formulate the 6-DoF grasp loss l_{add-s} as a weighted minimum average distance between gripper points $\mathbf{v}^{gt}$ and $\mathbf{v}^{pred}$ where we take the symmetry of the gripper into account.

$$l_{add-s} = \frac{1}{n^+} \sum_i^{n^+} \hat{s}_i \min_u ||\mathbf{v}_i^{pred} - \mathbf{v}_u^{gt}||_2, \tag{5.8}$$

where n^+ is the size of $\mathcal{P}^+$. We weight each distance to the closest ground truth grasp points with the predicted contact success confidence $\hat{s}_i$.

Our proposed loss formulation has several advantages: (1) We can learn the different modes of the ground truth grasp distribution, e.g. different predicted

grasp approach directions $\hat{\mathbf{a}}$ can produce a small error. (2) The point-wise weighting with $\hat{s}_i$ couples the contact point classification with the grasp pose predictions. Contact confidence can only increase if the network predicts a 6-DoF grasp pose close to a ground truth pose. (3) Wrongly predicted grasps in regions far away from any ground truth grasp, e.g. at artificial edges from occlusions, produce a high loss and are thus avoided.

On the grasp width bin predictions, we optimize a weighted, multi-label binary cross entropy loss l_{width}. Since small grasp widths are highly over-represented, we weight the bin losses anti-proportional to bin size. Our total loss is $l = \alpha l_{bce,k} + \beta l_{add-s} + \gamma l_{width}$ with $\alpha = 1, \beta = 10, \gamma = 1$.

5.1.3.5 Implementation Details

We use the Adam optimizer with an initial learning rate of 0.001 and a step-wise decay to 0.0001. Our set abstraction layers have 3 parallel branches with query ball radii [0.02,0.04,0.08], [0.04,0.08.0.16] and [0.08,0.16,0.32]. For inference the point cloud is centered at its mean in camera coordinates. For training, we generate 10000 table top scenes by placing 8-12 grasp annotated ShapeNet models [194] at random stable poses. We use rejection sampling to avoid collisions. We train with a batch size of 3 for 144.000 iterations which takes $\sim$ 40 hours on a single Nvidia V100 GPU. Convergence is significantly faster than on previous methods [176, 165, 147] which take up to one week on a single GPU for training. This also reflects the effectiveness of our proposed grasp representation.

5.1.4 Experimental Evaluation

We evaluate our method in a grasping study with a Franka robot where we pick unknown objects in cluttered scenes. We also compare different variations of our method and of our data by executing a large number of predicted grasps in the FleX physics simulator [52].

5.1.4.1 Inference

Our inference pipeline is shown and described in Fig. 5.4. The Contact-GraspNet can also be applied to raw depth images by itself, but most robotic tasks require some kind of instance detection/segmentation to specify a target.

Figure 5.7: One advantage of our method is that it does not rely on an accurate segmentation of unknown objects. Here, successful grasp contacts are still found on the driller despite severe under-segmentation.

Local regions of interest can be optionally extracted around the 3D centroid of point cloud segments in order to maximize the number of potential contact points. In our experiments, we extract cubes with an edge length set to twice the largest spanning dimension, but at least $0.3m$ and at most $0.6m$.

Run time: The Contact-GraspNet has a run time of $0.28s$ for a full scene or $\sim 0.19s$ for a local region around a target object. Compared to other 6-DoF grasp generation methods this is quite fast and enables applications requiring reactive closed loop grasping.

Grasp Selection: At test time we select grasps by setting a contact confidence threshold of 0.23 and then use farthest point sampling on the (filtered) contact points to ensure broad grasp coverage. If the number of predicted grasps for an object is too low, we reduce the confidence threshold to 0.19. In the end we execute the most confident grasp that is kinematically reachable and where the robot does not collide with the scene [191].

5.1.4.2 Evaluation Metrics

In our robotic experiments we report the number of successful grasps and the number of trials. The latter is often disregarded when picking small objects from a bin. However, grasping in only one or two trials is crucial in cluttered scenes (e.g. in households) with large, densely packed objects where collisions should be avoided and stable grasp opportunities can vanish after objects tip over. We limit ourselves to a maximum of two grasp trials per object without rearrangements and report the success rate after a single trial as well.

Our simulator experiments allow us to also evaluate the diversity of grasps and ablate variations of our method. Here, we evaluate the success rate and coverage of the generated grasps following [147]. A grasp is considered successful if (1) the open gripper does not collide with the object/scene and (2) the object is still in the gripper after grasping and a shaking motion. This is a conservative measure, as most real world grasps can slightly collide and do not undergo a shaking motion. Coverage is the percentage of ground truth grasps (including occluded ones) whose base coordinates are within 2cm of any of the generated grasps.

5.1.4.3 Real robot grasp experiments

Setup: Our physical setup consists of a 7-DoF Franka Panda robot with a parallel-jaw gripper. We closely replicate the 9 cluttered scenes defined in [176] with a total of 51 unseen objects. The task is to pick the objects from the cluttered scene and place them into a bin. We manually select target objects and grasp them in the same random order as in [176]. In our experiments, we use the Intel Realsense L515 LiDAR camera mounted on a tripod for both RGB and depth data. Robot motions are generated using [191].

Results: Table 5.1 shows our grasp evaluation results on the robot. We observe a significantly higher grasp success rate of our method compared to [147] and [176] which themselves outperform other learning-based methods and analytic/heuristic baselines. Furthermore, our method strongly improves the grasp success at first trial and thereby reduces the number of re-grasps. We also addressed the shortcomings of cropping objects from the point cloud using potentially imprecise segmentation masks. Fig. 5.7 shows an imprecise segmentation example where cropping would be catastrophic but where our grasp filtering method can still extract successful grasps.

5.1.4.4 Ablations

Optimization Targets: In Fig. 5.5 we first investigate the effect of our loss targets. The weighted loss on the grasp width bins l_{width} is crucial to deal with the imbalanced widths in our grasp dataset. Without weighting the bins, the predictions mostly collapse into narrow grasp widths. Weighting also performs better than oversampling in our experiments. The average distance loss l_{add-s} improves the success rate of high confidence contacts which is important because

Table 5.1: Cluttered Scene Grasping: We achieve a clear improvement over recent state-of-the-art grasping pipelines

	Success	First attempt	#Attempts
6-DOF GraspNet[147]	62.7	-	-
[147] +CollisionNet[176]	80.39	68.63	67
Contact-GraspNet	**90.20**	**84.31**	**59**

most grasps that we execute lie in the first decimal of coverage. The connection of contact confidence with the grasp pose results in an overall improved calibration.

Data: In Fig. 5.6 we examine the effects of different training and test data. Zooming into local regions allows the network to concentrate potential contact points on the object and thus increases coverage. We also show the importance of a large and diverse grasp dataset like ACRONYM [194]. Training on a small grasp datasets with 110 objects from 5 categories [147] is not sufficient for out-of-category generalization irrespective of the method.

Failure Cases: We observe some failure cases for thick objects that only allow grasps almost at maximum grasp width. Here, grasp predictions are less confident presumably because of the discontinuous decision boundary. Injecting noise during training reduces this effect. Finally, small objects sometimes have contact points with low confidence possibly because of their small impact on the total loss.

5.1.5 Conclusions

We considered the fundamental problem of grasping unknown objects in structured clutter with a parallel jaw gripper. We proposed an efficient, accurate and simplifying 6-DoF grasp generation method called Contact-GraspNet. By transforming the hardly tractable 6-DoF grasp estimation problem into a grasp contact point classification and a grasp rotation estimate, we greatly limit the predicted pose space and facilitate the learning process. Through tailored optimization targets that take into account the multi-modality, imbalance and sparsity of the 6-DoF grasp distribution, our network learns to generate diverse grasps covering the whole graspable surface in a recorded scene. Gripper collisions are effectively avoided by considering them during training and by predicting grasps directly in

scenes. Our approach can incorporate segmentation predictions as well but is not dependent on accurate masks itself. It is also complementary to grasp ranking methods that use gripper and/or robot models as input. Grasping successfully with a single attempt is crucial in sensible environments. Our method showed strong advances in that regard and is a step towards reaching the required grasp reliability.

6 System Applications

In this chapter, I showcase several applications that were enabled in and outside the robotic laboratory at DLR by the object pose estimation (Chapter 4) and grasp pose estimation (Chapter 5) methods presented in the previous sections.

6.1 6-DoF Object Pose Estimation

6.1.1 Integration on Embedded Hardware

The presented AAEs were ported onto an embedded Nvidia Jetson TX2 board, together with a small footprint MobileNet [94] for bounding box detection. The low power consumption (7.5W) of such devices is crucial to the feasibility of mobile robotic applications. A webcam was connected, and this setup was demonstrated live at ECCV 2018, both in the demo session and during the oral presentation. For this demo we acquired several of the T-LESS objects. As can be seen in Figure 6.1, the lighting conditions are quite different from the test sequences of the T-LESS dataset which validates the robustness and applicability of our approach outside lab conditions. No ICP was used, so the errors in depth resulting from the scaling errors of the MobileNet, were not corrected. However, since small errors along the depth direction are less perceptible by humans as well, our approach could be interesting for augmented reality applications. The detection, pose estimation and visualization of the three test objects runs at over 13Hz on the Nvidia Jetson TX2.

6.1.2 Omnirob Integration

In Figure 6.1 (right), we also show an integration of our full 6-DoF object pose estimation pipeline on the mobile Omnirob [110] robotic platform. A pan-tilt mounted RC-visard stereo sensor with a random pattern projector captures an

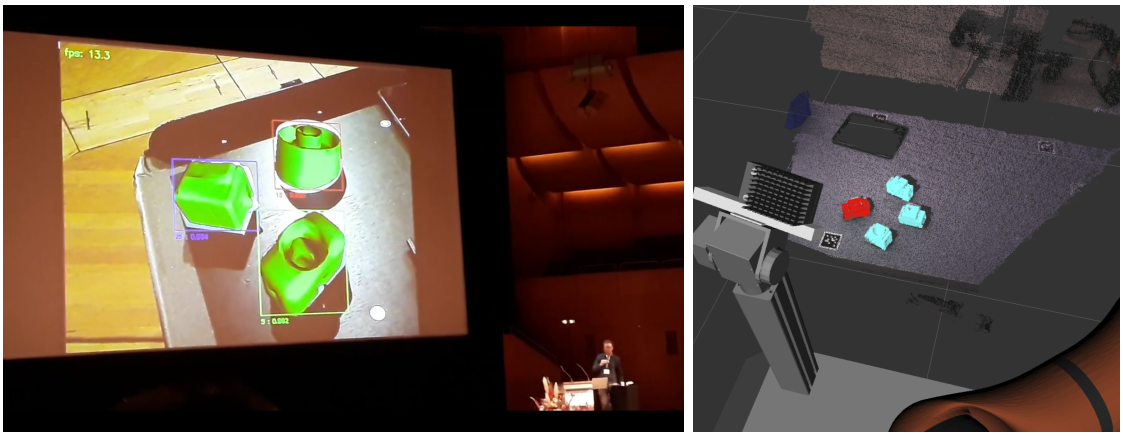


Figure 6.1: Left: MobileNetSSD [94] detection and AAE [124] pose estimation on T-LESS objects, demonstrated live at ECCV 2018 on an Nvidia Jetson TX2; Right: RetinaNet [120] detection and AAE [124] pose estimation on the mobile Omnirob platform visualized on top of the point cloud from an RC-visard stereo sensor.

RGB-D image of four fuses. RetinaNet [120] with AAEs [181] and a subsequent CUDA accelerated ICP [11] are applied to retrieve the 6-DoF object poses that are used for functional manipulation.

6.1.3 Humanoid David Emptying a Dishwasher

Autonomous robots that can assist in household tasks involving manipulation require a rich perception system. In this section, we walk through an end-to-end application involving many of the proposed works that enable the humanoid robot David [196] that was developed at DLR to empty a dishwasher.

In Section 4.7 we introduced the combination of 2D object detection, 6-DoF object pose estimation and 3D tracking. Such a pipeline is well suited for autonomous robotic manipulation tasks as it combines the consistency and accuracy of modern 3D tracking approaches [214] with the automatic global object pose estimation from an AAE [181] based pipeline. The latter ensure that losing track of objects when they are occluded or vanish from the field of view is not an issue anymore.

We first 3D scan five IKEA objects using a laser scanner mounted on top of a Faro arm. Next, we render 50K synthetic images using BlenderProc [133] and save them in the BOP dataset format. We define a range of plausible object materials including roughness, metallicness, specularity and coating. 25K images are rendered from the IKEA and distractor object models dropped into an empty

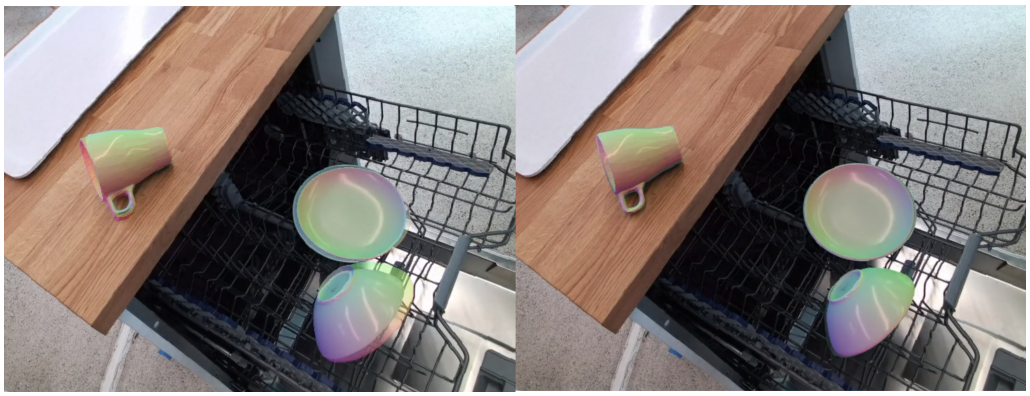


Figure 6.2: Left: Initial Yolov7 [225] + AAEs [181] 6-DoF pose estimates of various dishes. Right: ICG [214] pose refinement and tracking from initial estimates.

textured cube using Blender's physics engine as described in Section 4.6.1.1. The other half is rendered from flying IKEA and distractor object models with HDRi lighting and background maps [161]. We train both, a 2D object detector Yolov7 [225] and AAEs [124] on the created synthetic data. We also extract features for the sparse viewpoint model of ICG [214] from the 3D scanned IKEA objects. The humanoid robot David can then continuously detect new IKEA objects in the camera stream of a Kinect Azure RGB-D ToF sensor mounted to his head. From the 2D detections, initial poses are estimated with AAEs [124] which are then refined and tracked with ICG [124] until tracking loss occurs [212]. A successful dish grasp is displayed in Figure 1.1 and a video can be found here.

6.1.4 EDAN at CYBATHLON 2023

The EDAN wheelchair robot [182] is an assistive robotics system for people with severe motor impairments. A robot arm mounted on the wheelchair can be controlled using EMG muscle signals or inputs from a 3D space mouse [24]. However, since highly accurate steering in 6-DoF end-effector space is challenging, the systems implements a shared-autonomy approach that helps the user executing tasks with the assistance of a perception system. Specifically, a similar 6-DoF object pose estimation pipeline as for the David robot (Section 6.1.3) is implemented for EDAN as shown in Figure 6.3. This enables autonomous, functional grasping of known objects and actions such as opening doors, fridges or drawers.

The CYBATHLON [82], a non-profit project of ETH Zurich, acts as a platform that challenges teams from all over the world to develop assistive technologies suitable

Figure 6.3: EDAN at CYBATHLON 2023. Left: 6-DoF object pose estimates from Yolov7 [225], AAEs [181] and ICG [214] of the challenge objects. Right: Shared-autonomy pick and place tasks during the CYBATHTLON challenge. Matthias steers the arm towards the object and EDAN executes successful grasp using the estimated 6-DoF pose.

for everyday use with and for people with disabilities. In the 2023 CYBATHLON challenges, DLR's EDAN team won the assistant robot race with the pilot Mattias Atzenhofer. During the challenge, the 6-DoF object pose estimation pipeline was deployed to locate five different objects of various sizes in an IKEA shelf. All training data is again synthetically generated using BlenderProc [133]. During the challenge all objects were picked up and placed on top of the shelf with a 100% success rate in two subsequent rounds in a total of 4:29 minutes and the team won the first place.

6.2 Unknown Object Manipulation

6.2.1 Stereo Grasp Pipeline on the RACE-LAB System

The RACE-LAB system consists of a 7-DoF robot arm with an RobotiQ gripper mounted on a linear axis. A calibrated RC-Visard stereo sensor with a random pattern projector captures the cluttered scene with unknown objects.

The goals for this experiment were: (1) Test our novel Instance Stereo Transformer (INSTR) [193] for unknown object segmentation and replace the Unseen Clustering Network (UCN) [183] that is designed for RGB-D sensors and does less well with stereo depth. Use the segmentation masks to set region of interest and filter the grasps predicted by Contact-GraspNet [198]. (2) Test Contact-GraspNet's generalization to different depth data and different gripper types, i.e. RobotiQ. (3) Grasp objects from more severe clutter and test the feasibility of grasping

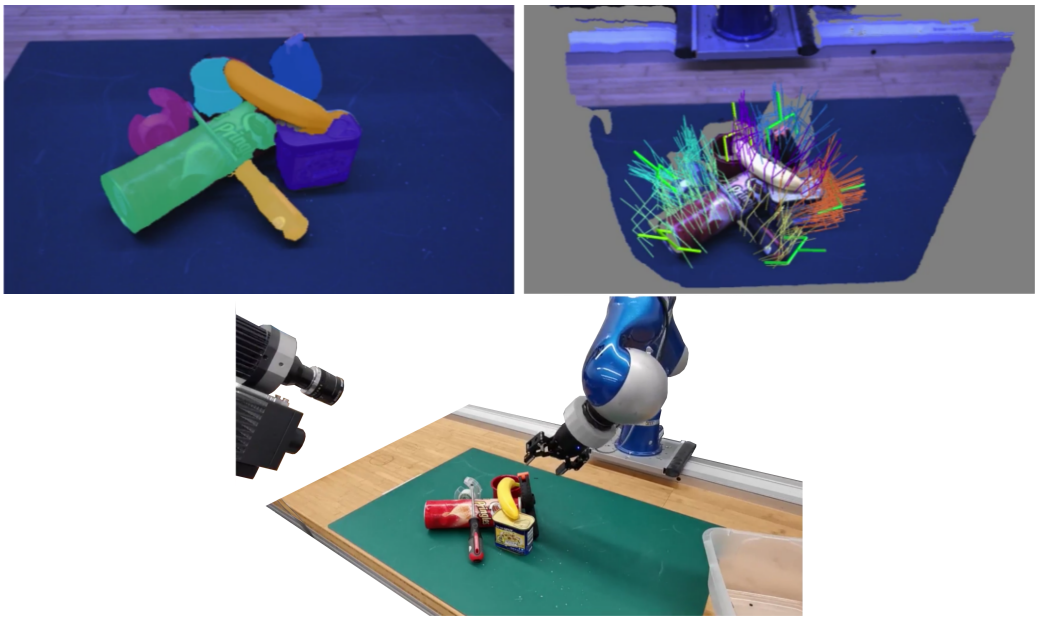


Figure 6.4: Top Left: Unknown instance masks predicted by INSTR [193]. Top Right: 6-DoF Grasp distribution predicted by Contact-GraspNet [198]. Bottom: Executed grasps on the Racelab system.

objects with metallic, dark and transparent materials.

While Contact-GraspNet [198] is trained on physics simulated grasps with a Pandas gripper, the predictions can be adapted to other gripper models with arbitrary gripper depth as well as equal or smaller gripper widths such as the RobotiQ gripper. A simple method is to scale the input point-cloud proportionally to the relative gripper widths and scale back the 6-DoF grasp predictions to match the original point cloud. A retraining with collision checks for the target gripper model instead of the original Panda model has shown to further improve results. Figure 6.4 displays the successful unknown object segmentation by INSTR and the subsequent 6-DoF grasp predictions. The whole scene could be cleared with only one repeated grasp attempt.

As shown in Figure 7.1, INSTR [193] and Contact-GraspNet [198] show improved robustness on transparent, dark and metallic objects over the original time-of-flight RGB-D sensor based pipeline since stereo is less susceptible to such materials. In fact, all objects were successfully picked up and placed in a bin in two or fewer attempts.

7 Conclusion and Future Work

In this dissertation we addressed the problem of 6-DoF Object Pose and 6-DoF Grasp Pose estimation from visual sensor data. Specifically, we introduced new learning-based methods that are not only fast and reliable but are also scalable in terms of training data, test environments and number of target objects. To achieve scalability in real world applications such as robotic manipulation and Augmented Reality, we broke with the practice of training on real pose annotated data and trained our approaches in simulation where abundant data with steerable variation can be generated.

First, we introduced Augmented Autoencoders, one of the first deep learning based methods that reached competitive performance in 6-DoF object pose estimation and resolved the issue of pose ambiguities through an implicit, appearance based training. The approach incorporates sim2real transfer techniques that help it generalize to diverse environments and sensors despite being solely trained on synthetic data. The approach won the Best Paper Award at ECCV 2018 where we also showcased its real-time capabilities in a live demonstration on embedded hardware.

Second, we introduced Multi-Path Encoders, an extension to the Augmented Autoencoders where one joint encoder learns to extract view-sensitive features for n objects at once while being supervised by n object-specific decoders. Thereby, we were able to efficiently scale the number of objects and even estimate the pose of objects unseen during training. Since decoders can be discarded after training, the method preserves a small footprint at inference time.

Third, to encourage broader, more realistic and unified evaluations in the field, we joined the organization of the Benchmark for 6D Object Pose Estimation (BOP Challenge). In 2019, we also participated ourselves with the Augmented Autoencoders that won the RGB-only track of the BOP challenge and were the overall best method with an inference time below one second. However, despite the progress achieved with domain adaptation and randomization techniques, we

observed that the gap between synthetic training data and real test environments limits the potential of all learning-based 6-DoF pose estimation methods. On datasets without real pose annotated training data, classical methods such as PPF based on depth data were still superior in 2019.

Fourth, in order to overcome the domain gap we developed BlenderProc, a new open-source tool for realistic, procedural, synthetic data generation and provided a total of 350K PBR training images of the seven core datasets to the participants of the BOP challenge. As a result, learning-based methods could for the first time outperform classical methods without real training data. In direct comparison, the winning method in 2020, CosyPose [173], showed an absolute improvement of 25.5% in Average Recall when trained on the PBR images from BlenderProc over OpenGL rendered images. In 2022, the sim2real gap further shrank to merely 2.5% when comparing training on BlenderProc PBR data to training on both, real and PBR data. We also extended BlenderProc with forward and inverse kinematics utilities to create the "Robot Tracking Benchmark (RTB)", a photorealistic test dataset for articulated 6-DoF object tracking and provided a strong baseline [222].

Fifth, along with the continual organization of the BOP challenges and parallel workshops on "Recovering 6D Pose" at the ECCV / ICCV conferences, we have reviewed and analyzed the results and developments in the field [223, 168]. In the BOP challenge 2022, we evaluated the influence of object detection of and segmentation methods on 6-DoF object pose estimation results. In the BOP challenge 2023, we introduced three new tasks on evaluating unseen object detection, segmentation and 6-DoF pose estimation. Similar to the Multi-Path Encoder, submitted methods had to generalize to novel objects that had not been seen during training but only in a short 5-minute onboarding phase. In practice this allows robots to quickly adapt to novel objects without lengthy training or synthetic data generation phases.

Sixth, we introduced Contact-GraspNet, an end-to-end network that efficiently generates a distribution of 6-DoF parallel-jaw grasps directly from a single depth view. In combination with unknown object segmentation, we showed over 90% success rates in grasping unknown objects from cluttered scenes. We found a representation that roots the 6-DoF grasps in the visible point cloud which ensures high accuracy and coverage and at the same time reduces the dimensionality of the learning problem.

Figure 7.1: The use of stereo images and algorithms improves performance on transparent and black materials over ToF images.

Finally, we showcased the presented methods in action on different robotic systems. For example, our 6-DoF object pose estimation enabled the humanoid robot David to perform tasks such as emptying a dishwasher. They also empowered the wheelchair robot EDAN during its winning run of the CYBATHLON 2023 [82], where objects had to be located and picked up from shelves. The 6-DoF unknown object grasping methods were applied e.g. to the Lightweight Rover Unit (LRU) for collecting objects such as rock samples as well as on the industrial SARA arm for picking up objects from a conveyor belt. Since all presented methods have corresponding open source code, we hope to inspire other researchers and enable practitioners to tackle more real-world applications outside the laboratory.

Of course, research always continues and there are exciting future directions ahead. One emerging research area is few-shot object pose estimation, i.e. instead of referencing test images with 3D object models, they are directly matched against object template views [216, 213]. Based on this, we are currently developing a method to learn functional manipulation for unknown object from a single kinesthetic demonstration and one or more template views. Other open challenges are robust predictions on transparent [207], deformable or articulated objects. To enable autonomous robotic manipulation outside the lab, global 6-DoF grasp predictions such as Contact-GraspNet could also be combined with local, closed-loop, force-controlled grasping. Finally, extending object grasping to multi-finger hands is another interesting avenue [218].

Bibliography

[1] J. Grunert. "Das Pothenotische Problem in erweiterter Gestalt nebst seine Anwendungen in der Geodäsie". In: *Grunerts Archiv für Mathematik und Physik*. 1841.

[2] L. G. Roberts. "Machine perception of three-dimensional solids". PhD thesis. Massachusetts Institute of Technology, 1963.

[3] D. Brown. "Decentering distortion of lenses, D. Brown Associates". In: *Inc, Eau Gallie, Florida, Tech. Rep* (1966).

[4] B. T. Phong. "Illumination for computer generated pictures". In: *Communications of the ACM* 18.6 (1975), pp. 311–317.

[5] W. Kabsch. "A solution for the best rotation to relate two sets of vectors". In: *Acta Crystallographica Section A: Crystal Physics, Diffraction, Theoretical and General Crystallography* 32.5 (1976), pp. 922–923.

[6] M. A. Fischler and R. C. Bolles. "Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography". In: *Communications of the ACM* 24.6 (1981), pp. 381–395.

[7] D. E. Rumelhart, G. E. Hinton, and R. J. Williams. *Learning internal representations by error propagation*. Tech. rep. California Univ San Diego La Jolla Inst for Cognitive Science, 1985.

[8] D. G. Lowe. "Three-dimensional object recognition from single two-dimensional images". In: *Artificial intelligence* 31.3 (1987), pp. 355–395.

[9] R. Y. Tsai, R. K. Lenz, et al. "A new technique for fully autonomous and efficient 3D robotics hand/eye calibration". In: *IEEE Transactions on robotics and automation* 5.3 (1989), pp. 345–358.

[10] Y. Chen and G. Medioni. "Object modelling by registration of multiple range images". In: *Image and vision computing* 10.3 (1992), pp. 145–155.

[11] Z. Zhang. "Iterative point matching for registration of free-form curves and surfaces". In: *International Journal of Computer Vision* 13.2 (1994), pp. 119–152.

[12] D. F. DeMenthon and L. S. Davis. "Model-based object pose in 25 lines of code". In: *International Journal of Computer Vision* 15 (1995), pp. 123–141.

[13] D. G. Lowe. "Object recognition from local scale-invariant features". In: *Proceedings of the seventh IEEE International Conference on Computer Vision*. Vol. 2. Ieee. 1999, pp. 1150–1157.

[14] A. Bicchi and V. Kumar. "Robotic grasping and contact: a review". In: *Proceedings 2000 ICRA. Millennium Conference. IEEE International Conference on Robotics and Automation. Symposia Proceedings (Cat. No.00CH37065)*. Vol. 1. 2000, 348–353 vol.1. DOI: 10.1109/ROBOT.2000.844081.

[15] G. Bradski. "The OpenCV Library". In: *Dr. Dobb's Journal of Software Tools* (2000).

[16] F. Jurie and M. Dhome. "Real Time Robust Template Matching". In: *The 13th British Machine Vision Conference (BMVC'02)*. The British Machine Vision Association. 2002, pp. 123–132.

[17] V. Lepetit, J. Pilet, and P. Fua. "Point matching as a classification problem for fast and robust object pose estimation". In: *Proceedings of the 2004 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 2004. CVPR 2004.* Vol. 2. IEEE. 2004, pp. II–II.

[18] H. Hirschmüller. "Accurate and Efficient Stereo Processing by Semi-Global Matching and Mutual Information". In: *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*. Vol. 2. 2005, pp. 807–814.

[19] H. Bay, T. Tuytelaars, and L. Van Gool. "Surf: Speeded up robust features". In: *Lecture notes in computer science* 3951 (2006), pp. 404–417.

[20] S. Hinterstoisser, S. Benhimane, V. Lepetit, P. Fua, and N. Navab. "Simultaneous Recognition and Homography Extraction of Local Patches with a Simple Linear Classifier". In: *Proceedings of the British Machine Conference, pages*. 2008, pp. 10–1.

[21] D. Prattichizzo and J. C. Trinkle. "Grasping". In: *Springer Handbook of Robotics*. Ed. by B. Siciliano and O. Khatib. Berlin, Heidelberg: Springer Berlin Heidelberg, 2008, pp. 671–700. ISBN: 978-3-540-30301-5. DOI: 10.1007/978-3-540-30301-5_29. URL: https://doi.org/10.1007/978-3-540-30301-5_29.

[22] N. Hansen. "Benchmarking a BI-Population CMA-ES on the BBOB-2009 Function Testbed". In: *Workshop Proceedings of the GECCO Genetic and Evolutionary Computation Conference*. ACM, July 2009, pp. 2389–2395.

[23] V. Lepetit, F. Moreno-Noguer, and P. Fua. "Epnp: An accurate o (n) solution to the pnp problem". In: *International Journal of Computer Vision* 81.2 (2009), pp. 155–166.

[24] M. Rahman, S. Gustafson, P. Irani, and S. Subramanian. "Tilt techniques: investigating the dexterity of wrist-based input". In: *Proceedings of the SIGCHI conference on human factors in computing systems*. 2009, pp. 1943–1952.

[25] M. Ulrich, C. Wiedemann, and C. Steger. "CAD-based recognition of 3D objects in monocular images." In: *ICRA*. Vol. 9. 2009, pp. 1191–1198.

[26] B. Drost, M. Ulrich, N. Navab, and S. Ilic. "Model globally, match locally: Efficient and robust 3D object recognition". In: *2010 IEEE computer society Conference on Computer Vision and Pattern Recognition*. Ieee. 2010, pp. 998–1005.

[27] M. Everingham, L. Van Gool, C. K. Williams, J. Winn, and A. Zisserman. "The pascal visual object classes (voc) challenge". In: *International Journal of Computer Vision* 88.2 (2010), pp. 303–338.

[28] X. Glorot and Y. Bengio. "Understanding the difficulty of training deep feedforward neural networks". In: *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics*. 2010, pp. 249–256.

[29] S. Hinterstoisser, V. Lepetit, S. Ilic, P. Fua, and N. Navab. "Dominant orientation templates for real-time detection of texture-less objects". In: *2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*. IEEE. 2010, pp. 2257–2264.

[30] Y. LeCun and C. Cortes. "MNIST handwritten digit database". In: (2010). URL: http://yann.lecun.com/exdb/mnist/.

[31] P. Vincent, H. Larochelle, I. Lajoie, Y. Bengio, and P.-A. Manzagol. "Stacked denoising autoencoders: Learning useful representations in a deep network with a local denoising criterion". In: *Journal of Machine Learning Research* 11.Dec (2010), pp. 3371–3408.

[32] S. Hinterstoisser, S. Holzer, C. Cagniart, S. Ilic, K. Konolige, N. Navab, and V. Lepetit. "Multimodal templates for real-time detection of texture-less objects in heavily cluttered scenes". In: *Computer Vision (ICCV), 2011 IEEE International Conference on*. IEEE. 2011, pp. 858–865.

[33] M. Krainin, P. Henry, X. Ren, and D. Fox. "Manipulator and Object Tracking for In-hand 3D Object Modeling". In: *International Journal of Robotics Research* 30.11 (2011), pp. 1311–1327.

[34] R. A. Newcombe, S. Izadi, O. Hilliges, D. Molyneaux, D. Kim, A. J. Davison, P. Kohi, J. Shotton, S. Hodges, and A. Fitzgibbon. "KinectFusion: Real-time dense surface mapping and tracking". In: *Mixed and augmented reality (ISMAR), 2011 10th IEEE international symposium on*. IEEE. 2011, pp. 127–136.

[35] C. Papazov and D. Burschka. "An efficient ransac for 3d object recognition in noisy and occluded scenes". In: *Computer Vision–ACCV 2010: 10th Asian Conference on Computer Vision, Queenstown, New Zealand, November 8-12, 2010, Revised Selected Papers, Part I 10*. Springer. 2011, pp. 135–148.

[36] R. B. Rusu and S. Cousins. "3D is here: Point Cloud Library (PCL)". In: *IEEE International Conference on Robotics and Automation (ICRA)*. Shanghai, China, May 2011.

[37] M. Everingham, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman. *The PASCAL Visual Object Classes Challenge 2012 (VOC2012) Results*. 2012.

[38] S. Hinterstoisser, V. Lepetit, S. Ilic, S. Holzer, G. Bradski, K. Konolige, and N. Navab. "Model Based Training, Detection and Pose Estimation of Texture-Less 3D Objects in Heavily Cluttered Scenes". In: *ACCV* (2012).

[39] S. Hinterstoisser, C. Cagniart, S. Ilic, P. Sturm, N. Navab, P. Fua, and V. Lepetit. "Gradient response maps for real-time detection of textureless objects". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 34.5 (2012), pp. 876–888.

[40] S. Hinterstoisser, V. Lepetit, S. Ilic, S. Holzer, G. Bradski, K. Konolige, and N. Navab. "Model based training, detection and pose estimation of texture-less 3D objects in heavily cluttered scenes". In: *Asian conference on computer vision*. Springer. 2012, pp. 548–562.

[41] A. Krizhevsky, I. Sutskever, and G. E. Hinton. "Imagenet classification with deep Convolutional Neural Networks". In: *Advances in neural information processing systems*. 2012, pp. 1097–1105.

[42] M. Klingensmith, T. Galluzzo, C. M. Dellin, M. Kazemi, J. A. Bagnell, and N. Pollard. *Closed-loop Servoing using Real-time Markerless Arm Tracking*. 2013.

[43] F. Pomerleau, F. Colas, R. Siegwart, and S. Magnenat. "Comparing ICP Variants on Real-World Data Sets". In: *Autonomous Robots* 34.3 (Feb. 2013), pp. 133–148.

[44] J. Bohg, J. Romero, A. Herzog, and S. Schaal. "Robot Arm Pose Estimation through Pixel-Wise Part Classification". In: *IEEE International Conference on Robotics and Automation*. 2014, pp. 3143–3150.

[45] E. Brachmann, A. Krull, F. Michel, S. Gumhold, J. Shotton, and C. Rother. "Learning 6D object pose estimation using 3D object coordinates". In: *European Conference on Computer Vision*. Springer. 2014, pp. 536–551.

[46] M. Cimpoi, S. Maji, I. Kokkinos, S. Mohamed, and A. Vedaldi. "Describing textures in the wild". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2014, pp. 3606–3613.

[47] J. Donahue, Y. Jia, O. Vinyals, J. Hoffman, N. Zhang, E. Tzeng, and T. Darrell. "Decaf: A deep convolutional activation feature for generic visual recognition". In: *International conference on machine learning*. 2014, pp. 647–655.

[48] M. Elbanhawi and M. Simic. "Sampling-Based Robot Motion Planning: A Review". In: *IEEE Access* 2 (2014), pp. 56–77.

[49] K. Hang, J. A. Stork, and D. Kragic. "Hierarchical Fingertip Space for multi-fingered precision grasping". In: *2014 IEEE/RSJ International Conference on Intelligent Robots and Systems*. 2014, pp. 1641–1648. DOI: 10.1109/IROS.2014.6942775.

[50] D. Kingma and J. Ba. "Adam: A method for stochastic optimization". In: *arXiv preprint arXiv:1412.6980* (2014).

[51] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick. "Microsoft coco: Common objects in context". In: *European conference on computer vision*. Springer. 2014, pp. 740–755.

[52] M. Macklin, M. Müller, N. Chentanez, and T.-Y. Kim. "Unified particle physics for real-time applications". In: *ACM Transactions on Graphics (TOG)* 33.4 (2014), pp. 1–12.

[53] M. Oquab, L. Bottou, I. Laptev, and J. Sivic. "Learning and transferring mid-level image representations using convolutional neural networks". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2014, pp. 1717–1724.

[54] S. Salti, F. Tombari, and L. Di Stefano. "SHOT: Unique signatures of histograms for surface and texture description". In: *Computer Vision and Image Understanding* 125 (2014), pp. 251–264.

[55] A. Tejani, D. Tang, R. Kouskouridas, and T.-K. Kim. "Latent-class hough forests for 3D object detection and pose estimation". In: *ECCV* (2014).

[56] A. X. Chang, T. Funkhouser, L. Guibas, P. Hanrahan, Q. Huang, Z. Li, S. Savarese, M. Savva, S. Song, H. Su, et al. "Shapenet: An information-rich 3d model repository". In: *arXiv preprint arXiv:1512.03012* (2015).

[57] D. Fischinger, A. Weiss, and M. Vincze. "Learning grasps with topographic features". In: *The International Journal of Robotics Research* 34.9 (2015), pp. 1167–1194. DOI: 10.1177/0278364915577105.

[58] A. Kendall, M. Grimes, and R. Cipolla. "Posenet: A convolutional network for real-time 6-DoF camera relocalization". In: *Proceedings of the IEEE International Conference on Computer Vision*. 2015, pp. 2938–2946.

[59] I. Lenz, H. Lee, and A. Saxena. "Deep learning for detecting robotic grasps". In: *IJRR* (2015).

[60] S. Marschner and P. Shirley. *Fundamentals of computer graphics*. CRC Press, 2015.

[61] F. Michel, A. Krull, E. Brachmann, M. Yang, S. Gumhold, and C. Rother. "Pose Estimation of Kinematic Chain Instances via Object Coordinate Regression". In: *British Machine Vision Conference*. 2015, pp. 181.1–181.11.

[62] K. Pauwels and D. Kragic. "SimTrack: A Simulation-based Framework for Scalable Real-time Object Pose Detection and Tracking". In: *IEEE/RSJ International Conference on Intelligent Robots and Systems*. 2015, pp. 1300–1307.

[63] S. Ren, K. He, R. Girshick, and J. Sun. "Faster r-cnn: Towards real-time object detection with region proposal networks". In: *Advances in neural information processing systems*. 2015, pp. 91–99.

[64] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, A. C. Berg, and L. Fei-Fei. "ImageNet Large Scale Visual Recognition Challenge". In: *International Journal of Computer Vision (IJCV)* 115.3 (2015), pp. 211–252. DOI: 10.1007/s11263-015-0816-y.

[65] T. Schmidt, K. Hertkorn, R. Newcombe, Z. Marton, M. Suppa, and D. Fox. "Depth-based tracking with physical constraints for robot manipulation". In: *IEEE International Conference on Robotics and Automation*. 2015, pp. 119–126.

[66] H. Su, C. R. Qi, Y. Li, and L. J. Guibas. "Render for CNN: Viewpoint estimation in images using CNNs trained with rendered 3D model views". In: *Proceedings of the IEEE International Conference on Computer Vision*. 2015, pp. 2686–2694.

[67] P. Wohlhart and V. Lepetit. "Learning descriptors for object recognition and 3D pose estimation". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2015, pp. 3109–3118.

[68] 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.

[69] E. Brachmann, F. Michel, A. Krull, M. Ying Yang, S. Gumhold, et al. "Uncertainty-driven 6D pose estimation of objects and scenes from a single RGB image". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2016, pp. 3364–3372.

[70] A. Doumanoglou, R. Kouskouridas, S. Malassiotis, and T.-K. Kim. "Recovering 6D Object Pose and Predicting Next-Best-View in the Crowd". In: *CVPR* (2016).

[71] J. Hao, J. Dong, W. Wang, and T. Tan. "What is the best practice for cnns applied to visual instance retrieval?" In: *arXiv preprint arXiv:1611.01640* (2016).

[72] K. He, X. Zhang, S. Ren, and J. Sun. "Deep residual learning for image recognition". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2016, pp. 770–778.

[73] S. Hinterstoisser, V. Lepetit, N. Rajkumar, and K. Konolige. "Going further with Point Pair Features". In: *European Conference on Computer Vision*. Springer. 2016, pp. 834–848.

[74] T. Hodaň, J. Matas, and Š. Obdržálek. "On evaluation of 6D object pose estimation". In: *European Conference on Computer Vision*. Springer. 2016, pp. 606–619.

[75] W. Kehl, F. Milletari, F. Tombari, S. Ilic, and N. Navab. "Deep learning of local RGB-D patches for 3D object detection and 6D pose estimation". In: *European Conference on Computer Vision*. Springer. 2016, pp. 205–220.

[76] S. Levine, P. Pastor, A. Krizhevsky, and D. Quillen. "Learning Hand-Eye Coordination for Robotic Grasping with Deep Learning and Large-Scale Data Collection". In: *International Symposium on Experimental Robotics (ISER)* (2016).

[77] W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. Reed, C.-Y. Fu, and A. C. Berg. "SSD: Single shot multibox detector". In: *European Conference on Computer Vision*. Springer. 2016, pp. 21–37.

[78] Y. Movshovitz-Attias, T. Kanade, and Y. Sheikh. "How useful is photo-realistic rendering for visual learning?" In: *European Conference on Computer Vision*. Springer. 2016, pp. 202–217.

[79] M. Pharr, W. Jakob, and G. Humphreys. *Physically based rendering: From theory to implementation*. Morgan Kaufmann, 2016.

[80] C. Rennie, R. Shome, K. E. Bekris, and A. F. De Souza. "A Dataset for Improved RGBD-Based Object Detection and Pose Estimation for Warehouse Pick-and-Place". In: *RA-L* (2016).

[81] S. R. Richter, V. Vineet, S. Roth, and V. Koltun. "Playing for data: Ground truth from computer games". In: *European Conference on Computer Vision*. Springer. 2016, pp. 102–118.

[82] R. Riener. "The Cybathlon promotes the development of assistive technology for people with physical disabilities". In: *Journal of neuroengineering and rehabilitation* 13 (2016), pp. 1–4.

[83] J. L. Schönberger and J.-M. Frahm. "Structure-from-Motion Revisited". In: *Conference on Computer Vision and Pattern Recognition (CVPR)*. 2016.

[84] Z. Wu, C. Shen, and A. v. d. Hengel. "Bridging category-level and instance-level semantic image segmentation". In: *arXiv preprint arXiv:1605.06885* (2016).

[85] V. Balntas, A. Doumanoglou, C. Sahin, J. Sock, R. Kouskouridas, and T.-K. Kim. "Pose Guided RGB-D Feature Learning for 3D Object Pose Estimation". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2017, pp. 3856–3864.

[86] 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)*. Vol. 1. 2017, p. 7.

[87] G. Csurka. "Domain adaptation for visual applications: A comprehensive survey". In: *arXiv preprint arXiv:1702.05374* (2017).

[88] B. Drost, M. Ulrich, P. Bergmann, P. Hartinger, and C. Steger. "Introducing MVTec ITODD – A dataset for 3D object recognition in industry". In: *ICCVW* (2017).

[89] D. Dwibedi, I. Misra, and M. Hebert. "Cut, paste and learn: Surprisingly easy synthesis for instance detection". In: *ICCV* (2017).

[90] K. He, G. Gkioxari, P. Dollár, and R. Girshick. "Mask R-CNN". In: *ICCV*. 2017.

[91] S. Hinterstoisser, V. Lepetit, P. Wohlhart, and K. Konolige. "On Pre-Trained Image Features and Synthetic Images for Deep Learning". In: *arXiv preprint arXiv:1710.10710* (2017).

[92] T. Hodan. *SIXD Challenge 2017*, http://cmp.felk.cvut.cz/sixd/challenge_2017/. 2017. (Visited on 06/06/2017).

[93] T. Hodaň, P. Haluza, Š. Obdržálek, J. Matas, M. Lourakis, and X. Zabulis. "T-LESS: An RGB-D Dataset for 6D Pose Estimation of Texture-less Objects". In: *IEEE Winter Conference on Applications of Computer Vision (WACV)* (2017).

[94] A. G. Howard, M. Zhu, B. Chen, D. Kalenichenko, W. Wang, T. Weyand, M. Andreetto, and H. Adam. "Mobilenets: Efficient convolutional neural networks for mobile vision applications". In: *arXiv preprint arXiv:1704.04861* (2017).

[95] W. Kehl, F. Manhardt, F. Tombari, S. Ilic, and N. Navab. "SSD-6D: Making RGB-Based 3D Detection and 6D Pose Estimation Great Again". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2017, pp. 1521–1529.

[96] S. Mahendran, H. Ali, and R. Vidal. "3d pose regression using convolutional neural networks". In: *Proceedings of the IEEE International Conference on Computer Vision Workshops*. 2017, pp. 2174–2182.

[97] J. Mahler, J. Liang, S. Niyaz, M. Laskey, R. Doan, X. Liu, J. A. Ojea, and K. Goldberg. "Dex-Net 2.0: Deep Learning to Plan Robust Grasps with Synthetic Point Clouds and Analytic Grasp Metrics". In: *RSS* (2017).

[98] L. Matthey, I. Higgins, D. Hassabis, and A. Lerchner. *dSprites: Disentanglement testing Sprites dataset*. https://github.com/deepmind/dsprites-dataset/. 2017.

[99] C. Mitash, K. E. Bekris, and A. Boularias. "A self-supervised learning system for object detection using physics simulation and multi-view pose estimation". In: *Intelligent Robots and Systems (IROS), 2017 IEEE/RSJ International Conference on*. IEEE. 2017, pp. 545–551.

[100] A. Pas, M. Gualtieri, K. Saenko, and R. Platt. "Grasp pose detection in point clouds". In: *The International Journal of Robotics Research* 36.13-14 (2017), pp. 1455–1473.

[101] C. R. Qi, L. Yi, H. Su, and L. J. Guibas. "PointNet++: Deep Hierarchical Feature Learning on Point Sets in a Metric Space". In: *Neural Information Processing Systems (NeurIPS)* (2017).

[102] M. Rad and V. Lepetit. "BB8: A scalable, accurate, robust to partial occlusion method for predicting the 3D poses of challenging objects without using depth". In: *ICCV* (2017).

[103] C. Raposo and J. P. Barreto. "Using 2 point+normal sets for fast registration of point clouds with small overlap". In: *ICRA* (2017).

[104] A. Shrivastava, T. Pfister, O. Tuzel, J. Susskind, W. Wang, and R. Webb. "Learning from Simulated and Unsupervised Images through Adversarial Training". In: *Computer Vision and Pattern Recognition (CVPR), 2017 IEEE Conference on*. IEEE. 2017, pp. 2242–2251.

[105] H. Tjaden, U. Schwanecke, and E. Schomer. "Real-time monocular pose estimation of 3D objects using temporally consistent local color histograms". In: *Proceedings of the IEEE International Conference on Computer Vision*. 2017, pp. 124–132.

[106] 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: *Intelligent Robots and Systems (IROS), 2017 IEEE/RSJ International Conference on*. IEEE. 2017, pp. 23–30.

[107] Y. Xiang, T. Schmidt, V. Narayanan, and D. Fox. "Posecnn: A convolutional neural network for 6d object pose estimation in cluttered scenes". In: *arXiv preprint arXiv:1711.00199* (2017).

[108] D. Barath and J. Matas. "Graph-cut RANSAC". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2018, pp. 6733–6741.

[109] K. Bousmalis, A. Irpan, P. Wohlhart, Y. Bai, M. Kelcey, M. Kalakrishnan, L. Downs, J. Ibarz, P. Pastor, K. Konolige, et al. "Using simulation and domain adaptation to improve efficiency of deep robotic grasping". In: *2018 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE. 2018, pp. 4243–4250.

[110] M. Brucker, M. Durner, Z. Marton, F. Bálint-Benczédi, **M. Sundermeyer**, and R. Triebel. "6DoF Pose Estimation for Industrial Manipulation based on Synthetic Data". In: *International Symposium on Experimental Robotic (ISER)*. 2018.

[111] B. O. Community. *Blender – a 3D modelling and rendering package*. Blender Foundation, 2018. URL: http://www.blender.org.

[112] K. Fang, Y. Zhu, A. Garg, A. Kuryenkov, V. Mehta, L. Fei-Fei, and S. Savarese. "Learning Task-Oriented Grasping for Tool Manipulation from Simulated Self-Supervision". In: *Robotics: Science and Systems (RSS)* (2018).

[113] R. Geirhos, P. Rubisch, C. Michaelis, M. Bethge, F. A. Wichmann, and W. Brendel. "ImageNet-trained CNNs are biased towards texture; increasing shape bias improves accuracy and robustness". In: *arXiv preprint arXiv:1811.12231* (2018).

[114] T. Hodan, F. Michel, E. Brachmann, W. Kehl, A. GlentBuch, D. Kraft, B. Drost, J. Vidal, S. Ihrke, X. Zabulis, et al. "BOP: benchmark for 6D object pose estimation". In: *Proceedings of the European Conference on Computer Vision (ECCV)*. 2018, pp. 19–34.

[115] D. Kalashnikov, A. Irpan, P. Pastor, J. Ibarz, A. Herzog, E. Jang, D. Quillen, E. Holly, M. Kalakrishnan, V. Vanhoucke, and S. Levine. "QT-Opt: Scalable Deep Reinforcement Learning for Vision-Based Robotic Manipulation". In: *Conference on Robot Learning* (2018).

[116] L. Kiforenko, B. Drost, F. Tombari, N. Krüger, and A. G. Buch. "A performance evaluation of point pair features". In: *Computer Vision and Image Understanding* 166 (2018), pp. 66–80.

[117] A. Kundu, Y. Li, and J. M. Rehg. "3d-rcnn: Instance-level 3d object reconstruction via render-and-compare". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2018, pp. 3559–3568.

[118] C. Li, J. Bai, and G. D. Hager. "A unified framework for multi-view multi-class object pose estimation". In: *Proceedings of the European Conference on Computer Vision (ECCV)*. 2018, pp. 254–269.

[119] Y. Li, G. Wang, X. Ji, Y. Xiang, and D. Fox. "Deepim: Deep iterative matching for 6d pose estimation". In: *Proceedings of the European Conference on Computer Vision (ECCV)*. 2018, pp. 683–698.

[120] T.-Y. Lin, P. Goyal, R. Girshick, K. He, and P. Dollár. "Focal loss for dense object detection". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* (2018).

[121] F. Manhardt, W. Kehl, N. Navab, and F. Tombari. "Deep Model-Based 6D Pose Refinement in RGB". In: *The European Conference on Computer Vision (ECCV)*. Sept. 2018.

[122] P. Marion, P. R. Florence, L. Manuelli, and R. Tedrake. "Label Fusion: A Pipeline for Generating Ground Truth Labels for Real RGBD Data of Cluttered Scenes". In: *2018 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE. 2018, pp. 3325–3242.

[123] C. Sahin and T.-K. Kim. "Category-level 6D Object Pose Recovery in Depth Images". In: *Proceedings of the European Conference on Computer Vision (ECCV)*. 2018, pp. 0–0.

[124] **M. Sundermeyer**, Z.-C. Marton, M. Durner, M. Brucker, and R. Triebel. "Implicit 3D Orientation Learning for 6D Object Detection from RGB Images". In: *European Conference on Computer Vision (ECCV)*. 2018.

[125] **M. Sundermeyer**, E. Y. Puang, Z.-C. Marton, M. Durner, and R. Triebel. "Learning Implicit Representations of 3D Object Orientations from RGB". In: *International Conference on Robotics and Automation (ICRA) Workshops*. 2018.

[126] B. Tekin, S. N. Sinha, and P. Fua. "Real-time seamless single shot 6d object pose prediction". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2018, pp. 292–301.

[127] J. Tremblay, T. To, B. Sundaralingam, Y. Xiang, D. Fox, and S. Birchfield. "Deep Object Pose Estimation for Semantic Robotic Grasping of Household Objects". In: *Conference on Robot Learning*. 2018, pp. 306–316.

[128] J. Vidal, C.-Y. Lin, X. Lladó, and R. Marti. "A Method for 6D Pose Estimation of Free-Form Rigid Objects Using Point Pair Features on Range Data". In: *Sensors* (2018).

[129] X. Yan, J. Hsu, M. Khansari, Y. Bai, A. Pathak, A. Gupta, J. Davidson, and H. Lee. "Learning 6-dof grasping interaction via deep geometry-aware 3d representations". In: *2018 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE. 2018, pp. 3766–3773.

[130] A. Zeng, S. Song, S. Welker, J. Lee, A. Rodriguez, and T. Funkhouser. "Learning synergies between pushing and grasping with self-supervised deep reinforcement learning". In: *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE. 2018, pp. 4238–4245.

[131] W. Boerdijk, **M. Sundermeyer**, M. Durner, and R. Triebel. "Self-Supervised Object-in-Gripper Segmentation from Robotic Motions". In: *Conference on Robot Learning (CoRL)*. 2019.

[132] X. Deng, A. Mousavian, Y. Xiang, F. Xia, T. Bretl, and D. Fox. "PoseRBPF: A Rao-Blackwellized Particle Filter for 6D Object Pose Tracking". In: *Robotics: Science and Systems (RSS)*. 2019.

[133] M. Denninger, M. Sundermeyer, D. Winkelbauer, Y. Zidan, D. Olefir, M. Elbadrawy, A. Lodhi, and H. Katam. "Blenderproc". In: *arXiv preprint arXiv:1911.01911* (2019).

[134] T. Do, T. Pham, M. Cai, and I. Reid. "Real-time monocular object instance 6d pose estimation". In: *BMVC Press* (2019).

[135] F. Hagelskjær and A. G. Buch. "PointPoseNet: Accurate Object Detection and 6-DoF Pose Estimation in Point Clouds". In: *arXiv preprint arXiv:1912.09057* (2019).

[136] S. Hinterstoisser, O. Pauly, H. Heibel, M. Martina, and M. Bokeloh. "An annotation saved is an annotation earned: Using fully synthetic training for object detection". In: *ICCVW* (2019).

[137] T. Hodaň, E. Brachmann, B. Drost, F. Michel, M. Sundermeyer, J. Matas, and C. Rother. *BOP Challenge 2019* https://bop.felk.cvut.cz/media/bop_challenge_2019_results.pdf. 2019.

[138] T. Hodaň, V. Vineet, R. Gal, E. Shalev, J. Hanzelka, T. Connell, P. Urbina, S. Sinha, and B. Guenter. "Photorealistic Image Synthesis for Object Instance Detection". In: *IEEE International Conference on Image Processing (ICIP)* (2019).

[139] T. Hodaň, V. Vineet, R. Gal, E. Shalev, J. Hanzelka, T. Connell, P. Urbina, S. Sinha, and B. Guenter. "Photorealistic Image Synthesis for Object Instance Detection". In: *IEEE International Conference on Image Processing (ICIP)* (2019).

[140] R. Kaskman, S. Zakharov, I. Shugurov, and S. Ilic. "HomebrewedDB: RGB-D Dataset for 6D Pose Estimation of 3D Objects". In: *ICCVW* (2019).

[141] Z. Li, G. Wang, and X. Ji. "CDPN: Coordinates-Based Disentangled Pose Network for Real-Time RGB-Based 6-DoF Object Pose Estimation". In: *The IEEE International Conference on Computer Vision (ICCV)*. Oct. 2019.

[142] Z. Li, G. Wang, and X. Ji. "Cdpn: Coordinates-based disentangled pose network for real-time rgb-based 6-dof object pose estimation". In: *Proceedings of the IEEE/CVF International Conference on Computer Vision*. 2019, pp. 7678–7687.

[143] H. Liang, X. Ma, S. Li, M. Görner, S. Tang, B. Fang, F. Sun, and J. Zhang. "PointNetGPD: Detecting Grasp Configurations from Point Sets". In: *IEEE International Conference on Robotics and Automation (ICRA)*. 2019.

[144] J. Mahler, M. Matl, V. Satish, M. Danielczuk, B. DeRose, S. McKinley, and K. Goldberg. "Learning ambidextrous robot grasping policies". In: *Science Robotics* 4.26 (2019), eaau4984.

[145] F. Manhardt, D. M. Arroyo, C. Rupprecht, B. Busam, T. Birdal, N. Navab, and F. Tombari. "Explaining the ambiguity of object detection and 6d pose from visual data". In: *Proceedings of the IEEE/CVF International Conference on Computer Vision*. 2019, pp. 6841–6850.

[146] L. Manuelli, W. Gao, P. Florence, and R. Tedrake. "kPAM: KeyPoint Affordances for Category-Level Robotic Manipulation." In: *International Symposium on Robotics Research (ISRR)* (2019).

[147] A. Mousavian, C. Eppner, and D. Fox. "6-dof graspnet: Variational grasp generation for object manipulation". In: *Proceedings of the IEEE International Conference on Computer Vision*. 2019, pp. 2901–2910.

[148] K. Park, T. Patten, and M. Vincze. "Pix2Pose: Pixel-Wise Coordinate Regression of Objects for 6D Pose Estimation". In: *The IEEE International Conference on Computer Vision (ICCV)*. Oct. 2019.

[149] K. Park, T. Patten, and M. Vincze. "Pix2Pose: Pixel-Wise Coordinate Regression of Objects for 6D Pose Estimation". In: *ICCV* (2019).

[150] S. Peng, Y. Liu, Q. Huang, X. Zhou, and H. Bao. "Pvnet: Pixel-wise voting network for 6dof pose estimation". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2019, pp. 4561–4570.

[151] P. Rodrigues, M. Antunes, C. Raposo, P. Marques, F. Fonseca, and J. Barreto. "Deep segmentation leverages geometric pose estimation in computer-aided total knee arthroplasty". In: *Healthcare Technology Letters* (2019).

[152] M. Sundermeyer, Z.-C. Marton, M. Durner, and R. Triebel. "Augmented Autoencoders: Implicit 3D Orientation Learning for 6D Object Detection". In: *IJCV* (2019).

[153] Z. Tian, C. Shen, H. Chen, and T. He. "Fcos: Fully convolutional one-stage object detection". In: *Proceedings of the IEEE/CVF International Conference on Computer Vision*. 2019, pp. 9627–9636.

[154] C. Wang, D. Xu, Y. Zhu, R. Martin-Martin, C. Lu, L. Fei-Fei, and S. Savarese. "DenseFusion: 6D object pose estimation by iterative dense fusion". In: *CVPR* (2019).

[155] H. Wang, S. Sridhar, J. Huang, J. Valentin, S. Song, and L. J. Guibas. "Normalized object coordinate space for category-level 6d object pose and size estimation". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2019, pp. 2642–2651.

[156] L. Wright. *Ranger - a synergistic optimizer.* https://github.com/lessw2020/Ranger-Deep-Learning-Optimizer. 2019.

[157] C. Xie, Y. Xiang, A. Mousavian, and D. Fox. "The Best of Both Modes: Separately Leveraging RGB and Depth for Unseen Object Instance Segmentation". In: *Conference on Robot Learning (CoRL)*. 2019.

[158] X. Yan, M. Khansari, J. Hsu, Y. Gong, Y. Bai, S. Pirk, and H. Lee. *Data-Efficient Learning for Sim-to-Real Robotic Grasping using Deep Point Cloud Prediction Networks*. 2019. arXiv: 1906.08989 [cs.RO].

[159] S. Zakharov, I. Shugurov, and S. Ilic. "DPOD: 6D Pose Object Detector and Refiner". In: *ICCV* (2019).

[160] Y. Zhou, C. Barnes, J. Lu, J. Yang, and H. Li. "On the continuity of rotation representations in neural networks". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2019, pp. 5745–5753.

[161] L. Demes. *CC0 Textures*. https://cc0textures.com/. 2020.

[162] X. Deng, Y. Xiang, A. Mousavian, C. Eppner, T. Bretl, and D. Fox. "Self-supervised 6D Object Pose Estimation for Robot Manipulation". In: *International Conference on Robotics and Automation (ICRA)*. 2020.

[163] M. Denninger, **M. Sundermeyer**, D. Winkelbauer, D. Olefir, T. Hodan, Y. Zidan, M. Elbadrawy, M. Knauer, H. Katam, and A. Lodhi. "Blenderproc: Reducing the reality gap with photorealistic rendering". In: *International Conference on Robotics Sciene and Systems (RSS) Workshops*. 2020.

[164] M. Denninger and R. Triebel. "3d scene reconstruction from a single viewport". In: *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XXII*. Springer. 2020, pp. 51–67.

[165] H.-S. Fang, C. Wang, M. Gou, and C. Lu. "GraspNet-1Billion: A Large-Scale Benchmark for General Object Grasping". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2020, pp. 11444–11453.

[166] T. Hodan, D. Barath, and J. Matas. "Epos: Estimating 6d pose of objects with symmetries". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2020, pp. 11703–11712.

[167] T. Hodaň and M. Sundermeyer. *BOP Toolkit*. 2020. URL: https://github.com/thodan/bop_toolkit.

[168] T. Hodaň, **M. Sundermeyer**, B. Drost, Y. Labbé, E. Brachmann, F. Michel, C. Rother, and J. Matas. "BOP challenge 2020 on 6D object localization". In: *European Conference on Computer Vision Workshops (ECCVW)*. 2020.

[169] T. Hodaň, M. Sundermeyer, R. Kouskouridas, T.-K. Kim, J. Matas, C. Rother, V. Lepetit, A. Leonardis, K. Walas, C. Steger, E. Brachmann, B. Drost, and J. Sock. *6th International Workshop on Recovering 6D Object Pose*. http://cmp.felk.cvut.cz/sixd/workshop_2020/. 2020.

[170] Y. Hu, P. Fua, W. Wang, and M. Salzmann. "Single-stage 6d object pose estimation". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2020, pp. 2930–2939.

[171] *Intel Open Image Denoise*. 2020. URL: https://www.openimagedenoise.org/.

[172] R. Koenig and B. Drost. "A Hybrid Approach for 6DoF Pose Estimation". In: *ECCVW* (2020).

[173] Y. Labbé, J. Carpentier, M. Aubry, and J. Sivic. "CosyPose: Consistent multi-view multi-object 6D pose estimation". In: *ECCV* (2020).

[174] J. Levinson, C. Esteves, K. Chen, N. Snavely, A. Kanazawa, A. Rostamizadeh, and A. Makadia. "An analysis of svd for deep rotation estimation". In: *Advances in Neural Information Processing Systems* 33 (2020), pp. 22554–22565.

[175] J. Liu, Z. Zou, X. Ye, X. Tan, E. Ding, F. Xu, and X. Yu. "Leaping from 2D Detection to Efficient 6DoF Object Pose Estimation". In: *ECCVW* (2020).

[176] A. Murali, A. Mousavian, C. Eppner, C. Paxton, and D. Fox. "6-dof grasping for target-driven object manipulation in clutter". In: *2020 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE. 2020, pp. 6232–6238.

[177] P. Ni, W. Zhang, X. Zhu, and Q. Cao. "Pointnet++ grasping: Learning an end-to-end spatial grasp generation algorithm from sparse point clouds". In: *2020 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE. 2020, pp. 3619–3625.

[178] Y. Qin, R. Chen, H. Zhu, M. Song, J. Xu, and H. Su. "S4g: Amodal single-view single-shot se (3) grasp detection in cluttered scenes". In: *Conference on robot learning*. PMLR. 2020, pp. 53–65.

[179] S. Song, A. Zeng, J. Lee, and T. Funkhouser. "Grasping in the Wild: Learning 6DoF Closed-Loop Grasping from Low-Cost Demonstrations". In: *Robotics and Automation Letters* (2020).

[180] **M. Sundermeyer**, M. Durner, E. Y. Puang, Z.-C. Marton, N. Vaskevicius, K. O. Arras, and R. Triebel. "Multi-path Learning for Object Pose Estimation across Domains". In: *Conference on Computer Vision and Pattern Recognition (CVPR)*. 2020.

[181] **M. Sundermeyer**, Z.-C. Marton, M. Durner, and R. Triebel. "Augmented Autoencoders: Implicit 3D Orientation Learning for 6D Object Detection". In: *International Journal of Computer Vision (IJCV)*. 2020.

[182] J. Vogel, A. Hagengruber, M. Iskandar, G. Quere, U. Leipscher, S. Bustamante, A. Dietrich, H. Höppner, D. Leidner, and A. Albu-Schäffer. "EDAN: An EMG-controlled daily assistant to help people with physical disabilities". In: *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE. 2020, pp. 4183–4190.

[183] Y. Xiang, C. Xie, A. Mousavian, and D. Fox. "Learning RGB-D Feature Embeddings for Unseen Object Instance Segmentation". In: *Conference on Robotic Learning (CORL)* (2020).

[184] W. Zhao, S. Liu, Y. Shu, and Y.-J. Liu. "Towards better generalization: Joint depth-pose learning without posenet". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2020, pp. 9151–9161.

[185] W. Zhao, S. Zhang, Z. Guan, W. Zhao, J. Peng, and J. Fan. "Learning deep network for detecting 3d object keypoints and 6d poses". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2020, pp. 14134–14142.

[186] W. Agnew, C. Xie, A. Walsman, O. Murad, Y. Wang, P. Domingos, and S. Srinivasa. "Amodal 3d reconstruction for robotic manipulation via stability and connectivity". In: *Conference on Robot Learning*. PMLR. 2021, pp. 1498–1508.

[187] R. Alghonaim and E. Johns. "Benchmarking domain randomisation for visual sim-to-real transfer". In: *2021 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE. 2021, pp. 12802–12808.

[188] W. Boerdijk, **M. Sundermeyer**, M. Durner, and R. Triebel. "What's This?– Learning to Segment Unknown Objects from Manipulation Sequences". In: *International Conference on Robotics and Automation (ICRA)*. 2021.

[189] R. Brégier. "Deep Regression on Manifolds: A 3D Rotation Case Study". In: *2021 International Conference on 3D Vision (3DV)*. 2021.

[190] M. Breyer, J. J. Chung, L. Ott, R. Siegwart, and J. Nieto. "Volumetric grasping network: Real-time 6 dof grasp detection in clutter". In: *Conference on Robot Learning*. PMLR. 2021, pp. 1602–1611.

[191] M. Danielczuk, A. Mousavian, C. Eppner, and D. Fox. "Object Rearrangement Using Learned Implicit Collision Functions". In: *2021 IEEE International Conference on Robotics and Automation (ICRA)*. 2021.

[192] G. Du, K. Wang, S. Lian, and K. Zhao. "Vision-based robotic grasping from object localization, object pose estimation to grasp estimation for parallel grippers: a review". In: *Artificial Intelligence Review* 54.3 (2021), pp. 1677–1734.

[193] M. Durner, W. Boerdijk, **M. Sundermeyer**, W. Friedl, Z.-C. Marton, and R. Triebel. "Unknown Object Segmentation from Stereo Images". In: *International Conference on Intelligent Robots and Systems (IROS)*. 2021.

[194] C. Eppner, A. Mousavian, and F. Dieter. "ACRONYM: A Large-Scale Grasp Dataset Based on Simulation". In: *IEEE International Conference on Robotics and Automation (ICRA)* (2021).

[195] Z. Ge, S. Liu, F. Wang, Z. Li, and J. Sun. "Yolox: Exceeding yolo series in 2021". In: *arXiv preprint arXiv:2107.08430* (2021).

[196] J. Reinecke, A. Dietrich, A. Shu, B. Deutschmann, and M. Hutter. "A robotic torso joint with adjustable linear spring mechanism for natural dynamic motions in a differential-elastic arrangement". In: *IEEE Robotics and Automation Letters* 7.1 (2021), pp. 9–16.

[197] I. Shugurov, S. Zakharov, and S. Ilic. "Dpodv2: Dense correspondence-based 6 dof pose estimation". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 44.11 (2021), pp. 7417–7435.

[198] **M. Sundermeyer**, A. Mousavian, R. Triebel, and D. Fox. "Contact-Graspnet: Efficient 6-DoF Grasp Generation in Cluttered Scenes". In: *International Conference on Robotics and Automation (ICRA)*. 2021.

[199] M. Tölgyessy, M. Dekan, L. Chovanec, and P. Hubinsky. "Evaluation of the Azure Kinect and Its Comparison to Kinect V1 and Kinect V2". In: *Sensors* 21.2 (2021).

[200] G. Wang, F. Manhardt, F. Tombari, and X. Ji. "GDR-Net: Geometry-Guided Direct Regression Network for Monocular 6D Object Pose Estimation". In: *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. June 2021, pp. 16611–16621.

[201] L. Yang, Y. Z. Wei, Y. He, W. Sun, Z. Huang, H. Huang, and H. Fan. "ishape: A first step towards irregular shape instance segmentation". In: *arXiv preprint arXiv:2109.15068* (2021).

[202] W. Boerdijk, M. Durner, **M. Sundermeyer**, and R. Triebel. "Towards Robust Perception of Unknown Objects in the Wild". In: *ICRA Workshop: Robotic Perception and Mapping: Emerging Techniques*. 2022.

[203] Y. Di, R. Zhang, Z. Lou, F. Manhardt, X. Ji, N. Navab, and F. Tombari. "Gpv-pose: Category-level object pose estimation via geometry-guided point-wise voting". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2022, pp. 6781–6791.

[204] J. Hasselgren, N. Hofmann, and J. Munkberg. "Shape, Light & Material Decomposition from Images using Monte Carlo Rendering and Denoising". In: *NeurIPS*. 2022.

[205] R. L. Haugaard and A. G. Buch. "SurfEmb: Dense and Continuous Correspondence Distributions for Object Pose Estimation with Learnt Surface Embeddings". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2022, pp. 6749–6758.

[206] Y. Hu, P. Fua, and M. Salzmann. "Perspective flow aggregation for data-limited 6d object pose estimation". In: *European Conference on Computer Vision*. Springer. 2022, pp. 89–106.

[207] H. Jung, S.-C. Wu, P. Ruhkamp, H. Schieber, P. Wang, G. Rizzoli, H. Zhao, S. D. Meier, D. Roth, N. Navab, et al. "HouseCat6D–A Large-Scale Multi-Modal Category Level 6D Object Pose Dataset with Household Objects in Realistic Scenarios". In: *arXiv preprint arXiv:2212.10428* (2022).

[208] L. Lipson, Z. Teed, A. Goyal, and J. Deng. "Coupled Iterative Refinement for 6D Multi-Object Pose Estimation". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2022, pp. 6728–6737.

[209] X. Liu, R. Zhang, C. Zhang, B. Fu, J. Tang, X. Liang, J. Tang, X. Cheng, Y. Zhang, G. Wang, and X. Ji. *GDRNPP*. https://github.com/shanice-l/gdrnpp_bop2022. 2022.

[210] Z. Liu, H. Mao, C.-Y. Wu, C. Feichtenhofer, T. Darrell, and S. Xie. "A ConvNet for the 2020s". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. June 2022, pp. 11976–11986.

[211] K. Mattas, G. Albano, R. Donà, M. C. Galassi, R. Suarez-Bertoa, S. Vass, and B. Ciuffo. "Driver Models for the Definition of Safety Requirements of Automated Vehicles in International Regulations. Application to Motorway Driving Conditions". In: *SSRN Electronic Journal* (Jan. 2022). DOI: 10.2139/ssrn.4025980.

[212] J. Rothe. *Combining Global and Local 6DoF Pose Estimation for Real-World Applications*. Master Thesis. 2022.

[213] I. Shugurov, F. Li, B. Busam, and S. Ilic. "Osop: A multi-stage one shot object pose estimation framework". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2022, pp. 6835–6844.

[214] M. Stoiber, **M. Sundermeyer**, and R. Triebel. "Iterative Corresponding Geometry: Fusing Region and Depth for Highly Efficient 3D Tracking of Textureless Objects". In: *Conference on Computer Vision and Pattern Recognition (CVPR)*. 2022.

[215] Y. Su, M. Saleh, T. Fetzer, J. Rambach, N. Navab, B. Busam, D. Stricker, and F. Tombari. "ZebraPose: Coarse to Fine Surface Encoding for 6DoF Object Pose Estimation". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2022, pp. 6738–6748.

[216] J. Sun, Z. Wang, S. Zhang, X. He, H. Zhao, G. Zhang, and X. Zhou. "Onepose: One-shot object pose estimation without cad models". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2022, pp. 6825–6834.

[217] S. Tyree, J. Tremblay, T. To, J. Cheng, T. Mosier, J. Smith, and S. Birchfield. "6-DoF Pose Estimation of Household Objects for Robotic Manipulation: An Accessible Dataset and Benchmark". In: *International Conference on Intelligent Robots and Systems (IROS)*. 2022.

[218] D. Winkelbauer, B. Bäuml, M. Humt, N. Thuerey, and R. Triebel. "A Two-stage Learning Architecture that Generates High-Quality Grasps for a Multi-Fingered Hand". In: *2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE. 2022, pp. 4757–4764.

[219] Y. Wu, A. Javaheri, M. Zand, and M. Greenspan. "Keypoint cascade voting for point cloud based 6DoF pose estimation". In: *2022 International Conference on 3D Vision (3DV)*. IEEE. 2022, pp. 176–186.

[220] M. Denninger, D. Winkelbauer, M. Sundermeyer, W. Boerdijk, M. W. Knauer, K. H. Strobl, M. Humt, and R. Triebel. "Blenderproc2: A procedural pipeline for photorealistic rendering". In: *Journal of Open Source Software* 8.82 (2023), p. 4901.

[221] Y. Labbé, L. Manuelli, A. Mousavian, S. Tyree, S. Birchfield, J. Tremblay, J. Carpentier, M. Aubry, D. Fox, and J. Sivic. "MegaPose: 6D Pose Estimation of Novel Objects via Render & Compare". In: *Conference on Robot Learning*. PMLR. 2023, pp. 715–725.

[222] M. Stoiber, **M. Sundermeyer**, W. Boerdijk, and R. Triebel. *"A Multi-body Tracking Framework–From Rigid Objects to Kinematic Structures"*. Submitted to *Transactions on Pattern Analysis and Machine Intelligence* (TPAMI). 2023.

[223] **M. Sundermeyer**, T. Hodaň, Y. Labbe, G. Wang, E. Brachmann, B. Drost, C. Rother, and J. Matas. "BOP Challenge 2022 on Detection, Segmentation and Pose Estimation of Specific Rigid Objects". In: *Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*. 2023.

[224] M. Ulmer, M. Durner, **M. Sundermeyer**, M. Stoiber, and R. Triebel. "6D Object Pose Estimation from Approximate 3D Models for Orbital Robotics". In: *International Conference on Intelligent Robots and Systems (IROS)*. 2023.

[225] C.-Y. Wang, A. Bochkovskiy, and H.-Y. M. Liao. "YOLOv7: Trainable bag-of-freebies sets new state-of-the-art for real-time object detectors". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2023, pp. 7464–7475.