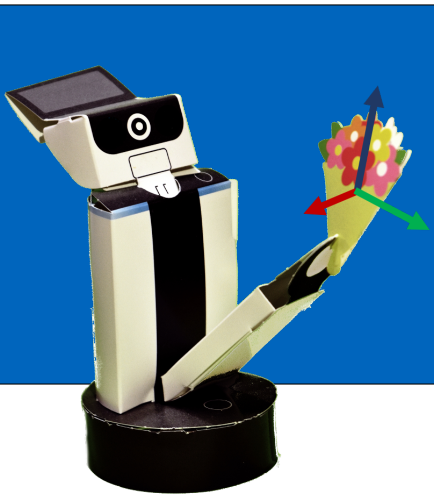
Computer Aided Medical Procedures
Prof. Dr. Nassir Navab



Dissertation

Towards Monocular 6D Object Pose Estimation

Fabian Manhardt

Fakultät für Informatik
Technische Universität München

Technische Universität München
Fakultät für Informatik

Towards Monocular 6D Object Pose Estimation

Fabian Manhardt

Vollständiger Abdruck der von der Fakultät für Informatik der Technischen Universität München zur Erlangung eines

Doktors der Naturwissenschaften (Dr. rer. nat.)

genehmigten Dissertation.

Vorsitzende(r):	Prof. Dr. - Ing. Matthias Nießner
Prüfer der Dissertation:	1. Prof. Dr. Nassir Navab
	2. Prof. Dr. Vincent Lepetit

Die Dissertation wurde am 26.05.2021 bei der Technischen Universität München eingereicht und durch die Fakultät für Informatik am 28.10.2021 angenommen.

Fabian Manhardt
Towards Monocular 6D Object Pose Estimation
Dissertation, Version 1.0

Technische Universität München
Fakultät für Informatik
Lehrstuhl für Informatikanwendungen in der Medizin
Boltzmannstraße 3
85748 and Garching bei München

Abstract

Estimating the 6D object pose is one of the most fundamental problems in Computer Vision as it is essential for various applications, including robotic grasping and autonomous driving. Thereby, the 6D object pose describes the orientation and position of the object in 3D space and is often a key step-stone for further 3D reasoning and manipulation.

Given its importance, it is not surprising that 6D pose estimation is a well researched field. Nonetheless, despite great advances, the accuracy is still not satisfactory when it comes to real applications. Moreover, there are still many open challenges which are oftentimes neglected in literature. Exemplary, many learning driven methods severely suffer from the lack of real labeled data and the very limited number of objects they can simultaneously handle. In addition, most related works used to rely on depth data in order to estimate accurate 6D pose information. While possessing depth significantly simplifies the task, it also induces several problems. First, depth sensors are usually expensive and, second, they are often not capable of obtaining depth for certain surfaces (*e.g.* glass).

Due to its high relevance for many applications, this dissertation focuses on the problem of 6D pose estimation. Nonetheless, as depth is often not provided nor completely reliable, this dissertation puts a particular emphasize on inferring the orientation and translation from monocular data alone. Since monocular 6D pose estimation is a difficult ill-posed problem, there are several open challenges. In fact, this dissertation is concerned with three different aspects for estimating the 6D pose. First, this dissertation introduces one of the first deep learning based method for fast and reliable inference and tracking/refining of the 6D object pose from RGB images. Second, several external factors which can complicate inference are investigated. Exemplary, repetitive patterns or occlusion can lead to ambiguities in pose (*i.e.* multiple poses are equivalent under perspective projection), depraving learning. Furthermore, also changes in illumination can further deteriorate performance. Third, most prior works only deal with a single object at a time, in fact, often even limiting the 6D pose space to only cover objects standing on a plane. While this allows proper reasoning at fast speed, it is an unrealistic assumption and cannot comply many applications. Therefore, generative models are paired with 6D pose estimation to enable metric reconstruction of the objects of interest.

Zusammenfassung

Die Schätzung der 6D Pose von Objekten ist eines der grundlegendsten Probleme in der Computer Vision, da sie für verschiedene Anwendungen, wie z.B. das Greifen von Robotern und autonomes Fahren, unerlässlich ist. Die 6D Pose beschreibt die Orientierung und Position des Objekts im 3D-Raum und ist oft ein wichtiger Schritt für weitere Schlussfolgerungen in 3D und für Manipulations Aufgaben von Objekten.

Angesichts ihrer Bedeutung ist es nicht verwunderlich, dass die Schätzung der 6D Pose ein gut erforschtes Gebiet ist. Trotz großer Fortschritte ist die Genauigkeit jedoch immer noch nicht zufriedenstellend, wenn es um reale Anwendungen geht. Außerdem gibt es noch viele offene Herausforderungen, die in der Literatur oft vernachlässigt werden. Zum Beispiel leiden viele lernende Methoden stark unter dem Mangel an echten annotierten Daten und der sehr begrenzten Anzahl von Objekten, die sie gleichzeitig handhaben können. Darüber hinaus sind die meisten verwandten Arbeiten auf Tiefendaten angewiesen, um genaue 6D Posen zu schätzen. Während die Verwendung von Tiefendaten die Aufgabe erheblich vereinfacht, bringt sie auch einige Probleme mit sich. Erstens sind Tiefensensoren in der Regel teuer und zweitens sind sie oft nicht in der Lage, die Tiefe für bestimmte Oberflächen (z.B. Glas) zu erfassen.

Aufgrund der hohen Relevanz für viele Anwendungen konzentriert sich diese Dissertation auf das Problem der Schätzung von der 6D Pose. Da die Tiefeninformation jedoch oft nicht zur Verfügung steht und auch nicht vollständig zuverlässig ist, wird in dieser Arbeit ein besonderer Schwerpunkt auf die Ableitung von Orientierung und Position aus rein monokularen Daten gelegt. Da die monokulare 6D Posensschätzung ein schwieriges, in der Tat ein sogar unlösbares Problem ist, gibt es mehrere offene Herausforderungen. Tatsächlich befasst sich diese Dissertation mit drei verschiedenen Aspekten für die Schätzung der 6D Pose. Erstens wird in dieser Dissertation eine der ersten Deep-Learning basierte Methode zur schnellen und zuverlässigen Inferenz und Tracking/Verfeinerung der 6D-Objektpose aus RGB-Bildern vorgestellt. Zweitens werden mehrere externe Faktoren, die die Inferenz erschweren können, untersucht. Exemplarisch können wiederholende Muster oder Verdeckungen zu Mehrdeutigkeiten in der Pose führen (*i.e.* mehreren Posen sind unter perspektivischer Projektion äquivalent), was das Lernen erschwert. Außerdem können Änderungen in der Beleuchtung die Päzision weiter verschlechtern. Drittens befassen sich die meisten früheren Arbeiten jeweils nur mit einem einzigen Objekt und beschränken den 6D-Raum oft sogar nur auf Objekte, die auf einer waagrechten Ebene stehen. Dies ermöglicht zwar eine genaue Bestimmung der Pose bei hoher Geschwindigkeit, ist aber eine unrealistische Annahme und kann vielen Anwendungen nicht gerecht werden. Daher werden generative Modelle mit der

6D Poseschätzung gepaart, um eine metrische Rekonstruktion aller präsenten Objekte zu ermöglichen.

Acknowledgments

I believe that, to go through the journey of PhD you don't need colleagues, you need friends. Fortunately, I can gladly say that I was lucky enough to have amazing friends that I could always rely on when needed. Here I would like to say to all of you "*Thanks*".

First of all, I would like to thank my supervisors Federico and Nassir – for giving me the opportunity to do my PhD at the CAMP lab, for all the very fruitful discussions, and for their constant support throughout all these years. A special thanks goes to Wadim for his supervision during my Master Thesis and beyond. I'm also deeply grateful to our secretary Martina without whom a lot of things would have gone much less smooth (in the words of Nassir *"Go to Martina. She can make things happen"*).

Next I would like to thank Sven, Luca and Kobori-san from Toyota for the great collaboration that not only allowed me to conduct my research on 6D pose, but also helped me to turn my research into actual applications. Greetings also to Adrien and all my former colleagues at TRI, who taught me so many things during my research internship in sunny California.

I also appreciate all the time with my fellow PhDs inside and especially outside of work. Comrades in PhD from the early hour on – Anees, Benni, Christian, David, Huseyin, Iro, Keisuke-san, Mai, Marco, Mira, Niki, Niko, Sergej, Shun-Cheng, Tolga and Yida. Comrades that joined later during the journey – Evin, Gu, Mahdi, Manuel and Yan-yan – Hang in there guys! And finally my roomies – Hendrik and Johanna. Sorry if I forgot anybody, but know that I appreciate you all folks!

Cheers to my friends Dominik, Emil, Jonas, Matthäus and Sebastian for their great support and constant beer supplies dating back far beyond the course of this PhD. Also cheers to the whole CAMPioni team – Federico, Jakob and Maria. Best beach volley team ever. Next time we will win that cup!

Last but not least, the biggest thanks goes to my family and Helisa, my partner in life and finally also in research (*i.e. the couple project*). Thank you so much for your love and for always having my back in the good times and, in particular, in the bad times. You turned this depressing period into a amazing adventure.

Contents

III Conclusion and Outlook

IV Appendix

Chronological List of Authored and Co-authored Publications

2021

[1] G. Wang, **F. Manhardt**, F. Tombari, and X. Ji. "GDR-Net: Geometry-Guided Direct Regression Network For Monocular 6D Object Pose Estimation". In: *IEEE Conference on Computer Vision and Pattern Recognition (CVPR).* 2021

[2] **F. Manhardt***, L. Minciullo*, K. Yoshikawa, S. Meier, F. Tombari, and N. Kobori. "DB-GAN: Boosting Object Recognition Under Strong Lighting Conditions". In: *2021 IEEE Winter Conference on Applications of Computer Vision (WACV).* 2021

2020

[3] **F. Manhardt**, G. Wang, B. Busam, M. Nickel, S. Meier, L. Minciullo, X. Ji, and N. Navab. "CPS++: Improving Class-level 6D Pose And Shape Estimation From Monocular Images With Self-Supervised Learning". In: *arXiv preprint arXiv:2003.05848v3.* 2020

[4] G. Wang*, **F. Manhardt***, J. Shao, X. Ji, N. Navab, and F. Tombari. "Self6D: Self-Supervised Monocular 6D Object Pose Estimation". In: *European Conference on Computer Vision (ECCV).* 2020 **[Oral]**

2019

[5] **F. Manhardt***, D. Arroyo*, C. Rupprecht, B. Busam, T. Birdal, N. Navab, and F. Tombari. "Explaining The Ambiguity of Object Detection And 6D Pose From Visual Data". *IEEE International Conference on Computer Vision (ICCV).* 2019

[6] **F. Manhardt***, W. Kehl*, and A. Gaidon. "ROI-10D: Monocular lifting of 2d detection to 6d pose and metric shape". In: *IEEE Conference on Computer Vision and Pattern Recognition (CVPR).* 2019

2018

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

[8] T. Hodan, F. Michel, E. Brachmann, W. Kehl, A. G. Buch, D. Kraft, B. Drost, J. Vidal, S. Ihrke, X. Zabulis, C. Sahin, **F. Manhardt**, F. Tombari, T.-K. Kim, J. Matas, C. Rother. "BOP: Benchmark For 6D Object Pose Estimation". In: *European Conference on Computer Vision (ECCV).* 2018

2017

[9] **F. Manhardt***, W. Kehl*, F. Tombari, S. Ilic, and N. Navab. "SSD-6D: Making RGB-Based 3D Detection And 6D Pose Estimation Great Again". In: *IEEE International Conference on Computer Vision (ICCV).* 2017 [**Oral**]

Part I

Introduction

1 Introduction

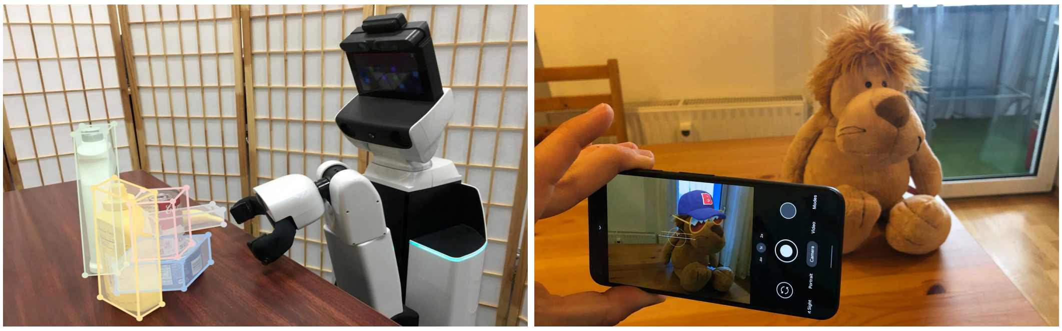


Figure 1.1. **Exemplary Applications of 6D Pose Estimation.** Left: The Toyota HSR robot is grasping the object of interest leveraging the estimated 6D object pose [10][©2019 IEEE]. Right: Coherent augmentations are applied to the lion object using the inferred 6D pose information.

1.1 Motivation and Main Objective

How much of an object can a human infer from a single RGB image? As humans we are constantly interacting with objects in the 3D world. In fact, the interactions range from grasping objects to steering cars as well as playing the guitar and much further. All these interactions, however, require a good understanding of the objects in the 3D world. In particular, we need to comprehend the 3D geometry as well as orientation and position of the object in order adequately manipulate it within the environment. Exemplary, when playing the guitar it is vital to understand its size and orientation so to play the correct strings. While driving it is crucial to be capable of estimating all 3D properties for each traffic participant to safely navigate through the streets.

Since humans possess binocular vision, they can naturally infer 3D information from the disparity of the sensed images [11]. Nonetheless, despite having stereo vision available, humans are even capable of interacting with the environment using only monocular input [12]. This can be easily demonstrated when grasping an object having one eye closed. Surprisingly, humans can accomplish this task without any efforts [13]. This can be contributed to the fact that we establish priors of the world, allowing us to estimate 3D properties despite this being an inherent ill-posed problem.

Being capable of accurately estimating these object shapes and 6D poses (*i.e.* orientation and translation), also drives many applications within other domains [14, 6, 15]. In particular, many tasks can be broken down into estimating the pose of the object and then adequately interacting with it.

Exemplary, the task of most industrial robots typically requires to grasp parts for assembling objects such as cars. Successful grasping of the object, however, relies on having a notion of the pose [16, 15]. Especially, when assembling the objects, the individual parts need to plugged together precisely, therefore, a high confidence in the pose of the object in hand is crucial. While there are methods that directly output grasping instructions for any geometry [17, 18], higher accuracy and level of interaction can be achieved when estimating the full 6D pose for a known 3D model [15, 19].

Many companies are currently aiming at developing service robots for various different use cases [20, 21, 22]. Thereby, one very important sector for service robots resides in, for instance, healthcare. Due to increasingly older societies, there is a lack of qualified specialist to take care of them. Therefore, service robots are developed which can help elderly people to accomplish basic tasks of their daily life. Toyota recently introduced the human support robot (HSR) [23], which is a omnidirectional moving robot equipped with an RGB-D camera and a gripper for manipulation (*c.f.* Figure 1.1 [left]). HSR can fulfill several diverse tasks such as tidying the room or bringing objects to the patient. Both tasks require knowledge of the pose and object in some form. Of course, there are various other disciplines in healthcare that profit from estimating the 6D pose. A prime example would be computer aided surgery, *e.g*. surgical tool detection/tracking [24, 25, 26]

Augmented reality is a relevant area which also benefits from 6D pose estimation [14]. It is a particular hot field due to the introduction of many novel head-mounted displays such as Microsoft Holelens [27] or Magic Leap [28]. Essentially, as illustrated in Figure 1.1, coherent augmentations of objects can be employed by means of the 6D pose and the associated 3D CAD model [10]. Moreover, virtual interactions with real objects can be accomplished via knowledge of the object's position and orientation.

Certainly, autonomous driving is another very important field which is highly dependant on estimating the pose of all traffic participants [29, 30]. As very precise pose estimates are required, it is still questionable if fully autonomous driving can be achieved from monocular data alone. Nevertheless, driver assistance systems oftentimes do not require this high level of precision. Hence, estimating the 6D pose allows to add new safety components by simply attaching a camera to the car [31, 32].

Similar to humans also these fields heavily rely on specific sensors which allow to capture the 3D world [33, 34]. Fairly accurate consumer depth cameras have basically become part of the standard equipment for any robot [23]. Even most mobile phones are nowadays relying on depth sensors, enabling many applications such as face identification or metric room planing [35]. Certainly, also autonomous vehicles capture 3D information by means of Lidar sensors [30, 36].

Although depth sensors enable so many applications, they also exhibit several downsides [37]. In essence, most sensors cannot retrieve measurements for very dark or bright structures. Moreover, surfaces such as glass leads to scattering of the laser for certain sensors, resulting in no depth information [38, 39]. Stereo sensors, on the other hand, can be noisy due to wrong matching of the left and right image parts and the uncertainty arising from triangulation for objects far away [40]. Finally, Lidar sensors also have difficulties measuring depth when dealing with strong snow or rain which is of course a crucial ability. Moreover, Lidar sensors only provide sparse 3D data in form of point clouds and are fairly expensive [41, 42].

Grounded on these drawbacks, it is only natural to investigate if the task of 6D pose estimation can be achieved only from monocular RGB images alone. Remember that humans are capable of approaching all these tasks only based on the information from a single eye using learned priors. Similarly, early works in depth prediction have demonstrated that also neural networks (NNs) are able to learn such priors and can estimate depth reliably from monocular images, despite being an ill-posed problem [43, 44]. Analogously, in this thesis we aim at studying the problem of 6D pose estimation from monocular input using deep learning. Thereby, we show how ideas from 2D object detection can be harnessed to also estimate the 6D pose at high inference speed. Afterwards, we focus on tackling particular challenges in 6D pose estimation, such as ambiguities due to repetitive patterns and geometry as well as extending pose estimation to deal with previously unseen objects.

1.2 Structure of this Dissertation

This section briefly outlines the structure of this dissertation.

Chapter 2: Theory and Fundamentals. Estimating the 6D object pose requires a good understanding of many fields in Computer Vision. This chapter introduces all concepts which are leveraged within this dissertation. Thereby, I briefly introduce basics in Computer Vision 2.1 and Deep Learning 2.2 as these are heavily harnessed throughout all works. Moreover, as 6D pose estimation typically requires to first localize the object in 2D image space, I will also briefly discuss current works in the field of 2D object detection in Section 2.3. In the following, I will take a look at more advanced topics, in particular, I will explain the idea of *Generative Adversarial Networks* (GANs) and *Differentiable Rendering* in Section 2.4 and 2.5, respectively.

Chapter 3: Monocular 6D Object Pose Estimation. This chapter serves to introduce relevant concepts from the field of 6D pose estimation. Essentially, I will introduce the problem statement and highlight common choices for representation of the 6D pose. Afterwards, the evaluation protocol is presented, including all employed datasets and metrics.

Chapter 4: Recent History of 6D Pose Estimation. Driven by deep learning, estimating the 6D pose from monocular data received an incredible amount of attention and the number of works within this field has increased at a vast speed. Hence, in this section, I will give an overview of the recent history after deep-learning has entered the realm. Thereby, I will first talk about works devoted to instance-level 6D pose estimation in section 4.1. Afterwards, I will discuss works for class-level 7D pose estimation in Section 4.2, before diving into the very recent field of class-level full 9D pose estimation in Section 4.3.

Chapter 5: Summary of Contributions. In this chapter the contributions of our works will be presented. Essentially, I will talk about general 6D pose estimation and the first deep-learning based approach in Section 5.1. The following sections of the chapter are devoted to individual challenges that can have a significant impact on the pose accuracy.

Chapter 6: Summary and Findings. This section summaries the contributions from chapter 5 and concludes on the findings.

Chapter 7: Future Work and Discussion. The last chapters serves to give an outlook on remaining open problems. Moreover, some first explorations to tackle these problems are presented.

Appendix. Contains abstracts of all publications which have not been discussed within the scope of this dissertation.

Theory and Fundamentals 2

2.1 Computer Vision

2.1.1 The Pinhole Camera Model

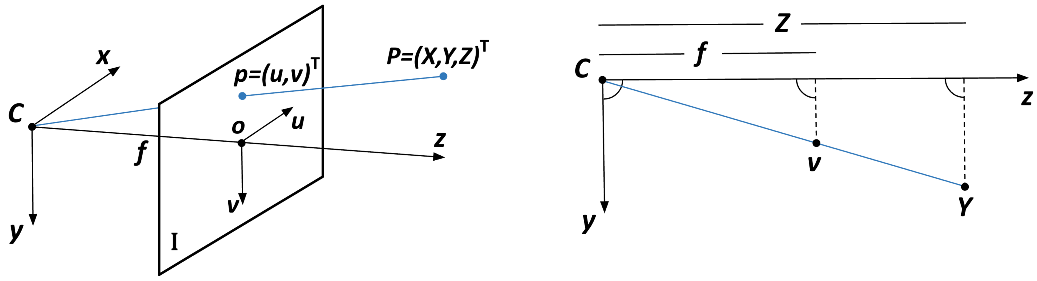


Figure 2.1. **The Pinhole Camera Model** describes the relationship between 3D points and their perspective projection on the camera's image plane.

The pinhole camera model describes the relationship between a 3D point $P = (X, Y, Z)^T$ and its associated image point $p = (u, v)^T$ on the image plane I of the camera centered at C under perspective projection [45]. Leveraging the focal length, defined as the distance from the pinhole to the image plane, the following pinhole camera equations can be established $\frac{v}{f} = \frac{Y}{Z}$ from which v can be inferred as $v = f\frac{Y}{Z}$ (*c.f.* Figure 2.1 [right]). Similarly, u can be computed according to $u = f\frac{X}{Z}$. Notice that in literature homogeneous coordinates are often employed to simplify notation. Thereby, a fourth coordinate is appended to a 3D point $P = (X, Y, Z, 1)^T$. Further, all points $P = (\lambda X, \lambda Y, \lambda Z, \lambda)^T$ with $\lambda \neq 0$ denote the same point in 3D space. All points that can be represented by such quadruples live in the projective space P^3. The actual 3D point can be simply inferred via division by the fourth coordinate $(X, Y, Z, k)^T \equiv (\frac{X}{k}, \frac{Y}{k}, \frac{Z}{k})^T$, except when $k = 0$, describing 3D points which lie at infinity. Projective spaces can thus explain points at infinity, while euclidean spaces are not able to. Using the homogeneous representation, the perspective projection turns into a linear system

$$p = \begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \begin{bmatrix} f\frac{X}{Z} \\ f\frac{Y}{Z} \\ 1 \end{bmatrix} \equiv \begin{bmatrix} fX \\ fY \\ Z \end{bmatrix} = \begin{bmatrix} f & 0 & 0 & 0 \\ 0 & f & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix}. \tag{2.1}$$

Most current image systems define the origin as the top-left corner of the image. Therefore, the points after projection are additionally shifted by the optical center $o = (o_x, o_y)^T$

$$p = \begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \begin{bmatrix} f\frac{X}{Z} + o_x \\ f\frac{Y}{Z} + o_y \\ 1 \end{bmatrix} \equiv \begin{bmatrix} fX + o_x Z \\ fY + o_y Z \\ Z \end{bmatrix} = \begin{bmatrix} f & 0 & o_x & 0 \\ 0 & f & o_y & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix} = KP. \tag{2.2}$$

The matrix K, describing the perspective projection, is commonly known as camera *intrinsics* matrix. Noteworthy, the camera commonly also possesses *extrinsics* parameters T, representing its position and orientation within the 3D world which turns $p = KP$ into $p = KTP$. However, since the object pose is estimated with respect to the camera, the camera is assumed to be always centered at the origin of the world coordinate system with $T = I$ being the identity matrix and can be hence neglected within the scope of this dissertation $p = KTP = KIP = KP$.

2.2 Deep Learning

In this section, I first want to give a brief overview about the history of deep learning. Afterwards, I will introduce the most important principles required to understand this dissertation. Finally, I will conclude by introducing the most important building blocks of the utilized neural networks.

2.2.1 Brief historical overview on Deep Learning

Due to its recent hype, deep learning seems to be an idea which has just very recently emerged. In fact, it has already existed under different terms, *i.e.* cybernetics and connectionism, since the 1940s [46]. Ever since then there have been three noticeable waves of deep learning, with the latter one still prevailing.

The first wave (cybernetics) was motivated from a neuroscientific perspective, attempting to model the brain function (*c.f.* Figure 2.2 [left]). In 1943, McCulloch and Pitts [48] introduced the artificial *neuron*, which was essentially a linear function (*c.f.* Figure 2.2 [right]). The artificial *neuron* was harnessed to recognize two different categories by discriminating if the value of the linear function $f(x, w) = \sum w_i x_i$ turns out positive or negative. Noteworthy, the weights had to be set correctly, which could be accomplished by a human operator. In 1958, Rosenblatt [47] introduced the *perceptron*, the first implementation that allowed learning of a single neuron from given samples via potentiometers. However, it soon became clear that linear models have several limitations – the most famous being that they are not capable of learning the XOR function [49]. In the following these findings led then to the first winter of AI.

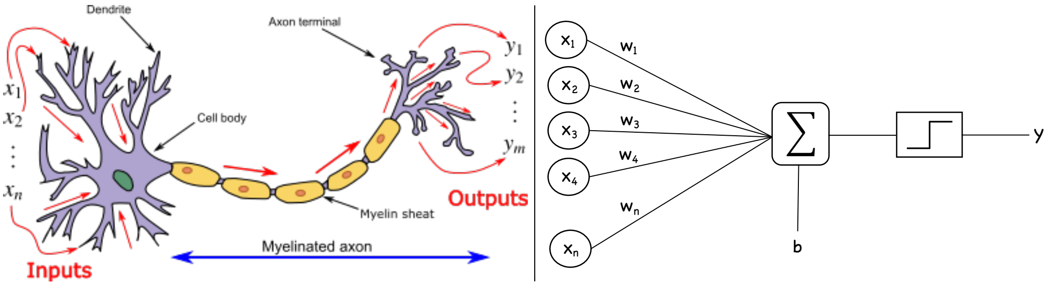


Figure 2.2. **The Neuron.** Left: An illustration of a neuron or nerve cell, the main component of the nervous system within humans. A neuron is an electrically excitable cell that communicates with other cells via specialized connections called synapses (source: Wikipedia, license: `https://creativecommons.org/licenses/by-sa/3.0/deed.en`). Right: A depiction of the rosenblatt perceptron [47]. The rosenblatt perceptron is a simple linear function, which can be optimized to learn a separation line for classification tasks. The perceptron can be considered as the artificial analogy to a neuron of the brain.

The second wave started around the 1980s from a movement know as connectionism, postulating the idea of describing a complex model via a connected network of simple computational units. Noteworthy, various concepts that are still leveraged today were introduced during the connectionism movement. One main contribution was the idea of *distributed representation* [50]. Instead of employing *role-specific* neurons for each individual target, Hinton *et al*. proposed to learn shared concepts (*i.e*. colors, objects), significantly reducing the required number of neurons. Another major contribution was the introduction of the *back-propagation* algorithm, allowing to efficiently train deep neural networks [51]. As of today, *back-propagation* is still the gold standard for training of deep learning based methods. Despite all the progress during the connectionism movement, in the mid-1990s the second winter of AI emerged as the unrealistically high ambitions of AI ventures could not be fulfilled.

Finally, the third wave of neural network research, which is still persisting, started with a break-trough in 2006. Hinton *et al*. [52] showed that deep belief networks could be efficiently trained using greedy layer-wise pre-training. Moreover, due to strong increase in computational resources, deeper networks could be successfully explored, eventually giving it the name *deep learning*. The biggest revolution in neural network research and probably even within Computer Vision as a whole, then arrived with the introduction of AlexNet [53]. AlexNet is a deep convolutional neural network (simultaneously trained on two GPUs), which was capable of surpassing all competing methods on the ImageNet challenge by a vast margin. In particular, AlexNet was able to reduce the top-5 error for image classification from 26.1% to 15.3%. Ever since then deep learning driven methods keep consistently improving, even exceeding human performance on this task [54, 55, 56, 57]. Moreover, AlexNet started a huge trend of applying neural networks to almost every discipline within Computer Vision, ranging from object detection [58, 59] over image style transfer [60] to 3D object reconstruction [61, 62] and way beyond.

2.2.2 Principles of Deep Learning

As aforementioned, the most simple network is a single-layer *perceptron*, which accumulates all the information from the input, adds a bias, and predicts the output as a weighted sum $f(x, w) = \sum w_i x_i + b_i$. Multiple outputs can be obtained by means of simply stacking a set of perceptrons (*e.g.* for two outputs $(y_1, y_2)^T = (\sum w_{i,1} x_{i,1} + b_{i,1}, \sum w_{i,2} x_{i,2} + b_{i,2})^T$). For easier notation, the matrix representation is utilized in the following with $f(x, W) = Wx + b$, where $W \in \mathbb{R}^{d_x \times d_y}$ and $b \in \mathbb{R}^{d_y}$. Since perceptrons exhibit severe limitations such as not being capable of modeling non-linear functions, in the following deeper models were proposed [51]. Yet, as any combination of linear function results again in a linear function $f(W_1, W_2, x) = W_2(W_1 x) = W_3 x$, non-linearities (also known as activation functions) $\sigma(\cdot)$ are required to gain depth via stacking of multiple layers $f(W_1, W_2, x) = W_2 \sigma(W_1(x))$.

Efficient Gradient Computation Using Backpropagation. In a supervised setup, we typically have a network consisting of parameters Θ and a set of input-output pairs (x_i, y_i) with $x_i \in \mathbb{R}^{d_x}$, $y_i \in \mathbb{R}^{d_y}$ and $i \in \{1, ..., N\}$. In the following, from this set we want to learn the mapping $f_\Theta : \mathcal{X} \to \mathcal{Y}$ from an input space $\mathcal{X}$ to the associated output space $\mathcal{Y}$. This is achieved by finding the optimal parameter Θ^* minimizing the error between the predictions $f_\Theta(x)$ and their associated ground truths y

$$\Theta^* = \underset{\Theta}{\operatorname{argmin}} \sum_i \mathcal{L}(f_\Theta(x_i), y_i). \tag{2.3}$$

Despite there are many different common loss functions for $\mathcal{L}$ existing, for the following example the Euclidean norm (l_2 loss) is employed as it is a very common function for many regression problems with $l_2(\hat{y}, y) = ||\hat{y} - y||_2$. Notice that finding an appropriate loss function is crucial to learn a good mapping, yet, as long as the loss function is differentiable with respect to Θ, the whole network can be optimized referring to *stochastic gradient descent*. Further, in this example a two-layer neural network is leveraged, *i.e.* $\Theta = \{W_1, W_2, b_1, b_2\}$ with $f_\Theta(x) = W_2 \sigma(W_1 x + b_1) + b_2$. In summary, we want to minimize the error of

$$\mathcal{L}(\Theta, x, y) = \sum_{i=1}^{N} ||W_2 \sigma(W_1 x_i + b_1) + b_2 - y_i||_2 \tag{2.4}$$

with respect to the model parameters Θ. To optimize this loss using gradient descent, we are required to calculate the gradients of the loss with respect to Θ

$$\frac{\partial \mathcal{L}}{\partial \Theta} = \frac{\partial}{\partial \Theta} \sum_{i=1}^{N} ||W_2 \sigma(W_1 x_i + b_1) + b_2 - y_i||_2 \tag{2.5}$$

The first important observation that can be made is, that the function of the neural network can be dissected into its employed layers. In other words, the mapping $f_\Theta(x) = g(h(x))$ is composed of the functions of the two underlying layers $h(x) = \sigma(W_1 x + b_1)$ and $g(x) = (W_2 x + b_2)$. The objective function can be thus rewritten as $\mathcal{L}(x, y) = l_2(g(h(x)), y)$. Let's now take the derivative of $\mathcal{L}$ with respect to W_2 and W_1

$$\frac{\partial \mathcal{L}}{\partial W_2} = \frac{\partial l_2}{\partial g} \frac{\partial g}{\partial W_2}, \qquad \frac{\partial \mathcal{L}}{\partial W_1} = \frac{\partial l_2}{\partial g} \frac{\partial g}{\partial h} \frac{\partial h}{\partial W_1}. \tag{2.6}$$

What can be observed is that parts of the computation of the derivatives of the inner layer with respect to W_1 are shared with the previous layer with respect to W_2. Similarly, we can also compute the derivatives for the bias terms b_2 and b_1

$$\frac{\partial \mathcal{L}}{\partial b_2} = \frac{\partial l_2}{\partial g}\frac{\partial g}{\partial b_2}, \qquad \frac{\partial \mathcal{L}}{\partial b_1} = \frac{\partial l_2}{\partial g}\frac{\partial g}{\partial h}\frac{\partial h}{\partial b_1}, \tag{2.7}$$

and make the same observation. In fact also the derivatives for weights and biases have shared terms. Hence, back-propagation is a simple way to compute the partial derivatives for each parameter in a very efficient manner, by avoiding to re-compute terms that have been already calculated before. Due to the nature of the chain rule, the gradients of a layer typically share terms with the gradients of its following layers, including the derivative with respect to the objective function. Therefore, during optimization we typically have two steps: a forward pass in which we compute the final loss with respect to the network output, and a backward pass in which we propagate the gradients from the last layers to the first layers. Hence, the name *back-propagation*.

Optimization Techniques. While optimization techniques harnessing second order derivatives are commonly more stable, due to the large number of parameters it is not yet feasible to employ them. In fact, computing and storing the Hessian matrix exceeds current memory limits by far. Therefore, all training methods are currently variants of gradient descent

$$\Theta^{t+1} = \Theta^{t} - \lambda\frac{\partial \mathcal{L}}{\partial \Theta}. \tag{2.8}$$

Thereby, λ refers to the step-size, also known as *learning rate*, and the parameters Θ^0 are randomly initialized. In theory, one would like to simultaneously optimize over all N samples from the dataset, however, this is typically not feasible due to memory limitations and the associated computational complexity. As consequence, *stochastic gradient descent* is instead employed. The actual step is thereby approximated using a small mini-batch of selected samples. The size of the batch is typically referred to as *batch-size* and oftentimes a very important hyper-parameter [63, 64].

Notice that barely any method directly applies stochastic gradient descent, but instead relies on improved variants. The most prominent being stochastic gradient descent with *momentum* [65]

$$g_t = \gamma g_{t-1} + \lambda\frac{\partial \mathcal{L}}{\partial \Theta} \tag{2.9}$$

$$\Theta^{t+1} = \Theta^{t} - g_t. \tag{2.10}$$

Thereby, the final gradient step g_t is a combination of the current gradients $\frac{\partial \mathcal{L}}{\partial \Theta}$ and the gradients g_{t-1} from the previous iteration, with γ being the momentum factor. Momentum serves two main functions: First, it accelerates training when subsequent gradients point towards the same direction. Second, it dampens oscillation when the optimization reaches close to the minimum. *Nesterov momentum* [66] leverages the knowledge of the anticipated gradient direction λg_{t-1} and attempts to look ahead by approximating the next position

$\Theta - \lambda g_{t-1}$. Harnessing the gradient at the probable next position can correct potential mistakes and helps stabilizing training

$$g_t = \gamma g_{t-1} + \lambda \frac{\partial \mathcal{L}(\Theta^t - \gamma g_{t-1})}{\partial \Theta} \tag{2.11}$$

$$\Theta^{t+1} = \Theta^t - g_t. \tag{2.12}$$

While these optimizers rely on a single learning rate for all parameters, adaptive optimizers, such as *Adagrad* [67], *Adadelta* [68] and *Adam* [69], leverage adaptive learning rates for each parameter. These optimizers are particularly suited for problems with sparse input data and can almost eliminate the need for hyper-tuning of the learning rate. For more information, I kindly refer the reader to the review from [70].

2.2.3 Basic Building Blocks of a Neural Network.

This section briefly introduces the most important building blocks of deep neural networks.

Fully-connected Layer. The most basic block is the multi-layer perceptron (MLP), *i.e.* a fully connected linear layer. As previously discussed, a fully connected layer computes the weighted sum of all input parameters and adds a bias to the output according to $f = Wx + b$ with $W \in \mathbb{R}^{d_x \times d_y}$ and $b \in \mathbb{R}^{d_y}$.

Convolutional Layer. The convolutional layer is the core layer of any Convolutional Neural Network (CNN) [71]. Since fully-connected layers take each individual input into account, it can soon become very expensive when dealing with high dimensional input data. Moreover, many input modalities such as RGB images are locally structured. This local structures allows to share parameters within local neighborhoods, enabling to save a lot of parameters. This can be achieved via convolutions.

A convolution is a set of N local filters with $W_i \in \mathbb{R}^{d_x \times m \times n}$ and $i \in \{1, ..., N\}$, denoting the parameters of the convolutional layer. Thereby, d_x is the depth of the input volume and $m \times n$ corresponds to the window size of the employed filter. During a forward pass, each filter is then convolved across the width and height of the input volume, computing the dot product between the entries of the filter and the input volume in order to produce a 2-dimensional activation map. Finally, by stacking all N activation maps onto each other, one can receive the output volume for this layer. Noteworthy, a first variant of a convolutional neural network dates back to 1980, when Fukushima [72] introduced the *Neocognitron*.

Pooling Layer. Pooling layers [71] are responsible for down-sampling the input volume. Since the locations of the key points are generally not very relevant, it is often sufficient to only maintain the relative locations. Thus, pooling layer can progressively reduce the spatial size of the output volume, leading too less parameters and faster convergence. Thereby, each feature map is individually convolved with a filter of size $m \times n$, and only the strongest (max-pooling) or the average (mean-pooling) response within the filter is forwarded. Notice that since $\mathrm{max}(\cdot)$ is a non-differentiable operation, the gradient is simply set to flow through the input having the maximum value.

Feature Normalization. A core problem of CNNs resides in the *vanishing gradient* problem. Due to the chain rule, the repetitive multiplication of mostly small terms (*i.e.* $\nabla_{ij} <= 1$) leads to increasingly smaller step sizes for earlier layers. Feature normalization is a way to tackle this problem. Thereby, the gradient is kept at a stable range by means of normalizing the output of the preceding layer to possess zero mean and unit standard deviation. In addition, the authors of [63] claim that it also helps to mitigate the *covariate shift*. Without normalization, subsequent layers have to anticipate the output of the previous layer in order to properly process the input. Hence, normalizing the output allows layers to learn independently.

The first normalization technique, known as *Local Response Normalization* (LRN), was introduced in AlexNet [53]. Nevertheless, Batch Normalization (BatchNorm) is the currently most used technique [63]. BatchNorm computes moments through the batch dimension, which are then leveraged to track an exponential moving average and an exponential moving standard deviation. Noteworthy, as shown in [64], BatchNorm does not work well in a low batch-size regime. To this end, Wu and He [64] propose GroupNorm, computing the mean and standard deviation through the channels dimension (*i.e.* the number of feature maps) in groups of a given size k.

Activation Functions. As aforementioned, activation functions σ are usually applied after each fully connected or convolutional layer and are required to employ deeper networks. Common choices for $\sigma(\cdot)$ involve "S"-shape functions such as sigmoid $\sigma(x) = \frac{1}{1+\exp^{-x}}$ and tanh $\sigma(x) = \frac{\exp^{x} - \exp^{-x}}{\exp^{x} + \exp^{-x}}$ or a variant of the Rectifier linear unit (ReLu) $\sigma(x) = \mathrm{max}(x, 0)$. Since ReLu is not only faster to compute, but partially also helps to mitigate the *vanishing gradient* problem, almost all current networks are relying on a variant of it. Nevertheless, as any negative output is mapped to zero, weight updates can occasionally cause a neuron to never activate again. Further, since the gradient is always 0 for any $x < 0$, this state cannot be left anymore (*dying Relu* problem). Hence, most methods employ variants of ReLu such as ELu or LeakyReLu, which allow small gradients for negative values.

Residual Connections. Residual connections are another important technique to tackle the vanishing gradient problem [56]. Thereby, instead of directly learning a mapping $H = f(x)$, He *et al.* propose to learn the residual mapping $H = x + f(x)$. On the one hand, this optimizes the gradient flow as gradients can take a shortcut through the skip connection around it, and, on the other hand, the authors claim that learning of a small residual $f(x)$ is an easier task than directly learning the whole mapping H. In fact, in their work the authors were capable of training extremely deep networks with more than 1000 layers. In this dissertation, we employ ResNets having around $10 - 50$ layers.

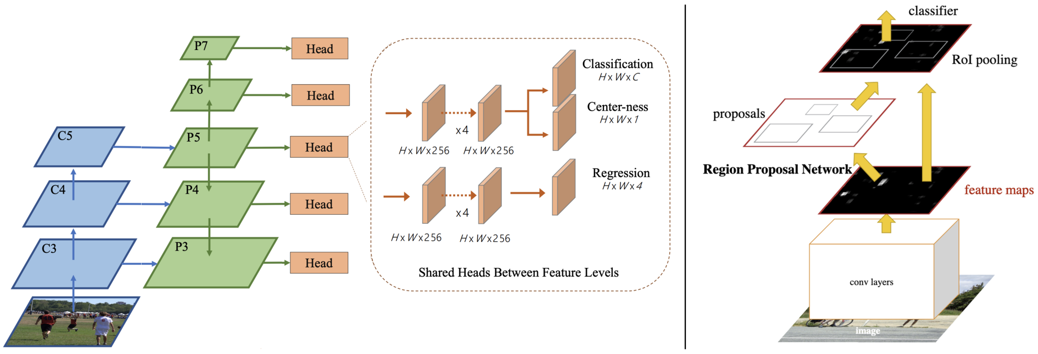


Figure 2.3. **2D Object Detection.** While single-stage methods, such as FCOS [73] [©2019 IEEE], directly output the 2D object detections [left], two-stage methods, such as Faster R-CNN [74] [©2017 IEEE], first compute region proposals which are subsequently scored by a classifier [right].

2.3 Localizing Objects 2D Image Space

Localizing the object in 2D is oftentimes the first step towards 6D object pose estimation, and can be achieved in many different ways. The most important lines of works can be separated into two categories. In particular, into single-stage [59, 73, 75] and two-stage [74, 76] detectors. While two-stage approaches first produce candidate boxes which are then scored by a second learning-based method, single-stage methods instead directly return the final bounding box together with associated object ID. Notice that single-stage detectors can be further subdivided into anchor-based [58] and anchor-free detectors [77, 78, 73, 79]. An example for a single-shot [59] and a two-stage detector [74] is depicted in Figure 2.3.

For two-stage methods, the proposal generation can be either grounded on classical methodologies such as Exhaustive Search [80, 81] or generated by another deep network [74, 76]. When leveraging a neural network for region proposal generation, multiple anchor boxes with different aspect ratios are commonly distributed at different levels of the network to cover objects at different scales. Boxes having a high IoU overlap (*e.g.* IoU > 0.7) with any ground truth are then considered as positive anchors, while boxes having a low IoU overlap (*e.g.* IoU < 0.3) are defined as negatives. The region proposal network is trained via minimizing a variant of the following loss

$$\mathcal{L}(\hat{p}_i, \hat{t}_i, \bar{p}_i, \bar{t}_i) = \frac{1}{N_{cls}} \sum_i L_{cls}(\hat{p}_i, \bar{p}_i) + \lambda \frac{1}{N_{reg}} \sum L_{reg}(\hat{t}_i, \bar{t}_i). \qquad (2.13)$$

Thereby, $\mathcal{L}_{cls}$ measures if any object is present in the associated anchor box i with $\hat{p}_i$ and $\bar{p}_i$ denoting the predicted and ground truth label of i (*i.e.* $\bar{p}_i = 1$ if an object is present in i, otherwise $\bar{p}_i = 0$). Further, for all N_{reg} assigned anchors ($\bar{p}_i = 1$), $\mathcal{L}_{reg}$ measures the offset from anchor box to the tight bounding box with $\hat{t}_i$ and $\bar{t}_i$ being the predicted and ground truth box offsets. Commont choices for $\mathcal{L}_{cls}$ and $\mathcal{L}_{reg}$ are the binary cross-entropy loss and the l_1-loss, respectively. The outgoing refined proposals are then cropped using RoI-Pooling [74] or RoI-Align [76] and forwarded to a classifier in order to retrieve the associated class for the detected object.

While two-stage methods tend to be more accurate, they are also significantly slower. Exemplary, with current hardware Faster R-CNN [74] runs at approximately 10 frames per second. In contrast, single-stage methods such as Single-Shot MultiBox Detector (SSD) [59] can often achieve real-time performance. As a major difference, each anchor usually directly infers the class of the encapsulated object ($\bar{p}_i = \mathtt{class\ id}$) , instead of only predicting if an object is present. The loss is, thus, a combination of the losses from Faster R-CNN with a classification loss (*i.e.* cross-entropy loss $\mathcal{L}_{ce}(\hat{p}, \bar{p}) = \frac{1}{N}\sum_i \bar{p}_i \log(\hat{p}_i))$ for $\mathcal{L}_{cls}$ to simultaneously score and label each anchor box. Since all these methods place anchor boxes at different levels of the network, the corresponding feature maps are representing different aspects of the image. In fact, while early anchor boxes rely on low-level features, later bounding boxes only have access to high-level features. As consequence, RetinaNet proposes to leverage Feature Pyramid Networks (FPNs) as backbone, fusing low- and high-level features before scoring of anchor boxes [75]. In addition, Lin *et al*. also propose the focal loss as scoring function. The focal loss basically re-weights the cross-entropy loss to tackle the class imbalance problem in object detection by down-weighting contributions of very confident anchors

$$\mathcal{L}_{focal}(\hat{p}, \bar{p}) = \frac{1}{N}\sum_i \bar{p}_i (1 - \hat{p}_i)^{\gamma} \log(\hat{p}_i). \tag{2.14}$$

Thereby, γ is a hyper-parameter controlling how strongly confident predictions are down-weighted. The authors empirically propose to use $\gamma = 2$. Noteworthy, while also being grounded on anchor boxes, Yolo [58] further segments the image into 7 × 7 superpixels from which it infers the class of each refined and scored anchor box.

To densely cover the whole image, a lot of anchor boxes are required, which is computationally costly. In addition, even when leveraging many anchors at different scales, it is not possible to cover all possible arrangements, thus, the detectors can exhibit blind spots. This is not the case for anchor-free approaches. For example, *CornerNet* regresses the top-left and bottom-right corners of the tight bounding box [77]. Thereby, for each pixel, Law *et al*. classify if a corner is present using a variant of the focal loss. To retrieve the final bounding box, these corners are eventually aggregated by means of feature descriptor matching harnessing metric learning. *CenterNet* extends *CornerNet* to also consider the center of the bounding box and enforcing additional constraints [78]. Finally, *FCOS* assigns the object class to each pixel in the image, based on the encapsulating ground truth bounding box. In the following, *FCOS* predicts the offset of each assigned center pixels to the left, right, bottom and top edge of the ground truth bounding box. Since pixels far from the center seem to produce less precise bounding boxes, Tian *et al*. additionally attempt to suppress predictions based on how far they are from the center of the box [73].

2.4 Generative Adversarial Networks

Generative Adversarial Networks (GANs) [82] (*c.f.* Figure 2.4) have been recently adopted to generate plausible results for a variety of Computer Vision tasks, including image inpainting [83, 84], image editing [85], style transfer [86, 87, 88], super-resolution [89] and 3D object

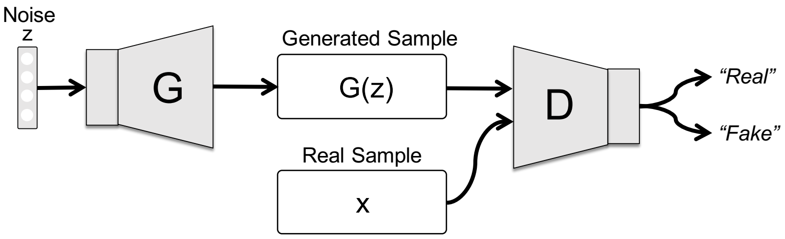


Figure 2.4. **Generative Adversarial Networks (GAN).** A GAN is composed of two individual networks, a generator G and a discriminator D. Thereby, the generator aims to generate samples from given noise which cannot by distinguished from real data by D. In contrast, D attempts to discriminate if the sample originates from the real domain or was generated by G. Training both networks with these adversarial objectives converges in G generating data which cannot be distinguished from real data anymore.

generation [90]. In essence, a GAN is composed of two networks, a generator G and a discriminator D, which are trained with conflicting objectives. In particular, G is fed with a noise vector z sampled from a gaussian distribution $z \sim \mathcal{N}(\mu, \sigma)$ and attempts to generate data, which is indistinguishable from real data of a target domain $\mathcal{X}$. On the contrary, D tries to discriminate if a given sample is taken from the real data distribution $x \in \mathcal{X}$ or generated by the generator $G(z) \notin \mathcal{X}$. Inspired by game theory, these two networks are then trained in a min-max fashion, in which both networks keep improving at their respective task, *i.e.* while D improves at determining the origin of the data, G improves at generating data which cannot be differentiated from real data by D. Eventually, this optimization ends in an equilibrium in which G produces samples that cannot be distinguished from real data of $\mathcal{X}$ anymore

$$G^* = \underset{G}{\operatorname{argmin}} \max_{D} \mathbb{E}_x[\log D(x)] + \mathbb{E}_z[\log(1 - D(G(z)))]. \tag{2.15}$$

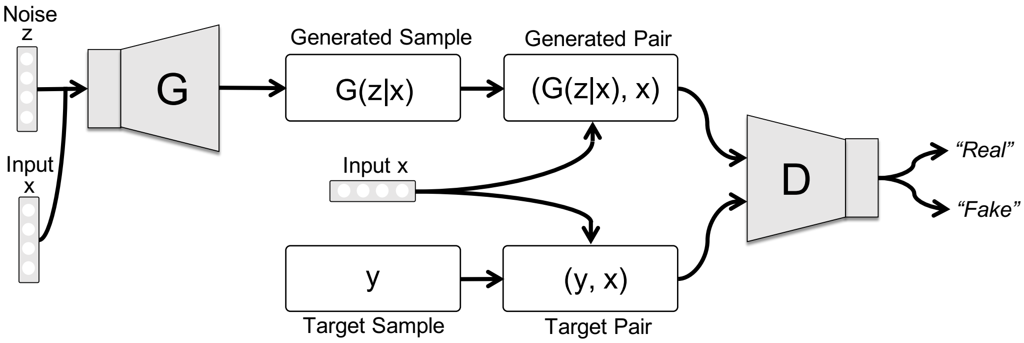


Figure 2.5. **Conditional Generative Adversarial Networks.** As for Conditional GANs, the generator is conditioned on a sample x and enforced to learn a mapping that transfers x from the source domain $\mathcal{X}$ to the target domain $\mathcal{Y}$. On the other hand, the discriminator ensures that the sample is indistinguishable from a real sample in $\mathcal{Y}$ and an appropriate translation of x to $\mathcal{Y}$.

While Goodfellow *et al.* [82] have shown that GANs are capable of generating new data from a distribution $\mathcal{X}$, GANs have been also very successfully applied to domain transfer tasks, mapping from source domain $\mathcal{X}$ to the target domain $\mathcal{Y}$ [87]. Thereby, the GAN is typically conditioned on priors $x \in \mathcal{X}$ and has to generate a sample which properly translates x to $\mathcal{Y}$ [91, 92, 87]. On the other hand, the discriminator has to discriminate if the sample is a real sample from $\mathcal{Y}$ and a correct counterpart to x. In this formulation, the optimization can be expressed as

$$\mathcal{L}_{\mathrm{cGAN}} = \mathbb{E}_{x,y}[\log D(x,y)] + \mathbb{E}_{x,z}[\log(1 - D(x, G(x,z)))]. \tag{2.16}$$

Thereby, $x \in \mathcal{X}$ represents the conditioned input, $y \in \mathcal{Y}$ the associated target sample, and z is again a sampled noise vector which is supposed to capture the nature of the one-to-many mapping from x to $\mathcal{Y}$. Notice that this is a particular strength of Conditional GANs (cGAN) over other domain transfer methods leveraging a reconstruction loss, as the discriminator D does only evaluate correct domain transfer and is therefore agnostic to ambiguities (*e.g.* when colorizing a car blue instead of red the reconstruction loss would be high despite correct transfer). An illustration of cGANs can be also found in Figure 2.5.

Isola *et al.* [87] propose a general–purpose Conditional GAN (cGAN) that can be used for a variety of image translation tasks, such as labels to street image, black and white to color photography, edges to photo, incomplete to full image etc. While the authors acknowledge the advantage of cGANs being agnostic to ambiguities, their results have demonstrated that a combination of l_1-loss for reconstruction together with a GAN-loss leads to the best results

$$G^* = \underset{G}{\arg\min}\, \underset{D}{\max}\, \mathcal{L}_{\mathrm{cGAN}} + \lambda||(y - G(x,z))||_1, \tag{2.17}$$

with λ being a weighting factor. Notice that many different priors have been utilized in literature, including lower resolution images [89], normal maps [93], incomplete image [83], text [94], labels [95], and illumination variations [2].

2.5 Differentiable Rendering

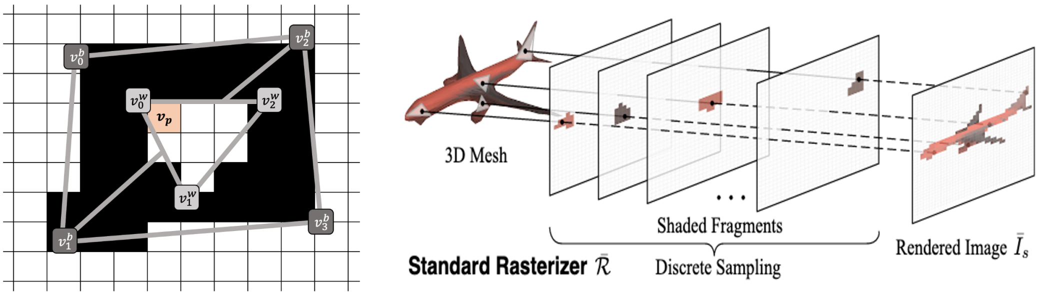


Figure 2.6. **Differentiable Rendering.** Right: Computing analytical gradients is oftentimes not useful due to the rasterization step, as rasterization simply assigns the color of the closest triangle for each pixel [©2019 IEEE]. Left: As the white triangle is the closest towards the camera, the color c_{v_p} of pixel v_p will be simply set to 1, even when using barycentric coordinates. Since the assigned color c_{v_p} is a constant, the derivative of the pixel color towards any vertex or model parameter is always zero.

In 3D Computer Vision, one typically aims at inferring 3D properties from a 2D image or scene. Thereby, supervision generally requires some form of knowledge about the 3D scene. Nevertheless, acquiring appropriate annotations is a difficult and time-consuming task. Interestingly, there exist a vast amount of large and labeled datasets of RGB images [53, 96]. Being capable of supervising 3D properties by means of RGB images is, thus, very intriguing as it enables many potential applications [97, 98]. Notice that perceiving the 3D scene onto the 2D image plane is known as *rendering* and has been extensively studied in Computer Graphics [99]. Consequently, when backpropagating the error through the rendering function, reasoning about 3D information could be theoretically learned from 2D images alone.

A rendering function $\mathcal{R}$ commonly takes as input a 3D model $\mathcal{M}$, camera parameters $\mathcal{C}$, material parameters M, and lighting parameters L to output a color image I, and/or depth image D. While there are different representations for shape such as voxels [100], pointclouds [101], or implicit functions [61], in this dissertation I focus on 3D meshes ($\mathcal{M} = (V, E)$ consisting of the model vertices V and the model triangles E) as 6D pose estimation commonly assumes the presence of a 3D CAD model.

For most traditional rendering pipelines it is difficult to utilize analytical gradients due to the rasterization step, as rasterization simply assigns the color of the closest triangle to each pixel (*c.f.* Figure 2.6 [right]) [102, 103]. Exemplary, in Figure 2.6 [left] there are two black triangles $t_1^b = \{v_0^b, v_1^b, v_2^b\}$ and $t_2^b = \{v_1^b, v_2^b, v_3^b\}$, and one white triangle $t_1^w = \{v_0^w, v_1^w, v_2^w\}$. As the white triangle is located closer towards the camera, the color c_{v_p} of pixel v_p will be simply set to 1, even when using barycentric coordinates $c_{v_p} = \sum w_i v_i^w = 1$ with $\sum w_i = 1$. Therefore, since the assigned color c_{v_p} is a constant, the derivative of the pixel color towards any model parameter $\Psi = \{\mathcal{M}, \mathcal{C}, M, L\}$ is always $\frac{\partial c_{v_p}}{\partial \Psi_i} = 0$. In addition, if v_p lies outside of any triangle, the gradients amount again to 0.

As consequence, a few works have recently been proposed to circumvent the hard assignment in order to re-establish the gradient flow [103]. Notice that, while the gradient with respect to each vertex is 0, moving v_0^w to the right will eventually change the color of v_p to black. Hence, altering vertices can impact the assigned color of pixels. Using this knowledge, one can compute gradients by allowing neighboring triangles to affect the color of v_p. In fact, there are two common strategies for re-establishing the gradient flow based on this idea. On the one hand, some works aim at approximating a *"useful"* gradient [104, 105], whereas others instead approximate the rendering itself [106, 107]. Notice that both directions have their advantages as well as disadvantages. For instance, while the latter works are capable of leveraging analytical gradients, this also comes with a cost in image quality.

As for approximating gradients, one of the first works, named *OpenDr* by Loper and Black [104], utilize approximated spatial gradients. Leveraging differential filters (such as the Sobel filter), the derivative for pixel v_p can be computed as $\frac{\partial c_{v_p}}{\partial v_p} = (\frac{\partial c_{v_p}}{\partial x}, \frac{\partial c_{v_p}}{\partial y})^T$. Hence, since background triangles also contribute to the gradient at v_p, the gradient does not necessarily amount to 0. Unfortunately, these gradients are only applied locally and not tailored towards the objective function of the given task. To deal with this dilemma, Kato *et al.* [105] propose to model the gradient for c_{v_p} by the potential change a vertex v_i can prompt to the pixel. Assuming the intensity of c_{v_p} changes from $c_{v_p}^0$ to $c_{v_p}^1$ when moving the x-coordinate x_i of v_i along the x-axis from x_i^0 to x_i^1. Thereby, x_i^0 is the current location and x_i^1 describes the position where the edge of the associated triangle collides with v_p, thus, inducing an intensity change. Since this a sudden intensity change, the gradient is not defined at this location. To this end, Kato *et al*. replace the sudden with a gradual change, approximating $\frac{\partial c_{v_p}}{\partial x_i}$. Thereby, $\frac{\partial c_{v_p}}{\partial x_i} = \frac{\delta^{c_{v_p}}}{\delta^{x_i}}$ between x_i^0 and x_i^1 with $\delta^{c_{v_p}} = c_{v_p}^1 - c_{v_p}^0$ and $\delta^{x_i} = x_i^1 - x_i^0$.

Since [104, 105] leverage standard rendering pipelines, they cannot let gradients flow into occluded triangles to properly optimize the z-component of these triangles. Therefore, the authors of *SoftRas* conduct rendering by aggregating the probabilistic contributions $c_i = \sum_j w_j^i c_j^i + w_b^i c_b$ of each mesh triangle with respect to the rendered pixels [106]. Thereby, c_b controls the background color. The weights w_j satisfy $\sum_j w_j^i + w_b = 1$ following the softmax operator according to $w_j^i = \frac{D_j^i \exp(z_j^i/\gamma)}{\sum_k D_k^i \exp(z_k^i/\gamma) + \exp(\epsilon/\gamma)}$. Thereby, z_j^i denotes the normalized inverse depth of the 3D point on the triangle e_i whose 2D projection is p_i, ϵ denotes a small constant allowing the usage of a background color and γ controls the sharpness of the aggregate function. In addition, D_j^i models the influence of the triangle e_j on p_i, with $D_j^i = \mathtt{sigmoid}(\delta_j^i \cdot \frac{d^2(i,j)}{\kappa})$. Here κ controls again the sharpness, d^2 denotes the euclidean distance and δ_j^i is a indicator function signalizing if p_i is covered ($\delta_j^i = 1$) by e_i or not ($\delta_j^i = -1$). To summarize, the closer the triangle e_j is located towards p_i the larger its contribution to the output color c_i. Remember that the color c_i is thus only an approximation of the real color at pixel i. Nevertheless, the gradient of p_i flows into every triangle and, thus, also flows into every vertex. *DIB-R* builds on top of *SoftRas* by considering foreground and background pixels independently [107]. For foreground pixels (*i.e.* all pixels that are covered by at least one face), *DIB-R* simply computes the color as barycentric interpolation of the closest triangle in an effort to avoid blurry outputs. In contrast, for background pixels *DIB-R* leverages a distance-based aggregation similar to [106]. In our work called Self6D, we further adjust *DIB-R* to additionally render the associated depth map. Since each depth value is calculated as a weighted sum of depth values from the vertices of the closest triangle, we can simply compute the analytic gradient without any approximation [4].

Noteworthy, similar ideas for optimization through rendering have also been proposed for voxels [108, 109], pointclouds [110, 111] and implicit functions [112, 113]. Moreover, while differentiable rendering resorts to well established principles from Computer Graphics, another line of works known as *neural rendering* instead aims at learning the whole rendering process via convolutional neural networks. These networks can then be leveraged to retrieve gradients via backpropagation [114].

Part II

Methodology

Monocular 6D Object Pose Estimation 3

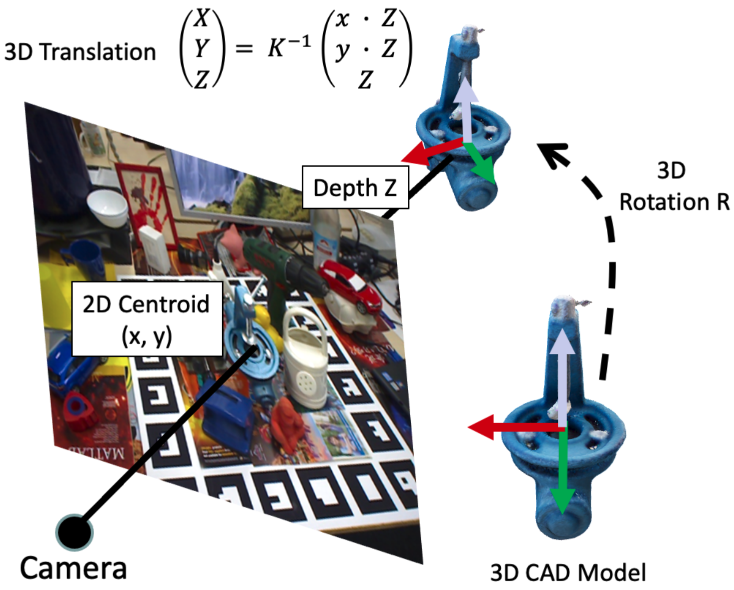

3D Translation $\begin{pmatrix} X \\ Y \\ Z \end{pmatrix} = K^{-1} \begin{pmatrix} x \cdot Z \\ y \cdot Z \\ Z \end{pmatrix}$
Depth Z
3D
Rotation R
2D Centroid
(x, y)
Camera
3D CAD Model

Figure 3.1. **Monocular 6D Object Pose Estimation.** Monocular object pose estimation aims at estimating all six degrees-of-freedom of an object $\mathcal{M}$, required to transform the object from object to the camera space. The 6D object pose consists of the object's 3D orientation R and translation t.

3.1 Problem Definition

Monocular object pose estimation describes the task of estimating the 3D rotation $R = (\phi_X, \phi_Y, \phi_Z)$, denoting yaw, pitch and role, and 3D translation $t = (t_x, t_y, t_z)$ from a single RGB image, transforming the detected object $\mathcal{M}$ from object space to the camera space. An illustration of the 6D object pose is provided in Figure 3.1. The object $\mathcal{M}$ is typically represented by 3D CAD model, consisting of 3D vertices $V = \{v_1, ..., v_N\}$, with $v_i \in \mathbb{R}^3$ and $V \in \mathbb{R}^{3\times N}$, and triangles $E = \{e_1, ..., e_M\}$ with $e_i \in \mathbb{R}^3$ and with $E \in \mathbb{R}^{3\times M}$ connecting the vertices. Further, in the multi-object scenario we want to detect and estimate the pose of all objects $\mathcal{M}_i$ from a set of N known objects $\mathcal{O} = \{\mathcal{M}_1, ..., \mathcal{M}_N\}$, being currently present in the image I. As this requires to estimate six parameters (or degrees-of-freedom) for each detection, *i.e.* 3 for translation and 3 for rotation, this task is also often referred to 6 DoF/6D object pose estimation. Estimating the 6D pose is particularly challenging as it requires the inference of 3D information from 2D data. Moreover, inferring 3D information from 2D data is an ill-posed problem, as information is lost due to the nature of the perspective projection (*c.f.* Section 2.1.1). While there is no mathematical solution to the under-constrained problem, the possession of the CAD model helps disambiguating the pose to some extent. Nevertheless, due to symmetries, repetitive pattern as well as occlusion, estimating the 6D pose can still be highly ambiguous [5]. Deep learning has recently demonstrated to be particularly suited for ill-posed problems such as monocular depth estimation, as they are able to exploit learned priors to infer absolute scales [43, 44]. Grounded on this success, we present new methodologies for estimating the 6D object pose using neural networks in Chapter 5.

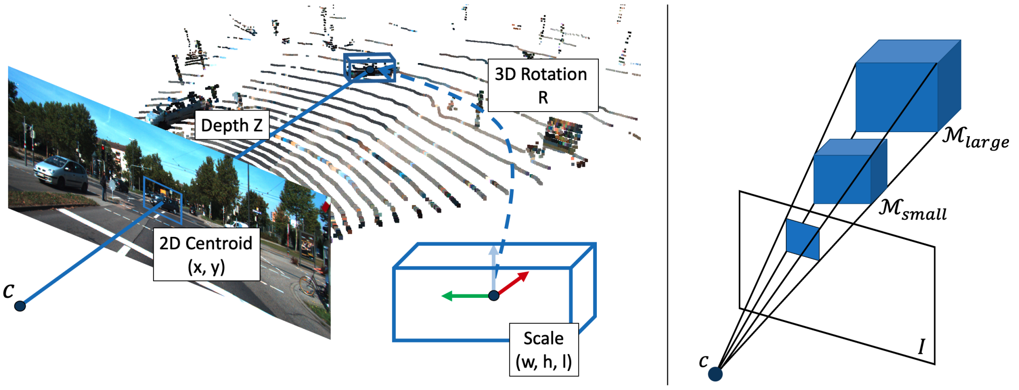


Figure 3.2. **Model-free 3D Object Detection.** Left: As no 3D CAD model is provided, model-free 3D object detection attempts to estimate a 3D bounding box tightly encapsulating the object of interest. Right: Computing a tight 3D bounding box from monocular data alone is even more challenging due to the presence of the scale-distance ambiguity. As illustrated the same object at different scales and distances can lead to the same image after perspective projection.

In many real life scenarios it is not really tractable to conduct instance-level pose estimation as the number of objects is simply too large. Exemplary, in autonomous driving there are thousands of different cars in the real-world. Also household robots that can only cope with a small number of particular objects are not very applicable, as each household typically has their own set of objects. Therefore, being capable of dealing with previously unseen objects is another very important aspect of 6D pose estimation. As this a very challenging problem, these objects are commonly instances of a known class, which allows to leverage and learn priors about object and pose. Exemplary, in autonomous driving most works estimate the size of cars as their deviation from the mean size [115, 6]. Further, notice that unseen in this scenario means that this particular sample has not been seen before, even though the network might have observed a different instance having the same or similar attributes. As illustrated in Figure 3.2 [left], in model-free pose estimation, the pose is usually parameterized by the 3D bounding box $\mathcal{B} \in \mathbb{R}^{3\times8}$, which tightly encapsulates the object of interest [116, 117]. Noteworthy, many fields, such as autonomous driving, assume the object to be always resting on the ground plane, reducing the degrees-of-freedom for rotation to 1 angle around the object's Y axis [118, 115, 119]. Nonetheless, this task is significantly more difficult, as one still needs to compute 3 additional parameters for the 3D scale $s = (w, h, l)$ of the object. Moreover, model-free pose estimation is also highly ambiguous. Let's assume we have an object $\mathcal{M}_i$ in two different scales, *i.e.* large $\mathcal{M}_{large}$ and small $\mathcal{M}_{small}$. The perspective projection on the image plane for $\mathcal{M}_{large}$ far away from the camera can be identical to $\mathcal{M}_{small}$ close by (*c.f.* Figure 3.2 [right]). Notice that as these requires to estimate in total 7 parameters, it is also often referred to as 7DoF/7D pose estimation. On another note, while for some tasks estimating the 3D bounding box is sufficient, for other tasks such as object manipulation a full 3D representation is crucial. Hence, a few methods even go beyond the 3D bounding box scenario and additionally estimate the object shape, increasing again the degrees-of-freedom [6, 3].

3.2 Representing the 6D Pose

The 3D translation is commonly represented by 3 scalars for the object's position along the X,Y, and Z axis of the camera. Direct regression of these values in 3D space is in practice, however, not very favorable due to the perspective projection. Therefore, several works instead represent the translation as the 2D projection $c = (c_x, c_y)$ of the object's 3D centroid and its distance z towards the image plane [9, 120]. While z is not given from a single image, there are several ways to recover it including RGB-D sensors or deep learning [44, 121]. Using projective geometry the 3D translation can be then inferred according to $t = K^{-1} z (c_x, c_y, 1)^T$ (*c.f.* Section 2.1.1). This significantly simplifies the problem, as two parameters can be computed in 2D space [5, 120]. Finally, moving the object from object to camera coordinate system can by achieved by simply adding t to the object vertices V

$$V' = V + t. \tag{3.1}$$

As aforementioned, the 3D rotation also has 3D degrees-of-freedom for rotating the object around its X,Y, and Z axis. Thereby, rotating the vertices V of $\mathcal{M}$ by ϕ around its X axis can be, for instance, achieved by multiplying V with the rotation matrix R_x according to

$$V' = R_x(\phi)V \quad \text{with} \quad R_x(\phi) = \begin{pmatrix} 1 & 0 & 0 \\ 0 & \cos(\phi) & -\sin(\phi) \\ 0 & \sin(\phi) & \cos(\phi) \end{pmatrix}. \tag{3.2}$$

Similarly, the matrices R_y and R_z rotate $\mathcal{M}$ around the Y and Z axis, respectively with

$$R_y(\phi) = \begin{pmatrix} \cos(\phi) & 0 & \sin(\phi) \\ 0 & 1 & 0 \\ -\sin(\phi) & 0 & \cos(\phi) \end{pmatrix} \quad \text{and} \quad R_z(\phi) = \begin{pmatrix} \cos(\phi) & -\sin(\phi) & 0 \\ \sin(\phi) & \cos(\phi) & 0 \\ 0 & 0 & 1 \end{pmatrix}. \tag{3.3}$$

Notice that applying R_x and R_y back to back $V' = R_y(\phi_y)R_x(\phi_x)V$, rotates $\mathcal{M}$ first by ϕ_x around its X axis and afterwards by ϕ_y around its Y axis. Consequently, the rotation matrix R which describes the full rotation in 3D space can be obtained as a product of individual rotations $R = R_z R_y R_x$. Further, every series of rotations can be also described by a single rotation matrix, *e.g.* $R_c V = R_3 R_2 R_1 V$ with $R_c = R_3 R_2 R_1$. It is important to understand that the coordinate system of $\mathcal{M}$ is rotated together with the object when applying R_x. Therefore, changing the order in which the rotation matrices are applied, also changes the out-coming rotation (*i.e.* in most cases $R_1 R_2 \neq R_2 R_1$). In other words, rotations are non-commutative operations. The most standard order for the individual rotations first rotates around X, then Y, and finally Z ($R = R_z R_y R_x$). All possible rotations of the 3-dimensional Euclidean space ($\mathbb{R}^3$) form a natural manifold known as special orthogonal group $SO(3)$, whose columns form a basis in $\mathbb{R}^3$. In particular, these orthogonal matrices are denoted as special as they have a determinant of 1 and thus preserve the orientation of the space.

It is worth mentioning that there are several different ways to represent the 3D rotation of an object and minds diverge a lot when it comes to choosing the right one. Rotation matrices are thereby not a very common choice since the same rotation can be achieved via different combinations of Euler angles (ϕ_x, ϕ_y, ϕ_z). Additionally, Euler angles also suffer from gimbal lock connoting that two or more rotation axes collapse. To this end, a lot of works utilize unit quaternions of the $\mathbb{H}$ algebra to represent the gimbal-lock free rotation in $SO(3)$ [120]. A quaternion is given by $q = q_1 + q_2 i + q_3 j + q_4 k = (q_1, q_2, q_3, q_4)$ with $(q_1, q_2, q_3, q_4) \in \mathbb{R}^4$ and $i^2 = j^2 = k^2 = ijk = -1$. Rotating V with respect to q can be conducted using $V' = q \cdot V \cdot q^{-1}$, with $\cdot$ denoting the hamilton product and q^{-1} being the quaternion conjugate $q^{-1} = (q_1, -q_2, -q_3, -q_4)$. Using this representation there are still two quaternions $q \equiv -q$ that represent the same rotation $\mathbb{R}^3$. Yet, this can be avoided when, for instance, restricting all quaternions to reside on the upper hemisphere of the $q_1 = 0$ plane [5]. Due to its uniqueness, in most of our works, we leverage quaternions to represent the rotation [7, 5, 6, 4, 3]. Interestingly, notice that [122] has very recently shown that all common representations for the rotation in $SO(3)$ exhibit discontinuities with respect to the 3D rotation. Moreover, they also demonstrated that the accuracy of neural networks for many tasks such as human pose estimation significantly decreases when being close to any discontinuity within the employed representation. Since $SO(3)$ does not embed in $\mathbb{R}^d$ for any $d < 5$, there exist no according homeomorphism which is required for a continuous mapping. Hence, there is no continuous space for rotations $\mathbb{R}^d$ with $d < 5$. As consequence, they propose a novel parameterizations having 5 or more parameters to resolve this issue. This representation has already been successfully applied to the domain of 6D pose estimation with great results [123, 1]. Consider that it is very easy to move between different representation for the 3D rotation.

When using homogeneous coordinates in P^4 the rigid body motion T of the Special Euclidean group $T \in SE(3)$ that brings the object from object to camera space, *i.e.* rotating $\mathcal{M}$ with respect to R and translating it by t, can be expressed by a single transformation matrix

$$V^{Camera} = TV^{Object} = \begin{pmatrix} r_{11} & r_{12} & r_{13} & t_1 \\ r_{21} & r_{22} & r_{13} & t_2 \\ r_{31} & r_{32} & r_{33} & t_3 \\ 0 & 0 & 0 & 1 \end{pmatrix} V^{Object} = \left(\begin{array}{c|c} R & t \\ 0 & 1 \end{array}\right) V^{Object} \tag{3.4}$$

Although all works included in this dissertation rely on directly regressing R and t from the image I, a large group of related works instead employ an intermediate representation, simultaneously encoding rotation and translation [10, 124, 125]. In fact, these works usually regress the 2D projection $p \in \mathbb{R}^{2\times N}$ of N assigned 3D keypoints $P \in \mathbb{R}^{3xN}$. In the following a variant of the Perspective-n-Point (PnP) [124, 126] algorithm is employed to solve for R and t, minimizing the reprojection error

$$\text{PnP}(p, P, K) = \operatorname*{argmin}_{R,t} \frac{1}{N} \sum_{i=1}^{N} ||p_i - \pi((RP_i + t), K)||_2^2, \tag{3.5}$$

with π representing the perspective projection (*c.f.* Section 2.1.1, Equation 2.2) and K being the camera intrinsics matrix. To improve robustness towards bad correspondences, PnP is commonly coupled with RANSAC [127]. These works argue that it is easier to solely infer 2D properties when working on images and, additionally, this representation for pose is invariant to changes of the viewpoint which are caused by a pure translation in 3D. Noteworthy, some methods also combine both ideas, *i.e.* while using keypoints to infer the 3D rotation, translation is instead directly regressed [128].

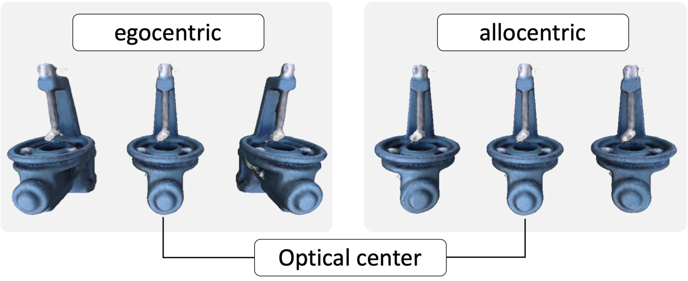


Figure 3.3. **Allocentric *v.s.* Egocentric Pose.** Left: As for the egocentric rotation, a mere 3D translation of the object leads to a different appearance on the image plane. Right: In contrast, the allocentric representation is viewpoint invariant under 3D translation.

Allocentric *v.s.* Egocentric Pose Representation Under perspective projection, a mere 3D translation of the object lateral to the image plane, leads to different object appearance. Therefore, for the same visual structure on the image plane, the network has to estimate different 3D rotations depending on the translation. This is obviously an undesired situation, which becomes particular challenging when cropping into the image [76, 81], as the prediction heads lose the spatial context. To tackle this issue, Kundu *et al*. [129] propose to utilize the allocentric rotation, which is agnostic towards the 3D translation. Knowing the allocentric rotation, one can easily obtain the egocentric rotation.

Given the object's estimated allocentric rotation R_a, the 2D projection of the centroid c, and the camera matrix K, the rotation R_c between the camera principal axis $l = [0, 0, 1]^T$ and the ray through the object center projection $o = K^{-1}c = \frac{t}{||t||}$ is computed. Essentially, R_c takes vector l to align with vector o according to

$$R_c = I_3 + (\sin\alpha)[a]_X + (1 - \cos\alpha)[a]_X^2. \tag{3.6}$$

with I_3 representing the identity matrix in $\mathbb{R}^{3\times3}$, $a = \frac{l \times o}{||l \times o||}$ being the axis between the object ray o and the optical center ray l, $\alpha = \arccos(l \cdot o)$ describing the angle between them, and $[\cdot]_X$ computing the skew-symmetric matrix according to

$$[a]_X = \begin{pmatrix} 0 & -a_3 & a_2 \\ a3 & 0 & -a_1 \\ -a_2 & a_1 & 0 \end{pmatrix}. \tag{3.7}$$

The final egocentric rotation can be derived as $R = R_c R_a$.

3.3 Evaluation

The following will give a brief introduction to all relevant datasets and metrics, leveraged in the works of this dissertation. Notice that $\bar{\cdot}$ always depicts the ground truth for the corresponding predicted parameter $\hat{\cdot}$. Exemplary, $\bar{R}$ and $\hat{R}$ reflect the ground truth and predicted 3D rotation, respectively.

In the last years a lot of methods for solving the 6D pose have been proposed. However, the employed evaluation protocols in terms of datasets and metrics do often not align very well with each other. Furthermore, while some methods use synthetic data alone [9, 130], others make additionally use of a few real samples so to close the domain gap [10, 128].

Hence, in an attempt to unify evaluation and, thus, simplify future comparison, Hodan *et al.* [8] recently established the *Benchmark for Object Pose* (BOP) challenge. BOP encompasses 11 datasets with 7 datasets constituting the core challenge datasets. Thereby, each method has to be evaluated on all core datasets in order to participate in the challenge. Moreover, all methods have to utilize the same synthetic-only training data. The authors essentially argue that annotating a large dataset for each object is infeasible in the real world.

3.3.1 Datasets for Evaluating The Object Pose

Model-based 6DoF pose estimation. Table 3.1 presents an overview of the most important dataset in 6D pose estimation. It is worth mentioning that all datasets are also included in the BOP benchmark [8]. As there are too many datasets to discuss each of them in detail, in the following, the focus lies on the datasets that are most relevant within the scope of this dissertation.

For monocular 6D pose, LineMod (LM) [131], LineMod-Occluded (LM-O) [132] and YCB-Video (YCB-V) [120] are probably the most utilized datasets for evaluation. Thereby, LM is a single-object per image dataset, consisting of 15 sequences with each possessing $\approx$ 1.2k images, possessing clutter and mild occlusion. As LM is a rather simple dataset and results start to saturate at over 90% in ADD (*c.f.* 3.3.1 and [128, 139]), most approaches additionally

			Train. im.		Val im.	Test im.		Test ins.	
Dataset	BOP Core	Objects	Real	Synth.	Real	All	Used	All	Used
LM [131]		15	–	50000	–	18273	3000	18273	3000
LM-O [132]	✓	8	–	50000	–	1214	200	9038	1445
T-LESS [133]	✓	30	37584	50000	–	10080	1000	67308	6423
ITODD [134]	✓	28	–	50000	54	721	721	3041	3041
HB [135]	✓	33	–	50000	4420	13000	300	67542	1630
YCB-V [120]	✓	21	113198	50000	–	20738	900	98547	4123
RU-APC [136]		14	–	–	–	5964	1380	5964	1380
IC-BIN [137]	✓	2	–	50000	–	177	150	2176	1786
IC-MI [138]		6	–	–	–	2067	300	5318	800
TUD-L [8]	✓	3	38288	50000	–	23914	600	23914	600
TYO-L [8]		21	–	–	–	1670	1670	1670	1670

Table 3.1. **Datasets for 6D Pose Estimation.** The table compares the most important datasets for evaluating the 6D object pose. All datasets are also contained in the BOP challenge [8]. The datasets encompass different challenges such as occlusions (LM-O, YCB-V), illumination changes (TUD-L, TYO-L), or symmetries and repetitive patterns (T-LESS). Furthermore, each dataset that is included in the BOP Core challenge also comes with 50k synthetic training images.

evaluate on LM-O or YCB-V since they are significantly more challenging. In particular, for LM-O, Brachmann *et al*. [132] extend one sequence of LM by additionally annotating the pose of 8 other objects being present within this sequence. In contrast to LM, LM-O is therefore a multi-object per image dataset in which objects often undergo medium occlusion. Similarly, YCB-V also exhibits multiple objects per image and possesses medium occlusion. Notice that the results on YCB-V have to be taken with a pinch of salt, since the pose annotations are occasionally not very accurate compared to most other datasets. In particular, YCB-V provides short video sequence, thereby, always annotating the first frame and propagting the labels through the sequence by means of camera tracking. Thus, as the error from camera tracking accumulates over time, the pose quality of later frames degrades due to the drift. Despite occlusion, there are many more challenges aggravating pose estimation. One being ambiguities arising from symmetries. For instance, IC-MI [138] consists of 6 objects with at least 5 being partially symmetric. Thereby, due to symmetries poses can easily become ambiguous when not exposing enough textural information, which can significantly complicate the given task (*c.f.* Section 5.2.1). T-Less [133] consists of many, oftentimes symmetric, simple industrial parts with little texture and a high amount of occlusion. Hence, the lack of textures coupled with object symmetries make it even further challenging than IC-MI. Besides occlusions also other external factors can impede estimating the pose. In fact, changes in illumination can have significant impact on the inferred results. Exemplary, nearby refrigerators the contrast ratio can go beyond 1:1000. TUD Light (TUD-L) and Toyota Light (TYO-L) [8] are trying to tackle this exact problem. While the datasets are rather simple in the geometry of objects and the amount of occlusion, several light sources are added to measure pose accuracy in different illumination setups.

	KITTI [30]	NuScene [140]	Argoverse [141]	Waymo [142]	Lyft Level 5 [36]
Scenes	22	100	113	1150	366
Ann. Lidar Fr.	15K	40K	22K	230K	46K
Hours	1.5	5.5	1	6.4	2.5
3D Boxes	80K	1.4M	993k	12M	1.3M
2D Boxes	80K	–	–	9.9M	323K
Lidars	1	1	2	5	3
Cameras	4	6	9	5	7
Avg Points/Frame	120K	34K	107K	177K	–
Maps	No	Yes	Yes	No	Yes
Visited Area (km^2)	–	5	1.6	76	–

Table 3.2. **Datasets for 7D Pose Estimation.** The table compares the most important datasets for evaluating 3D object detection from monocular data. The main difference resides in the sensor setup and the amount of labeled images. While KITTI was released in 2012 with 80K labeled 3D boxes, new large-scale datasets with millions of annotated instances are now available.

Model-free 7D pose estimation. Table 3.2 shows an overview of important datasets for 7D pose estimation. The most prominent being the KITTI3D benchmark [30] released in 2012. In the following, KITTI3D was for a long time the only public dataset available for evaluating 3D object detection. Only very recently, new large-scale datasets with millions of annotated object instances were released, starting with NuScenes [140] announced in 2018. As collecting and annotating these large datasets is extremely expensive, companies such as Waymo [142] or Lyft [36] are mostly supporting these new datasets. Noteworthy, as these datasets are deliberately large and all related methods used to evaluate on KITTI3D, the transition is still happening at a very low pace. In fact, most works, including ours [6], are purely evaluated on KITTI3D [119, 97].

KITTI3D consists of 7481 training images and 7518 test images recorded while driving in Germany. KITTI3D includes urban as well as high-way scenes, collected in mostly sunny weather. The dataset possesses 3D annotation for 80K instances, involving Cars, Pedestrians, Cyclists, and Vans. Chen *et al.* [118] further splits the training images into training and validation with each split composed of around 3.5K samples. The KITTI3D benchmark suite relies on three different difficulties: easy, moderate, hard dependant on the amount of occlusion the size of the 2D bounding box in pixel space.

3.3.2 Evaluation Metrics

Model-based 6D pose estimation. In [131], Hinterstoisser *et al*. proposed the *Average Distance of Distinguishable Model Points* (ADD) metric which is still to date the most utilized metric for 6D pose estimation. ADD measures whether the average deviation ϵ of the transformed model points V is less than 10% of the object's diameter

$$\epsilon_{ADD} = \underset{v \in V}{\text{avg}} \|(\hat{R}v + \hat{t}) - (\bar{R}v + \bar{t})\|_2. \tag{3.8}$$

Since ADD leverages direct correspondences, it does not work well under symmetries as multiple poses can reflect the same correct transformation [5]. Thus, for *symmetric* objects the *Average Distance of Indistinguishable Model Points* (ADD-S) metric is commonly employed. ADI measures the error as the average distance to the *closest* model point [131, 143].

$$\epsilon_{ADI} = \underset{v_2 \in V}{\text{avg}} \underset{v_1 \in V}{\min} \|(\hat{R}v_1 + \hat{t}) - (\bar{R}v_2 + \bar{t})\|_2. \tag{3.9}$$

Additionally, Hodan *et al.* [143] introduced the *Visible Surface Discrepancy* (VSD) metric, arguing that ADI cannot cover all kinds of visual ambiguities. Let's assume that the handle of a cup is self-occluded. While visually predicting a correct pose, the handle could still point in a different direction, hence, canceling out a good pose estimates according to ADD or ADI. Therefore, VSD leverages only the visual surface to measure pose quality

$$\epsilon_{VSD}(\hat{D}, \bar{D}, \hat{A}, \bar{A}) = \text{avg}_{j \in \hat{A} \cup \bar{A}} \begin{cases} 0 & \text{if } j \in \hat{A} \cap \bar{A} \wedge |\hat{D}(j) - \bar{D}(j)| < \tau \\ 1 & \text{otherwise.} \end{cases} \tag{3.10}$$

Thereby, $\hat{D}$ and $\bar{D}$ depict the predicted and ground truth depth map, obtained from rendering $\mathcal{M}$ with the predicted pose $(\hat{R}|\hat{t})$ and the ground truth pose $(\bar{R}|\bar{t})$, respectively. Further, $\hat{A}$ and $\bar{A}$ illustrate the respective visible surface, computed via comparing $\hat{D}$ and $\bar{D}$ with the depth map from the sensor D^S (*e.g.* $\hat{A}_j$ is equal to 1 if $\hat{D}_j \leqslant D^S_j$ and 0 otherwise), and j denotes the pixel location in I. The parameter τ constitutes the misalignment tolerance.

Since for many applications in augmented reality it is sufficient to have a high visual overlap, *i.e.* depth perfect results are not required, the *Visible Surface Similarity* (VSS) metric relaxes VSD by only considering the visual overlap [9, 7]

$$\epsilon_{VSS}(\hat{A}, \bar{A}) = \text{avg}_{j \in \hat{A} \cup \bar{A}} \begin{cases} 0 & \text{if } j \in \hat{A} \cap \bar{A} \\ 1 & \text{otherwise.} \end{cases} \tag{3.11}$$

Thereby, a pose is accepted as correct if the error ϵ_{VSS} is below 0.5.

Model-free 7D pose estimation. The performance of 3D object detectors is commonly measured referring to the intersection of union (IoU) metric. Thereby, the 2D IoU with respect to the 2D bounding boxes in the image space and the 3D IoU with respect to the 3D bounding boxes in 3D space are measured [30].

The IoU of two bounding boxes is defined as the area of intersection over the area of union.

$$\text{IoU}(\hat{S}, \bar{S}) = \frac{S_{\text{intersection}}}{S_{\text{union}}} = \frac{\hat{S} \cap \bar{S}}{\hat{S} \cup \bar{S}}. \tag{3.12}$$

Given the predicted and ground truth 2D bounding box $\hat{B}^{2D}$ and $\bar{B}^{2D}$, we first calculate the respective area $\hat{S}^{2D}$ and $\bar{S}^{2D}$ for each bounding box. From this the 2D IoU is computed as $\mathrm{IoU}(\hat{S}^{2D}, \bar{S}^{2D})$. The 3D IoU follows the same principle, however, computes the IoU with respect to the volume $\hat{S}^{3D}$ and $\bar{S}^{3D}$ of the 3D bounding boxes $\hat{B}^{3D}$ and $\bar{B}^{3D}$ according to $\mathrm{IoU}(\hat{S}^{3D}, \bar{S}^{3D})$.

Since it is not crucial to know the exact height of other traffic participants to drive safely, most works additionally leverage the 2D IoU with respect the 3D box as perceived from the bird's eye view (BEV). The BEV IoU is computed by means of an orthogonal projection $B^{2D}_{BEV} = \pi_{ortho}(B^{3D})$ of the 3D bounding box onto the $y = 0$ plane with $pi_{ortho}((X, Y, Z)^T) = (X, Z)^T$. Finally, the 2D IoU is computed on the resulting 2D bounding boxes after projection $\mathrm{IoU}(\hat{S}^{2D}_{BEV}, \bar{S}^{2D}_{BEV})$, with $\hat{S}^{2D}_{BEV}$ and $\bar{S}^{2D}_{BEV}$ denoting the predicted and ground truth area in 2D of the projected 3D boxes $\hat{B}^{2D}_{BEV}$ and $\bar{B}^{2D}_{BEV}$, respectively.

For all metrics a prediction is considered as positive if the associated IoU is larger than a threshold τ. Exemplary, the default for τ is set to 0.7 for KITTI3D [30]. Furthermore, for each metric the average precision (AP) is reported, computing the area underneath the precision-recall curve [144]. While recall computes the ratio of all correctly predicted samples over all positive samples within the dataset, precision measures the ratio of all correct predictions over all casted predictions (*i.e.* true and false positives).

4 Recent History of Monocular Object Pose Estimation

Estimating the object pose is a very active field of research and, in fact, is still growing rapidly [8]. In this chapter an overview over published methods related to this dissertation is provided. There are many ways to disentangle the field of 6D pose estimation. Exemplary, in this dissertation, the works are separated according to the amount of degrees-of-freedom they are estimating. Thereby, while the first section covers instance-level 6D pose estimation, the latter two sections tackle the problem of model-free 7D and 9D pose estimation, respectively.

Traditionally, monocular object pose estimation methods are ground on local image features such SIFT [145, 146] or template matching [147]. With the advent of consumer RGB-D cameras, the focus shifted more towards conducting object pose estimation from RGB-D data. While some works again propose to utilize template matching [131], others leverage handcrafted 3D descriptors such as point pair features [148, 149] or rely on learning-based methods [132, 150] in order to predict the 6D pose. Nonetheless, depth data also comes with limitations such as restricted field of view or high power consumption. Recently, CNN-based methods have demonstrated promising results for the task of monocular 6D pose estimation [8]. Hence, in the following, the focus will be on deep learning based methods as these are the most relevant within the scope of this dissertation.

4.1 Full Model-based 6D Pose Estimation

The field of monocular 6D pose estimation can be again divided into three different sub-areas. While the first line of works directly regress or classify the 6D pose, the latter either learn a latent embedding for subsequent retrieval or establish 2D-3D correspondences prior to estimating the 6D pose.

4.1.1 Direct Regression of The 6D Pose

As for the first branch, in our work SSD-6D [9], we extend SSD [59] to also retrieve the pose of the object, turning the regression into a classification problem. Thereby, we discretize the 3D rotation through binning of viewpoint and in-plane rotation and infer the 3D translation from the perspective ratio of the detection bounding box and the rendered bounding box. In our follow-up work [5], we adopt SSD-6D to implicitly deal with ambiguities by means of

Method Class	Method	Input Data	Backbone	Loss function
Direct Regression	SSD-6D [9]	RGB	Incpetion-V4 [57]	Viewpoint and in-plane classification
	PoseCNN [120]	RGB	VGG-16 [54]	Average distance of closest 3D model points
	Deep-6D-Pose [151]	RGB	Mask R-CNN [76]	L2 norm with rotation parameterized with Li algebra
	MHP [5]	RGB	Incpetion-V4 [57]	Extended SSD-6D with loss for multiple hypotheses
Latent Embedding	Wohlhart [152]	RGB-D	Global Patches	Triplet-Pairs loss
	Kehl [153]	RGB-D	Local Patches	Auto-Encoder
	Zakharov [154]	RGB-D	Global Patches	Triplet-Pairs loss with dynamic margin
	AAE [130]	RGB	Global Patches from SSD [59]	Auto-Encoder
	MP AAE [155]	RGB	Global Patches from Mask R-CNN [76]	Mutipath Auto-Encoder
Correspondence driven	Brachmann [132]	RGB-D	Random Forest	Dense 3D-3D
	Brachmann [156]	RGB	Random Forest	Dense 2D-3D
	BB-8 [10]	RGB	VGG-16 [54]	Sparse 2D-3D from BBox
	Yolo-6D [124]	RGB	Yolo [58]	Sparse 2D-3D from Bbox
	Oberweger [157]	RGB	Unet-like CNN with residuals	Sparse 2D-3D from Bbox
	PVNet [158]	RGB	ResNet-18 [55]	Sparse 2D-3D from farthest point
	CDPN [125]	RGB	Modified ResNet [55]	Dense 2D-3D from uv-texture
	Pix2Pose [159]	RGB	U-Net [160]	Dense 2D-3D with GAN loss
	CDPN [128]	RGB	Tiny Yolov3 [161]	Dense 2D-3D with translation regression
	Hu [162]	RGB	ResNet [55] & PointNet [101]	Sparse 2D-3D with learned PnP from PointNet-like architecture
	GDR-Net [1]	RGB	Faster-RCNN [74] & FCOS [73]	Dense 2D-3D with learned PnP using a CNN
	Hybrid-Pose [139]	RGB	ResNet [55]	Sparse 2D-3D with edge and symmetry features
	EPOS [163]	RGB	DeepLabV3 [164]	Dense 2D-3D using fragments

Table 4.1. **Related Works on Monocular 6D Object Pose Estimation.** The methods are coarsely divided into 3 branches and mostly differ in the employed backbone and loss functions. Notice that also some works from the RGB-D realm are shown as they served as inspiration for subsequent works within the category.

multiple hypotheses. The work of Xiang *et al.* [120] instead localizes the object in 2D image space using semantic segmentation paired with hough voting, and regresses the remaining parameters via minimization of a point matching loss. Similarly, also Li *et al.* [165] employ the point matching loss for the task of object pose refinement. Thereby, the authors propose to leverage a disentangled representation for 3D orientation and 3D translation to improve robustness to change in viewpoint induced by mere translation in 3D. Inspired by [165], Labbé *et al.* [123] estimate the pose by means of pose refinement using the point matching loss. In the following they match estimates from different views and recover a consistent scene model via global refinement of object and camera poses.

4.1.2 Latent Embeddings for 6D Pose Retrieval

The second line of works harnesses latent embeddings for pose and occasionally object to recover the 6D pose. Wohlhart and Lepetit [152] leverage ideas from metric learning to attain a robust descriptor for object and pose from RGB-D data. In particular, they employ a triplet-pairs loss to map each input to the appropriate location in feature space. Thereby, the triplet loss ensures that the same object in a similar pose is mapped close together in feature space, whereas a different object or the same object in a different pose is pushed far away from each other. In addition, the pairs loss is responsible to make the approach more robust with respect to noise and other distracting artifacts such as illumination. The work of Zakharov *et al*. [154] further adjusts [152] with a dynamic margin for the triplets term to enhance the class separation of the manifold. Rather than metric learning, Kehl *et al*. [166] instead leverage an AutoEncoder to learn a latent embedding from local RGB-D patches. Afterwards, they sample patches in a sliding window fashion and cast votes for object and pose via k-nearest neighbor look-up in an precomputed codebook. In the following, nearby votes for the same object are clustered and mean shift with a flat kernel is employed to recover the final pose. Following this line of works, Sundermeyer *et al*. [130] similarly employ an AutoEncoder for pose estimation. In order to make it robust to noise as well as occlusion and illumination, the authors propose to feed the AutoEncoder with augmented input samples, yet, compute the loss with respect to the clean version of the given input. Thus, the network is forced to map the same object and pose to the same location even under external impairments. Further, instead of using a sliding window based approach, they first localize the object in image space using SSD [59]. Afterwards, similar to [166], they conduct look-up in feature space with respect to a pre-computed codebook, to retrieve the 3D rotation. Following our work [9], they also estimate the object depth by means of bounding box ratios. The final 3D translation is then computed via backprojection of the 2D centroid given the camera intrinsics and the inferred depth. In their follow up work [155], the authors propose the use of a single shared encoder for multiple objects, while utilizing object specific decoders. As consequence, objects sharing similar features are not forced to be disentangled within the feature space, improving the scalability to multiple instances, categories and datasets. They also demonstrate that their "multi-path" training is particularly suited for synthetic to the real domain transfer.

4.1.3 Correspondence-driven 6D Pose Estimation

Finally, the most popular branch is based on establishing 2D-3D correspondences, which are then further processed to solve for the 6D pose using a variant of the PnP & RANSAC paradigm [127, 126]. Inspired by [132, 156], Rad and Lepetit [10] propose to regress the 2D projection of the 3D bounding box corners. Leveraging these correspondences, PnP [126] allows to solve for the 6D pose by minimizing the 2D reprojection error. To this end, the authors employ VGG [54] to detect the object of interest by means of semantic segmentation.

Thereafter, they crop the object from the image and leverage another VGG instance in order to regress the aforementioned 2D projections of the 3D bounding box corners in image space. Oberweger *et al*. [157] predict heatmaps from multiple small patches to improve robustness towards occlusion. To reduce the computation burden and thus enhance inference speed, the authors of [124] instead rely on Yolo [58] to regress the same control points as BB-8 in a single-shot fashion, enabling real-time complexity at higher accuracy. Noteworthy, Tekin *et al*. [124] add the projection of the 3D centroid as an additional control point to further stabilize pose inference. Unfortunately, as pointed out in [167], occlusion and other external factors can deteriorate the accuracy of individual correspondences, which has direct negative impact on the final pose estimate. Therefore, *et al*. [167] propose to cast multiple hypotheses for pose, in the form of 2D projections. In the core, YoloV3 [161] is adopted to segment the input image into $S \times S$ superpixels, with each segmented pixel casting votes for the 2D coordinates of the bounding box corners. After clustering, the best $n = 10$ predictions for each correspondence are eventually utilized together with PnP and RANSAC to obtain the final pose. In the following, there was a strong trend towards establishing 2D-3D correspondences with respect to the object model rather than the bounding box corners. Exemplary, Peng *et al*. [158] argue that keypoints farther away from the object surface introduce larger errors, and thus instead propose to use farthest point sampling on the object model to retrieve 8 keypoints for each object. In addition, instead of directly predicting these keypoints, the authors employ pixel-wise segmentation. Thereby, each segmented pixel predicts unit vectors pointing towards each keypoint. Via sampling of two vectors a candidate for each correspondence can be generated. Moreover, repeatedly sampling further enables to estimate the uncertainty within each correspondence based on the covariance of these samples. These obtained uncertainties are then exploited within PnP to enhance robustness in pose. Hu *et al*. [162] also compute multiple predictions for each 2D-3D correspondence, however, leverage a PointNet-like architecture to recover the final pose. As shown in their experiments, the learned PnP formulation is more robust towards noise than standard PnP with RANSAC or the uncertainty-driven variant from [158]. Similarly, *GDR-Net* also attempts to learn the PnP paradigm, yet, employs a common CNN instead of PointNet [1]. *HybridPose* leverages multiple intermediate representations to estimate the 6D pose. Besides 2D-3D correspondences, Song *et al*. [139] also infer edge vectors and symmetry correspondences to exploit more and diverse features to strengthen the robustness in case one representation fails due to *e.g.* occlusion. Most recent works, however, rely on dense pixel-level correspondences [125, 128, 159, 163]. Exemplary, provided the 3D model of interest, Zakharov *et al*. [125] apply a correspondence texture to the object. The resulting correspondence model is then employed to render groundtruth uv-maps with respect to model and pose. Similar to [9], the authors relax the regression problem by turning it into a classification task. In particular, the groundtruth uv-maps are discretized into 256 bins and optimized via cross-entropy loss with respect to each channel of the uv-map. Park *et al*. [159] instead predict the 3D model coordinate for each visible object pixel. Thereby, they first localize the object in 2D using a modified Faster R-CNN [74] framework, grounded on ResNet-101 [56]. Afterwards, they crop the object and feed them to an AutoEncoder with skip connections [160] to estimate dense correspondences. The core novelty of this work lies inside the employed GAN architecture and the simultaneous estimation of an correspondence error map. Alternately feeding the discriminator with real and predicted correspondence maps, the discriminator has to learn to distinguish between real and fake, which in turn forces the predictor to estimate coherent maps, *i.e*. neighboring pixel have to map to similar 3D coordinates. An error map is additionally employed to filter bad correspondences, thus, returning more reliable poses. In [163], Hodan *et al*. tackles the problem of ambiguities within correspondence-driven methods. Thereby, they disassemble the objects into fragments. In the following, for each visible object pixel the network attempts to classify on which object fragment it is residing. Furthermore, they also regress an offset to estimate the exact location of the keypoint within the fragment. Training this approach with cross-entropy loss allows to retrieve the actual underlying distribution under ambiguities. In essence, all fragments that are visually equal due to ambiguities are supposed to be scored with the same probability. Leveraging an efficient variant of the PnP & RANSAC [168, 169] paradigm, the final pose can be estimated despite the presence of ambiguities. Finally, *CDPN* also leverages dense correspondences, yet, disentangles rotation and translation [128]. In fact, correspondences from the regressed coordinates map are employed to only retrieve the 3D rotation. On the other hand, a second branch infers directly the scale-invariant translation, estimating the object depth relative to the resize ratio of the RoI.

4.2 Model-free 7D Pose Estimation.

Works in monocular 3D object detection attempt to regress a tight 3D bounding box for each object of interest. The objects are thereby not arbitrary but rather stem from known classes. Due to the ground-plane assumption the task involves the estimation of 7 degrees-of-freedom (*i.e*. 3 for translation, 3 for object extents, 1 for rotation) (*c.f.* Section 3.1). Monocular 3D object detection can be roughly divided into methods leveraging an extra module for depth estimation [6, 119] and methods directly outputting 3D detections [118, 117]. Approaches relying on depth prediction can be further partitioned into works which utilize pseudo-lidar representations [170, 171] and works that simply feed the predicted depth map as an extra input channel [6].

4.2.1 Direct 7D Pose Estimation

In 2016, Chen *et al*. [118] introduced *Mono3D* which is one of the pioneering works in the field. *Mono3D* instantiates 3D proposals based on multiple hand-crafted features grounded on semantic segmentation, location and spatial context. Afterwards, a simple CNN is employed to score each proposal and regress its extents and orientation. *GS3D* predicts 2D bounding boxes together with an orientation angle [172]. In addition, 3D guidance is established leveraging the training data distribution. Finally, surface features are extracted from the 3D guidance and employed to refine the 3D proposal in order to get the final estimate. Kundu *et al*. [129] predict pose and shape of cars employing a render-and-compare loss. Using PCA, a ten-dimensional shape basis $S_B \in \mathbb{R}^{n \times 10}$ is calculated based on truncated signed distance fields (TSDF). As standard rendering is not differentiable, Kundu *et al*. approximate gradients by means of finite differences. Noteworthy, poses are estimated only up-to-scale. Simonelli *et* *al*. [117] employ an extended RetinaNet [75] together with a signed IoU loss for 2D and our corner loss for 3D [6]. As these losses exhibit instabilities during training, they propose a new disentanglement strategy. Thereby, the loss is computed for each component individually, harnessing the ground truth for the remaining terms. In [173], the authors further improve their approach by means of virtual cameras. Thereby, they argue that covering the whole 3D space requires a lot of training data. In addition, when a car appears at an under-represented position, the network will most likely perform poorly. To this end, Simonelli *et al*. place virtual cameras such that each camera at least fully shows one object and the distance towards the virtual camera is always within a well-represented range. Ku *et al*. [174] leverage instance-centric 3D proposals and local shape reconstruction. Instance point clouds are estimated to recover local shape and scale, and to enforce 2D-3D consistency. Brazil and Liu [175] employ a shared 2D-3D space by means of 2D-3D anchors, which are based on pre-computed statistics from the training data. They further introduce the idea of depth-aware convolutions. Thereby, Brazil and Liu bin the height of the feature map into b individual bins with each bin possessing its own set of filters. According to the authors, the depth levels of a scene in autonomous driving can be roughly described by such binning. The predicted 3D bounding boxes from the associated anchors are then postprocessed enforcing 2D-3D consistencies. *Monopair* attempts to improve monocular 3D object detection by considering mutual spatial relationships of objects [176]. Essentially, besides each 3D bounding box, Chen *et al*. also predict the distance between the individual cars. In a second step, bounding boxes are globally optimized from a graph perspective in respect of the pair-wise constraints.

4.2.2 7D Pose Estimation With Depth Module

The second branch of works additionally leverages depth prediction to improve robustness. As aforementioned, this branch can be further subdivided into works that directly process the predicted depth map and approaches which instead utilize pseudo-liar representations. As for the former line of works, Xu and Chen [115] localize objects in 2D using Faster R-CNN [74] with VGG-16 backbone [54]. Afterwards, *MonoDepth* [121] is employed to estimate depth. The 3D data is then fused at multiple levels to regress the final output. In our work RoI-10D [6], we similarly use Faster R-CNN with a ResNet-34 [55] backbone for 2D detection, however, obtain depth from *SuperDepth* [177]. The cropped detections are then concatenated with the predicted depth maps using RoI-Align [76] and lifted to 3D bounding box and shape. To this end, a 3D AutoEncoder is trained on-top of TSDFs. Since weighting different loss terms is difficult and often leads to inferior results, we introduce a novel 3D lifting loss measuring the metric misalignment of the 3D bounding box corners. *MonoGRNet* utilizes instance-level depth together with a progressive scheme to localize the object in 3D [178]. Finally, the 3D rotation is recovered estimating local 3D corners with respect to the 3D centroid. Beker *et al*. [97] leverage differentiable rendering to optimize for the 3D bounding box. In essence, using the neural mesh renderer by Kato *et al*. [105](*c.f.* Section 2.5), the authors iteratively optimize pose and shape by comparing the rendering against the detections from an off-the-shelf Mask R-CNN detector [76] and the depth map from an off-the-shelf depth predictor [179]. Ding *et al*. [180] extend the idea of depth-aware convolutions from [175].

However, the authors learn the filters and receptive field from the depth maps, as estimated by *DORN* [181]. Thereby, different pixels of different images can employ different filters.

In regard of the second partition, one of most prominent works is known as *Pseudo-lidar* from [170]. In particular, Wang *et al*. [170] introduce the concept of pseudo-lidar obtained from backprojection of the depth map produced by *DORN* [181]. Eventually, an off-the-shelf 3D object detector [182, 29] is harnessed to retrieve the final 3D predictions. Similarly, Ma *et al*. [171] also estimate pseudo-lidar based on *DORN* [29]. Subsequently, they use a PointNet-like [101] backbone for segmentation and pose estimation, leveraging our 3D corner loss [6]. In their follow-up work, Ma *et al*. [119] again compute pseudo-lidar, however, argue that a PointNet-style architecture is not the preferable choice as it does not consider the fact that pseudo-lidar point clouds are organized with respect to the image plane. Hence, they instead apply standard 2D convolutions producing superior results.

4.3 Model-free 9D Pose Estimation.

Despite the focus being on monocular methods, there are only RGB-D methods for the domain of full model-free 9D pose estimation. Yet, since it is a very recent field which is fairly related to 6D pose estimation, I want to briefly outline the most important works. Similar to 7D pose estimation, the task involves the prediction of a tight 3D bounding box, however, the ground-plane assumption is not valid anymore, making it a problem with 9 degrees-of-freedom (*c.f.* Section 3.1).

Wang *et al*. [116] recently proposed the first method for class-level object detection together with full 9D pose estimation. Using an extended Mask R-CNN [76] backbone, the authors predict a 2D map constituting the projection of the Normalized Object Coordinate Space (NOCS). NOCS essentially depicts a 3D space spanned by a unit cube. All objects from a certain category are normalized to lie within the NOCS space and each vertex is assigned its 3D location within the cube. The predicted 2D NOCS map is then backprojected, using the associated depth map, to establish 3D-3D correspondences. Leveraging these correspondences together with the Umeyama algorithm [183], enables the estimation of both 6D pose and metric scale for previously unseen objects. Chen *et al*. [184] instead conducts class-level object pose and size estimation without the need for correspondences. They learn a canonical shape space (CASS) from 3D pointclouds using a variational AutoEncoder (VAE) grounded on PointNet [101, 19]. In the following, leveraging the input RGB-D data, Chen *et al*. predict the latent representation to infer shape and metric size. Further, after fusing geometric with photometric features, the 3D rotation and translation is estimated using the point-matching loss between predicted and groundtruth shape. Similarly, Tian *et al*. [185] also compute pose and the 3D shape of the object as pointcloud. To this end, Tian *et al*. leverage a PointNet-based AutoEncoder to compute a template shape for each class, obtained as the mean of all training shapes. During inference this mean shape is then deformed using a 3D deformation field to obtain the precise full 3D model. Further, based on NOCS, a correspondence matrix is learned to establish 3D-3D correspondences between the input pointcloud and the reconstruction. Finally, as in [116], pose and scale can be estimated harnessing the Umeyama algorithm [183].

Recently, Park *et al*. [186] propose the first method for 6D object pose estimation of *fully* unseen objects without any prior information (*i.e*. no category information is utilized). Taking multiple RGB-D views as input, a 2D-3D U-Net is leveraged to fuse the input into a latent object. Afterwards, harnessing neural rendering the latent object can be rendered from novel views. These views are than iteratively compared with the input data and the error is back-propagated to the pose parameters. Noteworthy, *LatentFusion* requires to compute gradients during inference which is typically slow and does not allow real-time performance.

Summary of Contributions

5

This chapter summarizes the main contributions and additionally provides the associated publication for each work. Thereby, Section 5.1 presents one of the first deep learning driven methods for monocular 6D pose estimation and refinement. Afterwards, Section 5.2 demonstrates different challenges and solutions when estimating the 6D pose. Finally, in Section 5.3 we tackle the problem of class-level object pose and shape estimation for previously unseen instances.

5.1 Fast and Reliable 6D Pose Estimation

This section presents our works for estimating the 6D object pose from monocular data alone. As poses are oftentimes noisy, most works typically rely on ICP for pose refinement [131]. Since we attempt at avoiding the use of depth data, we are not able to harness ICP. Hence, we additionally proposed one of the first methods for 6D pose refinement from monocular data employing deep learning.

5.1.1 SSD-6D: Making RGB-Based 3D Detection and 6D Pose Estimation Great Again (ICCV Oral 2017)

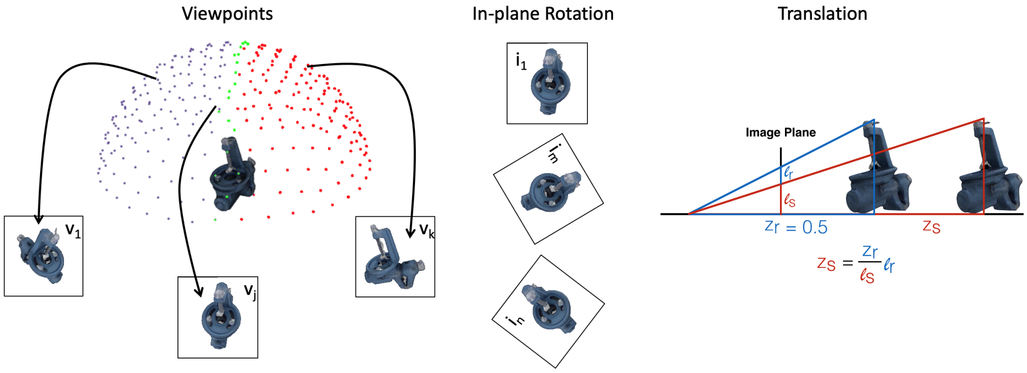


Figure 5.1. **Single-Shot Monocular 6D Pose Estimation.** Left: We turn the regression of the rotation into a classification problem via binning of viewpoint and in-plane rotation. Right: The associated 3D translation can be inferred comparing the predicted bounding box diagonal with the corresponding diagonal from the associated object rendered at a canonical distance of 0.5m.

In this work, we introduce one of the very first deep learning based approach for monocular 6D object pose estimation [9]. Essentially, building on top of recent advances in 2D object detection [74, 59, 76], we extend the Single Shot MultiBox Detector (SSD) [59] to classify viewpoint and in-plane rotation. As aforementioned in Section 2.3, SSD distributes anchor boxes at different scales over the image, which are then classified in a single pass. Thus, in contrast to other region proposal based approaches such as Faster R-CNN[74] or Marsk R-CNN[76] which run at a frame rate of approximately 7Hz, SSD can process images in real-time. During training we assigned all anchor boxes, having an IoU overlap (*c.f.* Section 3.3.1) larger than 0.5 with the ground truth box, with the corresponding label.

To relax the problem of directly regressing the 3D rotation, we turn it into a classification problem, via decomposition of rotation into viewpoint and in-plane rotation. As demonstrated in Figure 5.1 [left], we sample equidistant viewpoints on a unit sphere around the object. Similarly, we also sample discrete in-plane rotations with a step size of 5°. Afterwards, for a given rotation we compute the corresponding viewpoint & in-plane representation and map them to the closest discrete bin. Finally, we add two loss terms for classification of viewpoint $\mathcal{L}_{view}$ and in-plane rotation $\mathcal{L}_{inplane}$ to [59]

$$\mathcal{L}(Pos, Neg) := \sum_{b \in Neg} \mathcal{L}_{class} + \sum_{b \in Pos} (\mathcal{L}_{class} + \alpha\mathcal{L}_{fit} + \beta\mathcal{L}_{view} + \gamma\mathcal{L}_{inplane}). \tag{5.1}$$

Thereby, Pos and Neg depict occupied and unoccupied anchor boxes, respectively. We further employ the cross-entropy loss for $\mathcal{L}_{view}$ and $\mathcal{L}_{inplane}$, and the l_1-loss for $\mathcal{L}_{fit}$ which refine the 2D bounding boxes to obtain a tight fit. During inference we can thus easily obtain the 3D rotation from the retrieved viewpoint and in-plane rotation ID. To recover the 3D translation, we render the rotated object at a canonical centroid distance of $z_r = 0.5m$. We then estimate the depth z from the perspective ratio of the rendered bounding box diagonal l_r and the diagonal of the detected bounding box l_s according to $z = \frac{z_r}{l_s} l_r$ (*c.f.* Figure 5.1 [right]). The 3D translation is then simply computed via backprojection of the 2D centroid using the camera intrinsics together with the estimated distance $t = K^{-1}z\,(x, y, 1)^T$.

While other methods require substantial computation to infer the pose information and oftentimes limit the pose space, our method runs at 10Hz and is capable of even handling the full pose space. Moreover, we only rely on synthetic data, hence, do not require any large annotated dataset, which is another typical limitation for many deep learning driven methods [135, 8]. Our approach competes or surpasses state-of-the-art methods that leverage RGB-D data on multiple challenging datasets such as LM [131] and IC-MI [138] with respect to ADD and VSS.

Together with the work from Rad *et al.* [10], our approach started a still ongoing hype for the field monocular 6D pose estimation, constantly pushing the envelope of 6D pose estimation [120, 124, 128, 123].

Contributions I proposed and implemented the method for extending SSD with loss terms for 6D pose estimation. I also ran all evaluations on LM and IC-MI. Wadim Kehl implemented the pose refinement in 2D and 3D and helped conducting the experiments, especially for LM-(O).

SSD-6D: Making RGB-based 3D Detection And 6D Pose Estimation Great Again

Fabian Manhardt[2,*], Wadim Kehl[1,2,*], Federico Tombari[2], Slobodan Ilic[2,3], Nassir Navab[2]

[1] Toyota Research Institute, Los Altos
[2] Technical University of Munich
[3] Siemens R&D, Munich
* Equal Contribution

It is the accepted but not the published version of the paper due to copyright restrictions.

Published version: https://doi.org/10.1109/ICCV.2017.169

SSD-6D: Making RGB-Based 3D Detection and 6D Pose Estimation Great Again

Wadim Kehl [1,2,*] Fabian Manhardt [2,*] Federico Tombari [2] Slobodan Ilic [2,3] Nassir Navab [2]
[1] Toyota Research Institute, Los Altos [2] Technical University of Munich [3] Siemens R&D, Munich
wadim.kehl@tri.global fabian.manhardt@tum.de tombari@in.tum.de

Abstract

We present a novel method for detecting 3D model instances and estimating their 6D poses from RGB data in a single shot. To this end, we extend the popular SSD paradigm to cover the full 6D pose space and train on synthetic model data only. Our approach competes or surpasses current state-of-the-art methods that leverage RGB-D data on multiple challenging datasets. Furthermore, our method produces these results at around 10Hz, which is many times faster than the related methods. For the sake of reproducibility, we make our trained networks and detection code publicly available.[1]

1. Introduction

While category-level classification and detection from images has recently experienced a tremendous leap forward thanks to deep learning, the same has not yet happened for what concerns 3D model localization and 6D object pose estimation. In contrast to large-scale classification challenges such as PASCAL VOC [9] or ILSVRC [26], the domain of 6D pose estimation requires instance detection of known 3D CAD models with high precision and accurate poses, as demanded by applications in the context of augmented reality and robotic manipulation.

Most of the best performing 3D detectors follow a view-based paradigm, in which a discrete set of object views is generated and used for subsequent feature computation [31, 14]. During testing, the scene is sampled at discrete positions, features computed and then matched against the object database to establish correspondences among training views and scene locations. Features can either be an encoding of image properties (color gradients, depth values, normal orientations) [12, 16, 18] or, more recently, the result of learning [4, 29, 5, 6, 17]. In either case, the accuracy of both detection and pose estimation hinges on three aspects: (1) the coverage of the 6D pose space in terms of viewpoint and scale, (2) the discriminative power of the features to tell objects and views apart and (3) the robustness of matching towards clutter, illumination and occlusion.

CNN-based category detectors such as YOLO [25] or SSD [22] have shown terrific results on large-scale 2D datasets. Their idea is to inverse the sampling strategy such that scene sampling is not anymore a set of discrete input points leading to continuous output. Instead, the input space is dense on the whole image and the output space is discretized into many overlapping bounding boxes of varying shapes and sizes. This inversion allows for smooth scale search over many differently-sized feature maps and simultaneous classification of all boxes in a single pass. In order to compensate for the discretization of the output domain, each bounding box regresses a refinement of its corners.

The goal of this work is to develop a deep network for object detection that can accurately deal with 3D models and 6D pose estimation by assuming an RGB image as unique input at test time. To this end, we bring the concept of SSD over to this domain with the following contributions: (1) a training stage that makes use of synthetic 3D model information only, (2) a decomposition of the model pose space that allows for easy training and handling of symmetries and (3) an extension of SSD that produces 2D detections and infers proper 6D poses.

We argue that in most cases, color information alone can already provide close to perfect detection rates with good poses. Although our method does not need depth data, it is readily available with RGB-D sensors and almost all recent state-of-the-art 3D detectors make use of it for both feature computation and final pose refinement. We will thus treat depth as an optional modality for hypothesis verification and pose refinement and will assess the performance of our method with both 2D and 3D error metrics on multiple challenging datasets for the case of RGB and RGB-D data.

Throughout experimental results on multiple benchmark datasets, we demonstrate that our color-based approach is competitive with respect to state-of-the-art detectors that leverage RGB-D data or can even outperform them, while being many times faster. Indeed, we show that the prevalent trend of overly relying on depth for 3D instance detection is not justified when using color correctly.

[1] https://wadimkehl.github.io/

* The first two authors contributed equally to this work.

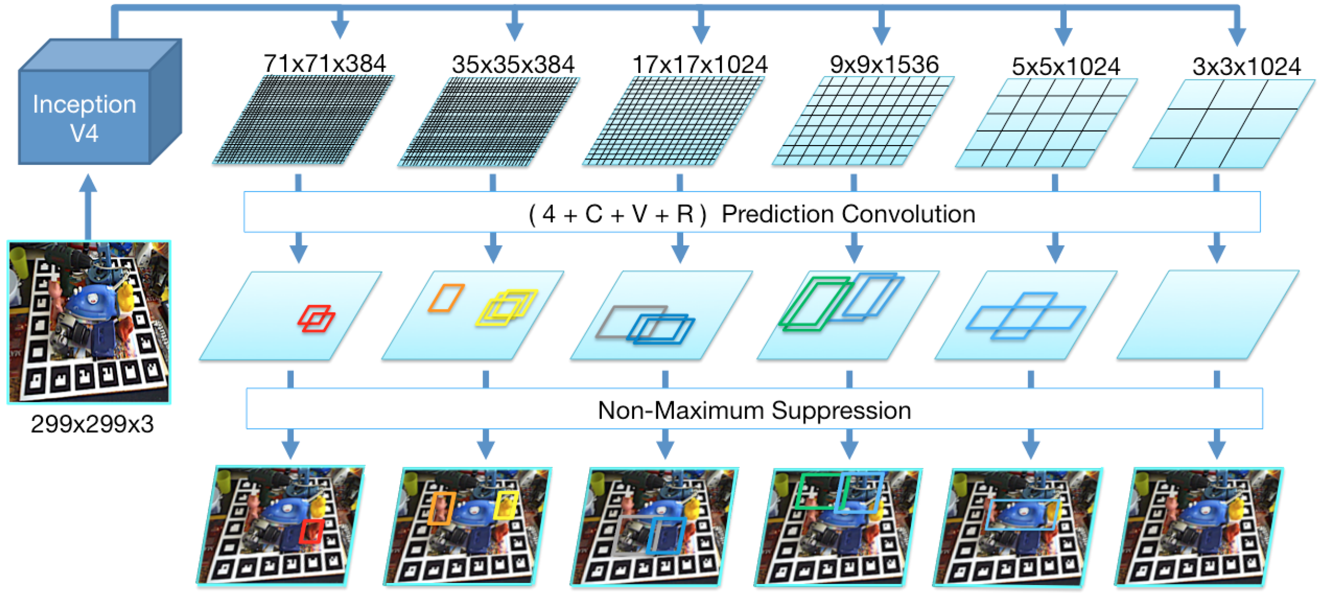


Figure 1: Schematic overview of the SSD-style network prediction. We feed our network with a 299 × 299 RGB image and produce six feature maps at different scales from the input image using branches from InceptionV4. Each map is then convolved with trained prediction kernels of shape (4 + C + V + R) to determine object class, 2D bounding box as well as scores for possible viewpoints and in-plane rotations that are parsed to build 6D pose hypotheses. Thereby, C denotes the number of object classes, V the number of viewpoints and R the number of in-plane rotation classes. The other 4 values are utilized to refine the corners of the discrete bounding boxes to tightly fit the detected object.

2. Related work

We will first focus on recent work in the domain of 3D detection and 6D pose estimation before taking a closer look at SSD-style methods for category-level problems.

To cover the upper hemisphere of one object with a small degree of in-plane rotation at multiple distances, the authors in [14] need 3115 template views over contour gradients and interior normals. Hashing of such views has been used to achieve sub-linear matching complexity [18, 16], but this usually trades speed for accuracy. Related scale-invariant approaches [16, 4, 29, 6, 17] employ depth information as an integral part for either feature learning or extraction, thus avoiding scale-space search and cutting down the number of views by around an order of magnitude. Since they require depth to work, they can fail when depth is missing or erroneous. While scale can be inferred with RGB-D data, there has not been yet any convincing work to eradicate the requirement of in-plane rotated views. Rotation-invariant methods are based on local keypoints in either 2D [32] or 3D [7, 3, 30] by explicitly computing or voting for an orientation or a local reference frame, but they fail for objects of poor geometry or texture.

Although rarely mentioned, all of the view-based methods cover only a very small, predefined 6D pose space. Placing the object differently, e.g. on its head, would lead to failure if this view had not been specifically included during training. Unfortunately, additional views increase computation and add to overall ambiguity in the matching stage. Even worse, for all discussed methods, scene sampling is crucial. If too coarse, objects of smaller scale can be missed whereas a fine-grained sampling increases computation and often leads to more false positive detections. Therefore, we explore a path similar to works on large-scale classification where dense feature maps on multiple scales have produced state-of-the-art results. Instead of relying on classifying proposed bounding boxes [10, 11, 21], whose performance hinges on the proposals' quality, recent single-shot detectors [25, 22] classify a (large) discrete set of fixed bounding boxes. This streamlines the network architecture and gives freedom to the a-priori placement of boxes.

As for works regressing the pose from RGB images, the related works of [24, 23] recently extended SSD to include pose estimates for categories. [23] infers 3D bounding boxes of objects in urban traffic and regresses 3D box corners and an azimuth angle whereas [24] introduces an additional binning of poses to express not only the category but also a notion of local orientation such as 'bike from the side' or 'plane from below'. The difference to us is that they train on real images to predict poses in a very constrained subspace. Instead, our domain demands training on synthetic model-based data and the need to encompass the full

6D pose space to accomplish tasks such as grasping or AR.

3. Methodology

The input to our method is an RGB image that is processed by the network to output localized 2D detections with bounding boxes. Additionally, each 2D box is provided with a pool of the most likely 6D poses for that instance. To represent a 6D pose, we parse the scores for viewpoint and in-plane rotation that have been inferred from the network and use projective properties to instantiate 6D hypotheses. In a final step, we refine each pose in every pool and select the best after verification. This last step can either be conducted in 2D or optionally in 3D if depth data is available. We present each part now in more detail.

3.1. Network architecture

Our base network is derived from a pre-trained InceptionV4 instance [27] and is fed with a color image (resized to 299 × 299) to compute feature maps at multiple scales. In order to get our first feature map of dimensionality 71 × 71 × 384, we branch off before the last pooling layer within the stem and append one 'Inception-A' block. Thereafter, we successively branch off after the 'Inception-A' blocks for a 35 × 35 × 384 feature map, after the 'Inception-B' blocks for a 17 × 17 × 1024 feature map and after the 'Inception-C' blocks for a 9 × 9 × 1536 map.[2] To cover objects at larger scale, we extend the network with two more parts. First, a 'Reduction-B' followed by two 'Inception-C' blocks to output a 5 × 5 × 1024 map. Second, one 'Reduction-B' and one 'Inception-C' to produce a 3 × 3 × 1024 map.

From here we follow the paradigm of SSD. Specifically, each of these six feature maps is convolved with prediction kernels that are supposed to regress localized detections from feature map positions. Let (w_s, h_s, c_s) be the width, height and channel depth at scale s. For each scale, we train a $3{\times}3{\times}c_s$ kernel that provides for each feature map location the scores for object ID, discrete viewpoint and in-plane rotation. Since we introduce a discretization error by this grid, we create B_s bounding boxes at each location with different aspect ratios. Additionally, we regress a refinement of their four corners. If C, V, R are the numbers of object classes, sampled viewpoints and in-plane rotations respectively, we produce a $(w_s, h_s, B_s \times (C{+}V{+}R{+}4))$ detection map for the scale s. The network has a total number of 21222 possible bounding boxes in different shapes and sizes. While this might seem high, the actual runtime of our method is remarkably low thanks to the fully-convolutional design and the good true negative behavior, which tend to yield a very confident and small set of detections. We refer to Figure 1 for a schematic overview.

[2] We changed the padding of Inception-B s.t. the next block contains a map with odd dimensionality to always contain a central position.

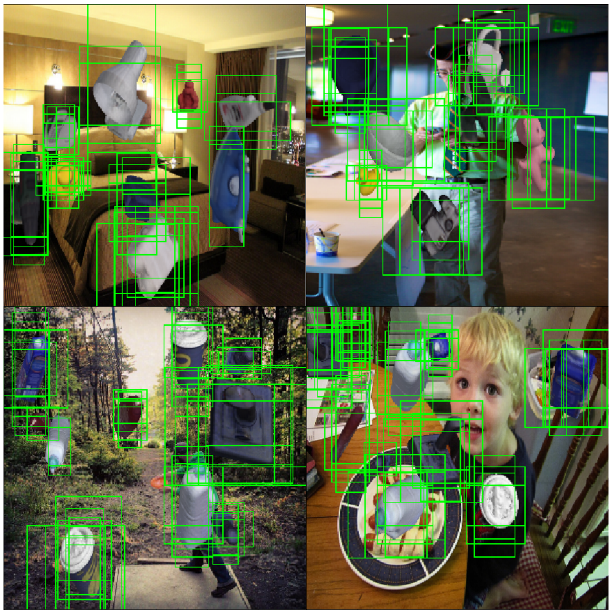


Figure 2: Exemplary training images for the datasets used. Using MS COCO images as background, we render object instances with random poses into the scene. The green boxes visualize the network's bounding boxes that have been assigned as positive samples for training.

Viewpoint scoring versus pose regression The choice of viewpoint classification over pose regression is deliberate. Although works that do direct rotation regression exist [19, 28], early experimentation showed clearly that the classification approach is more reliable for the task of detecting poses. In particular, it seems that the layers do a better job at scoring discrete viewpoints than at outputting numerically accurate translations and rotations. The decomposition of a 6D pose in viewpoint and in-plane rotation is elegant and allows us to tackle the problem more naturally. While a new viewpoint exhibits a new visual structure, an in-plane rotated view is a non-linear transformation of the same view. Furthermore, simultaneous scoring of all views allows us to parse multiple detections at a given image location, *e.g.* by accepting all viewpoints above a certain threshold. Equally important, this approach allows us to deal with symmetries or views of similar appearance in a straight-forward fashion.

3.2. Training stage

We take random images from MS COCO [20] as background and render our objects with random transformations into the scene using OpenGL commands. For each rendered instance, we compute the IoU (intersection over union) of each box with the rendered mask and every box b with IoU > 0.5 is taken as a positive sample for this object class. Additionally, we determine for the used transformation its

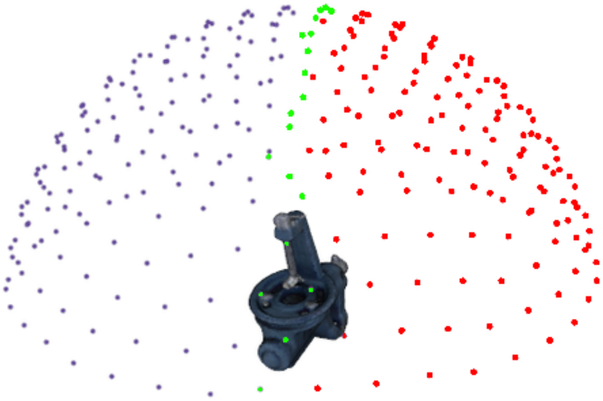


Figure 3: Discrete 6D pose space with each point representing a classifiable viewpoint. If symmetric, we use only the green points for view ID assignment during training whereas semi-symmetric objects use the red points as well.

closest sampled discrete viewpoint and in-plane rotation as well as set its four corner values to the tightest fit around the mask as a regression target. We show some training images in Figure 2.

Similar to SSD [22], we employ many different kinds of augmentation, such as changing the brightness and contrast of the image. Differently to them, though, we do not flip the images since it would lead to confusion between views and to wrong pose detections later on. We also make sure that each training image contains a 1:2 positives-negatives ratio by selecting hard negatives (unassigned boxes with high object probability) during back-propagation.

Our loss is similar to the MultiBox loss of SSD or YOLO, but we extend the formulation to take discrete views and in-plane rotations into account. Given a set of positive boxes Pos and hard-mined negative boxes Neg for a training image, we minimize the following energy:

$$L(Pos, Neg) := \sum_{b \in Neg} L_{class} + \sum_{b \in Pos} (L_{class} + \alpha L_{fit} + \beta L_{view} + \gamma L_{inplane}) \quad (1)$$

As it can be seen from (1), we sum over positive and negative boxes for class probabilities (L_{class}). Additionally, each positive box contributes weighted terms for viewpoint (L_{view}) and in-plane classification ($L_{inplane}$), as well as a fitting error of the boxes' corners (L_{fit}). For the classification terms, i.e., L_{class}, L_{view}, $L_{inplane}$, we employ a standard softmax cross-entropy loss, whereas a more robust smooth L1-norm is used for corner regression (L_{fit}).

Dealing with symmetry and view ambiguity Our approach demands the elimination of viewpoint confusion for proper convergence. We thus have to treat symmetrical or semi-symmetrical (constructible with plane reflection) objects with special care. Given an equidistantly-sampled sphere from which we take our viewpoints, we discard positions that lead to ambiguity. For symmetric objects, we solely sample views along an arc, whereas for semi-symmetric objects we omit one hemisphere entirely. This approach easily generalizes to cope with views which are mutually indistinguishable although this might require manual annotation for specific objects in practice. In essence, we simply ignore certain views from the output of the convolutional classifiers during testing and take special care of viewpoint assignment in training. We refer to Figure 3 for a visualization of the pose space.

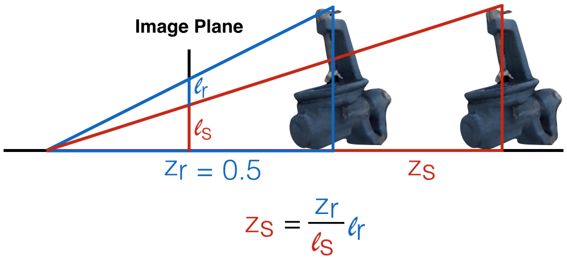


Figure 4: For each object we precomputed the perfect bounding box and the 2D object centroid with respect to each possible discrete rotation in a prior offline stage. To this end, we rendered the object at a canonical centroid distance $z_r = 0.5m$. Subsequently, the object distance z_s can be inferred from the projective ratio according to $z_s = \frac{l_r}{l_s} z_r$, where l_r denotes diagonal length of the precomputed bounding box and l_s denotes the diagonal length of the predicted bounding box on the image plane.

3.3. Detection stage

We run a forward-pass on the input image to collect all detections above a certain threshold, followed by non-maximum suppression. This yields refined and tight 2D bounding boxes with an associated object ID and scores for all views and in-plane rotations. For each detected 2D box we thus parse the most confident views as well as in-plane rotations to build a pool of 6D hypotheses from which we select the best after refinement. See Figure 5 for the pooled hypotheses and Figure 6 for the final output.

3.3.1 From 2D bounding box to 6D hypothesis

So far, all computation has been conducted on the image plane and we need to find a way to hypothesize 6D poses from our network output. We can easily construct a 3D rotation, given view ID and in-plane rotation ID, and can use the bounding box to infer 3D translation. To this end, we

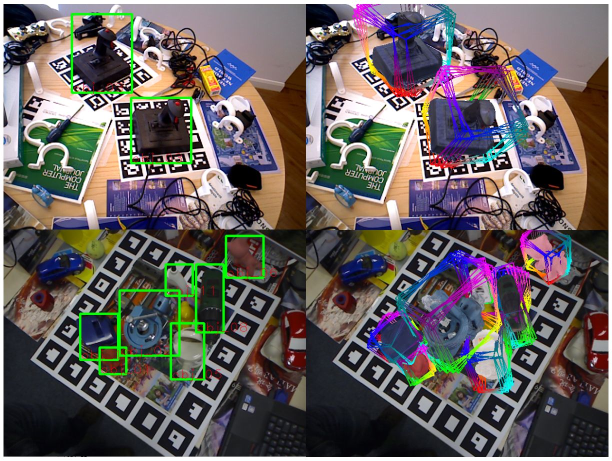


Figure 5: Prediction output and 6D pose pooling of our network on the Tejani dataset and the multi-object dataset. Each 2D prediction builds a pool of 6D poses by parsing the most confident views and in-plane rotations. Since our networks are trained with various augmentations, they can adapt to different global illumination settings.

render all possible combinations of discrete views and in-plane rotations at a canonical centroid distance $z_r = 0.5m$ in an offline stage and compute their bounding boxes. Given the diagonal length l_r of the bounding box during this offline stage and the one predicted by the network l_r, we can infer the object distance $z_s = \frac{l_r}{l_s} z_r$ from their projective ratio, as illustrated in Figure 4. In a similar fashion, we can derive the projected centroid position and back-project to a 3D point with known camera intrinsics.

3.3.2 Pose refinement and verification

The obtained poses are already quite accurate, yet can in general benefit from a further refinement. Since we will regard the problem for both RGB and RGB-D data, the pose refinement will either be done with an edge-based or cloud-based ICP approach. If using RGB only, we render each hypothesis into the scene and extract a sparse set of 3D contour points. Each 3D point X_i, projected to $\pi(X_i) = x_i$, then shoots a ray perpendicular to its orientation to find the closest scene edge y_i. We seek the best alignment of the 3D model such that the average projected error is minimal:

$$\arg\min_{R,t} \sum_i \left(||\pi(R \cdot X_i + t) - y_i||^2 \right). \qquad (2)$$

We minimize this energy with an IRLS approach (similar to [8]) and robustify it using Geman-McLure weighting. In the case of RGB-D, we render the current pose and solve with standard projective ICP with a point-to-plane formulation in closed form [2]. In both cases, we run multiple rounds of correspondence search to improve refinement and we use multi-threading to accelerate the process.

The above procedure provides multiple refined poses for each 2D box and we need to choose the best one. To this end, we employ a verification procedure. Using only RGB, we do a final rendering and compute the average deviation of orientation between contour gradients and overlapping scene gradients via absolute dot products. In case RGB-D data is available, we render the hypotheses and estimate camera-space normals to measure the similarity again with absolute dot products.

4. Evaluation

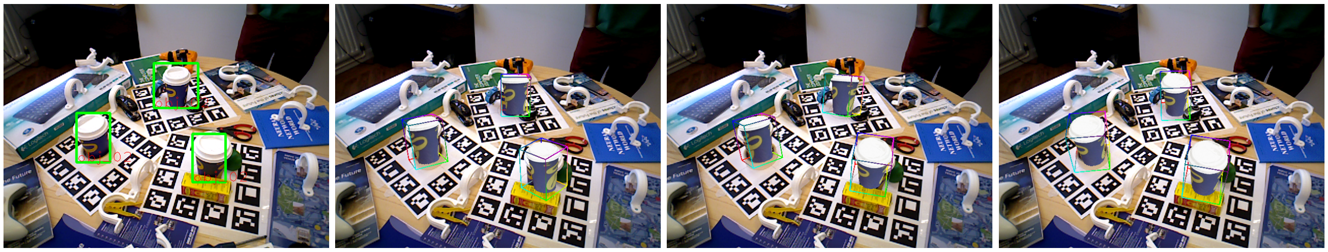
We implemented our method in C++ using TensorFlow 1.0 [1] and cuDNN 5 and ran it on a i7-5820K@3.3GHz with an NVIDIA GTX 1080. Our evaluation has been conducted on three datasets. The first, presented in Tejani et al. [29], consists of six sequences where each sequence requires the detection and pose estimation of multiple instances of the same object in clutter and with different levels of mild occlusion. The second dataset, presented in [14], consists of 15 sequences where each frame presents one instance to detect and the main challenge is the high amount of clutter in the scene. As others, we will skip two sequences since they lack a meshed model. The third dataset, presented in [4] is an extension of the second where one sequence has been annotated with instances of multiple objects undergoing heavy occlusions at times.

Network configuration and training To get the best results it is necessary to find an appropriate sampling of the model view space. If the sampling is too coarse we either miss an object in certain poses or build suboptimal 6D hypotheses whereas a very fine sampling can lead to a more difficult training. We found an equidistant sampling of the unit sphere into 642 views to work well in practice. Since the datasets only exhibit the upper hemisphere of the objects, we ended up with 337 possible view IDs. Additionally, we sampled the in-plane rotations from -45 to 45 degrees in steps of 5 to have a total of 19 bins.

Given the above configuration, we trained the last layers of the network and the predictor kernels using ADAM and a constant learning rate of 0.0003 until we saw convergence on a synthetic validation set. The balancing of the loss term weights proved to be vital to provide both good detections and poses. After multiple trials we determined $\alpha = 1.5$, $\beta = 2.5$ and $\gamma = 1.5$ to work well for us. We refer the reader to the supplementary material to see the error development for different configurations.

4.1. Single object scenario

Since 3D detection is a multi-stage pipeline for us, we first evaluate purely the 2D detection performance between



(a) 2D Detections (b) Unrefined (c) RGB refinement (d) RGB-D refinement

Figure 6: After predicting 2D detections (a), we build 6D hypotheses and run pose refinement and a final verification. While the unrefined poses (b) are rather approximate, contour-based refinement (c) produces already visually acceptable results. Occlusion-aware projective ICP with cloud data (d) leads to a very accurate alignment.

Sequence	LineMOD [12]	LC-HF [29]	Kehl [17]	Us
Camera	0.589	0.394	0.383	**0.741**
Coffee	0.942	0.891	0.972	**0.983**
Joystick	0.846	0.549	0.892	**0.997**
Juice	0.595	0.883	0.866	**0.919**
Milk	0.558	0.397	0.463	**0.780**
Shampoo	**0.922**	0.792	0.910	0.892
Total	0.740	0.651	0.747	**0.885**

Table 1: F1-scores on the re-annotated version of [29]. Although our method is the only one to solely use RGB data, our results are considerably higher than all related works.

our predicted boxes and the tight bounding boxes of the rendered groundtruth instances on the first two datasets. Note that we always conduct proper detection and not localization, *i.e.* we do not constrain the maximum number of allowed detections but instead accept all predictions above a chosen threshold. We count a detection to be correct when the IoU score of a predicted bounding box with the groundtruth box is higher than 0.5. We present our F1-scores in Tables 1 and 2 for different detection thresholds.

It is important to mention that the compared methods, which all use RGB-D data, allow a detection to survive after rigorous color- and depth-based checks whereas we use simple thresholding for each prediction. Therefore, it is easier for them to suppress false positives to increase their precision whereas our confidence comes from color cues only.

On the Tejani dataset we outperform all related RGB-D methods by a huge margin of 13.8% while using color only. We analyzed the detection quality on the two most difficult sequences. The 'camera' has instances of smaller scale which are partially occluded and therefore simply missed whereas the 'milk' sequence exhibits stronger occlusions in virtually every frame. Although we were able to detect the 'milk' instances, our predictors could not overcome the occlusions and regressed wrongly-sized boxes which were not tight enough to satisfy the IoU threshold. These were counted as false positives and thus lowered our recall[3].

On the second dataset we have mixed results where we can outperform state-of-the-art RGB-D methods on some sequences while being worse on others. For larger feature-rich objects like 'benchvise', 'iron' or 'driller' our method performs better than the related work since our network can draw from color and textural information. For some objects, such as 'lamp' or 'cam', the performance is worse than the related work. Our method relies on color information only and thus requires a certain color similarity between synthetic renderings of the CAD model and their appearance in the scene. Some objects exhibit specular effects (*i.e.* changing colors for different camera positions) or the frames can undergo sensor-side changes of exposure or white balancing, causing a color shift. Brachmann et al. [5] avoid this problem by training on a well-distributed subset of real sequence images. Our problem is much harder since we train on synthetic data only and must generalize to real, unseen imagery.

Our performance for objects of smaller scale such as 'ape', 'duck' and 'cat' is worse and we observed a drop both in recall and precision. We attribute the lower recall to our bounding box placement, which can have 'blind spots' at some locations and consequently, leading to situations where a small-scale instance cannot be covered sufficiently by any box to fire. The lower precision, on the other hand, stems from the fact that these objects are textureless and of uniform color which increases confusion with the heavy scene clutter.

4.1.1 Pose estimation

We chose for each object the threshold that yielded the highest F1-score and run all following pose estimation experiments with this setting. We are interested in the pose accuracy for all correctly detected instances.

[3] We refer to the supplement for more detailed graphs.

	ape	bvise	cam	can	cat	driller	duck	box	glue	holep	iron	lamp	phone
Our method	76.3	**97.1**	92.2	**93.1**	89.3	**97.8**	80.0	93.6	**76.3**	71.6	**98.2**	93.0	**92.4**
Kehl [17]	**98.1**	94.8	**93.4**	82.6	**98.1**	96.5	**97.9**	**100**	74.1	**97.9**	91.0	**98.2**	84.9
LineMOD [14]	53.3	84.6	64.0	51.2	65.6	69.1	58.0	86.0	43.8	51.6	68.3	67.5	56.3
LC-HF [29]	85.5	96.1	71.8	70.9	88.8	90.5	90.7	74.0	67.8	87.5	73.5	92.1	72.8

Table 2: F1-scores for each sequence of [14]. Note that the LineMOD scores are supplied from [29] with their evaluation since [14] does not provide them. Using color only we can easily compete with the other RGB-D based methods.

Sequence	IoU-2D	IoU-3D	VSS-2D	VSS-3D
Camera	0.973	0.904	0.693	0.778
Coffee	0.998	0.996	0.765	0.931
Joystick	1	0.953	0.655	0.866
Juice	0.994	0.962	0.742	0.865
Milk	0.970	0.990	0.722	0.810
Shampoo	0.993	0.974	0.767	0.874
Total	0.988	0.963	0.724	0.854

Table 3: Average pose errors for the Tejani dataset.

	RGB		
	Ours	Brachmann 2016 [5]	LineMOD [13]
IoU	99.4 %	97.5%	86.5%
ADD [12]	76.3%	50.2%	24.2%

	RGB-D		
	Ours	Brachmann 2016 [5]	Brachmann 2014 [4]
IoU	96.5 %	99.6%	99.1%
ADD [12]	90.9%	99.0%	97.4%

Table 4: Average pose errors for the LineMOD dataset.

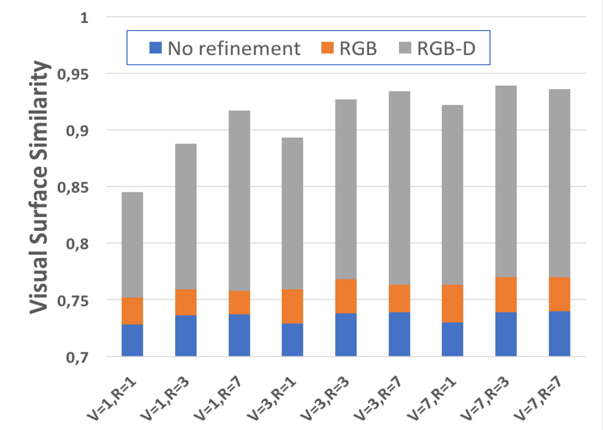


Figure 7: Average VSS scores for the 'coffee' object for different numbers of parsed views and in-plane rotations as well as different pose refinement options.

Error metrics To measure 2D pose errors we will compute both an IoU score and a Visual Surface Similarity (VSS) [15]. The former is different than the detection IoU check since it measures the overlap of the rendered masks' bounding boxes between groundtruth and final pose estimate and accepts a pose if the overlap is larger than 0.5. VSS is a tighter measure since it counts the average pixel-wise overlap of the mask. This measure assesses well the suitability for AR applications and has the advantage of being agnostic towards the symmetry of objects. To measure the 3D pose error we use the ADD score from [14]. This assesses the accuracy for manipulation tasks by measuring the average deviation between transformed model point clouds of groundtruth and hypothesis. If it is smaller than $\frac{1}{10}th$ of the model diameter, it is counted as a correct pose.

Refinement with different parsing values As mentioned, we parse the most confident views and in-plane rotations to build a pool of 6D hypotheses for each 2D detection. Here, we want to assess the final pose accuracy when changing the number of parsed views V and rotations R for different refinement strategies We present in Figure 7 the results on Tejani's 'coffee' sequence for the cases of no refinement, edge-based and cloud-based refinement (see Figure 6 for an example). To decide for the best pose we employ verification over contours for the first two cases and normals for the latter. As can be seen, the final poses without any refinement are imperfect but usually provide very good initializations for further processing. Additional 2D refinement yields better poses but cannot cope well with occluders whereas depth-based refinement leads to perfect poses in practice. The figure gives also insight for varying V and R for hypothesis pool creation. Naturally, with higher numbers the chances of finding a more accurate pose improve since we evaluate a larger portion of the 6D space. It is evident, however, that every additional parsed view V gives a larger benefit than taking more in-plane rotations R into the pool. We explain this by the fact that our viewpoint sampling is coarser than our in-plane sampling and thus reveals more uncovered pose space when parsed, which in turn helps especially depth-based refinement. Since we create a pool of $V \cdot R$ poses for each 2D detection, we fixed $V = 3, R = 3$ for all experiments as a compromise between accuracy and refinement runtime.

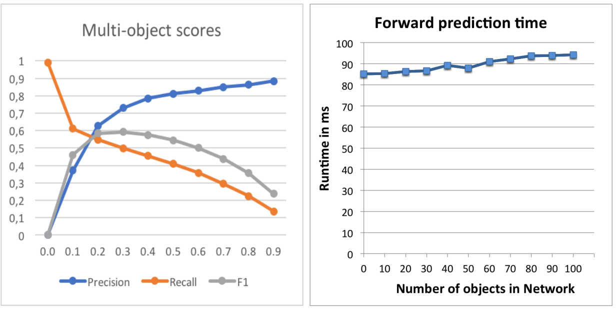


Figure 8: Left: Detection scores on the multi-object dataset for a different global threshold. Right: Runtime increase for the network prediction with an increased number of objects.

Performance on the two datasets We present our pose errors in Tables 3 and 4 after 2D and 3D refinement. Note that we do not compute the ADD scores for Tejani since each object is of (semi-)symmetric nature, leading always to near-perfect ADD scores of 1. The poses are visually accurate after 2D refinement and furthermore are boosted by an additional depth-based refinement stage. On the second dataset we are actually able to come very close to Brachmann et al. which is surprising since they have a huge advantage of real data training. For the case of pure RGB-based poses, we can even overtake their results. We provide more detailed error tables in the supplement.

4.2. Multiple object detection

The last dataset has annotations for 9 out of the 15 objects and is quite difficult since many instances undergo heavy occlusion. Different to the single object scenario, we have now a network with one global detection threshold for all objects and we present our scores in Figure 8 when varying this threshold. Brachmann et al. [5] can report an impressive Average Precision (AP) of 0.51 whereas we can report an AP of 0.38. It can be observed that our method degrades gracefully as the recall does not drop suddenly from one threshold step to the next. Note again that Brachmann et al. have the advantage of training on real images of the sequence whereas we must detect heavily-occluded objects from synthetic training only.

4.3. Runtime and scalability

For a single object in the database, Kehl et al. [17] report a runtime of around 650ms per frame whereas Brachmann et al. [4, 5] report around 450ms. Above methods are scalable and thus have a sublinear runtime growth with an increasing database size. Our method is a lot faster than the related work while being scalable as well. In particular, we can report a runtime of approximately 85ms for a single object. We show our prediction times in Figure 8 which reveals that we scale very well with an increasing number of objects in the network. While the prediction is fast, our pose refinement takes more time since we need to refine every pose of each pool. On average, given that we have about 3 to 5 positive detections per frame, we need a total of an additional 24ms for refinement, leading to a total runtime of around 10Hz.

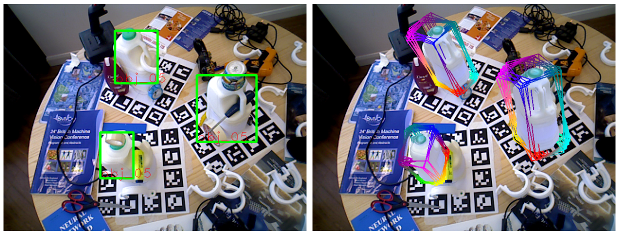


Figure 9: One failure case where incorrect bounding box regression, induced by occlusion, led to wrong 6D hypothesis creation. In such cases a subsequent refinement cannot always recover the correct pose anymore.

4.4. Failure cases

The most prominent issue is the difference in colors between synthetic model and scene appearance, also including local illumination changes such as specular reflections. In these cases, the object confidence might fall under the detection threshold since the difference between the synthetic and the real domain is too large. A more advanced augmentation would be needed to successfully tackle this problem. Another possible problem can stem from the bounding box regression. If the regressed corners are not providing a tight fit, it can lead to translations that are too offset during 6D pose construction. An example of this problem can be seen in Figure 9 where the occluded milk produces wrong offsets. We also observed that small objects are sometimes difficult to detect which is even more true after resizing the input to 299 × 299. Again, designing a more robust training as well as a larger network input could be of benefit here.

Conclusion

To our knowledge, we are the first to present an SSD-style detector for 3D instance detection and full 6D pose estimation that is trained on synthetic model information. We have shown that color-based detectors are indeed able to match and surpass current state-of-the-art methods that leverage RGB-D data while being around one order of magnitude faster. Future work should include a higher robustness towards color deviation between CAD model and scene appearance. Avoiding the problem of proper loss term balancing is also an interesting direction for future research.

Acknowledgments We would like to thank Toyota Motor Corporation for funding and supporting this work.

References

[1] M. Abadi, P. Barham, J. Chen, Z. Chen, A. Davis, J. Dean, M. Devin, S. Ghemawat, G. Irving, M. Isard, M. Kudlur, J. Levenberg, R. Monga, S. Moore, D. Murray, B. Steiner, P. Tucker, V. Vasudevan, P. Warden, M. Wicke, Y. Yu, and X. Zheng. TensorFlow: Large-scale machine learning on heterogeneous systems. In *OSDI*, 2016. 5

[2] P. Besl and N. McKay. A Method for Registration of 3-D Shapes. *TPAMI*, 1992. 5

[3] T. Birdal and S. Ilic. Point Pair Features Based Object Detection and Pose Estimation Revisited. In *3DV*, 2015. 2

[4] E. Brachmann, A. Krull, F. Michel, S. Gumhold, J. Shotton, and C. Rother. Learning 6D Object Pose Estimation using 3D Object Coordinates. In *ECCV*, 2014. 1, 2, 5, 7, 8

[5] E. Brachmann, F. Michel, A. Krull, M. Y. Yang, S. Gumhold, and C. Rother. Uncertainty-Driven 6D Pose Estimation of Objects and Scenes from a Single RGB Image. In *CVPR*, 2016. 1, 6, 7, 8

[6] A. Doumanoglou, R. Kouskouridas, S. Malassiotis, and T.-K. Kim. 6D Object Detection and Next-Best-View Prediction in the Crowd. In *CVPR*, 2016. 1, 2

[7] B. Drost, M. Ulrich, N. Navab, and S. Ilic. Model globally, match locally: efficient and robust 3D object recognition. In *CVPR*, 2010. 2

[8] T. Drummond and R. Cipolla. Real-time visual tracking of complex structures. *TPAMI*, 2002. 5

[9] M. Everingham, S. M. A. Eslami, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman. The Pascal Visual Object Classes Challenge: A Retrospective. *IJCV*, 2014. 1

[10] R. Girshick. Fast R-CNN. *arXiv:1504.08083*, 2015. 2

[11] K. He, X. Zhang, S. Ren, and J. Sun. Spatial Pyramid Pooling in Deep Convolutional Networks for Visual Recognition. *TPAMI*, 2015. 2

[12] 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. *TPAMI*, 2012. 1, 6, 7

[13] 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 *ICCV*, 2011. 7

[14] S. Hinterstoisser, V. Lepetit, S. Ilic, S. Holzer, G. Bradsky, K. Konolige, and N. Navab. Model Based Training, Detection and Pose Estimation of Texture-Less 3D Objects in Heavily Cluttered Scenes. In *ACCV*, 2012. 1, 2, 5, 7

[15] T. Hodan, J. Matas, and S. Obdrzalek. On Evaluation of 6D Object Pose Estimation. In *ECCV Workshop*, 2016. 7

[16] T. Hodan, X. Zabulis, M. Lourakis, S. Obdrzalek, and J. Matas. Detection and Fine 3D Pose Estimation of Textureless Objects in RGB-D Images. In *IROS*, 2015. 1, 2

[17] 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 *ECCV*, 2016. 1, 2, 6, 7, 8

[18] W. Kehl, F. Tombari, N. Navab, S. Ilic, and V. Lepetit. Hashmod: A Hashing Method for Scalable 3D Object Detection. In *BMVC*, 2015. 1, 2

[19] A. Kendall, M. Grimes, and R. Cipolla. PoseNet: A Convolutional Network for Real-Time 6-DOF Camera Relocalization. In *ICCV*, 2015. 3

[20] G. Lin, C. Shen, Q. Shi, A. V. D. Hengel, and D. Suter. Fast Supervised Hashing with Decision Trees for High-Dimensional Data. In *CVPR*, 2014. 3

[21] T.-Y. Lin, P. Dollár, R. Girshick, K. He, B. Hariharan, and S. Belongie. Feature Pyramid Networks for Object Detection. In *arXiv:1612.03144*, 2016. 2

[22] W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. Reed, C.-y. Fu, and A. C. Berg. SSD : Single Shot MultiBox Detector. In *ECCV*, 2016. 1, 2, 4

[23] A. Mousavian, D. Anguelov, J. Flynn, and J. Kosecka. 3D Bounding Box Estimation Using Deep Learning and Geometry. *arXiv:1612.00496*, 2016. 2

[24] P. Poirson, P. Ammirato, C.-Y. Fu, W. Liu, J. Kosecka, and A. C. Berg. Fast Single Shot Detection and Pose Estimation. In *3DV*, 2016. 2

[25] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi. You only look once: Unified, real-time object detection. In *CVPR*, 2016. 1, 2

[26] 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. *IJCV*, 2015. 1

[27] C. Szegedy, S. Ioffe, and V. Vanhoucke. Inception-v4, Inception-ResNet and the Impact of Residual Connections on Learning. *arxiv:1602.07261*, 2016. 3

[28] D. J. Tan, F. Tombari, S. Ilic, and N. Navab. A Versatile Learning-based 3D Temporal Tracker : Scalable , Robust , Online. In *ICCV*, 2015. 3

[29] A. Tejani, D. Tang, R. Kouskouridas, and T.-k. Kim. Latent-class hough forests for 3D object detection and pose estimation. In *ECCV*, 2014. 1, 2, 5, 6, 7

[30] F. Tombari, S. Salti, and L. Di Stefano. Unique signatures of histograms for local surface description. In *ECCV*, 2010. 2

[31] M. Ulrich, C. Wiedemann, and C. Steger. Combining scale-space and similarity-based aspect graphs for fast 3D object recognition. *TPAMI*, 2012. 1

[32] K. M. Yi, E. Trulls, V. Lepetit, and P. Fua. LIFT: Learned invariant feature transform. In *ECCV*, 2016. 2

5.1.2 Deep Model-based 6D Pose Refinement in RGB (ECCV Oral 2018)

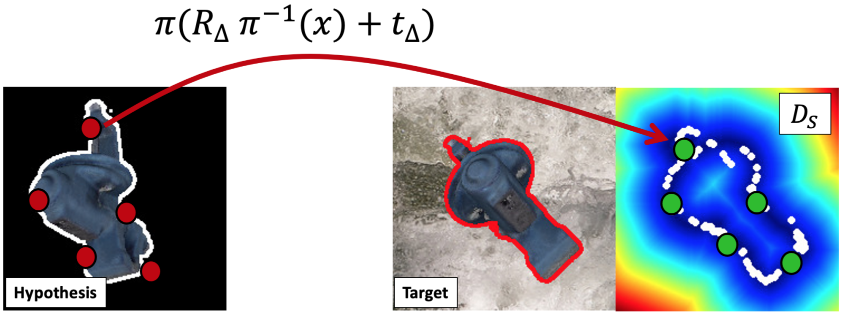


Figure 5.2. **Visual Proxy Loss for 6D Pose Refinement.** Inspired by ideas from contour tracking, we refine the 6D pose by means of contour alignment in 2D. Therefore, we sample contour points on the rendered hypothesis, which we then backproject to 3D to apply our predicted pose update. Subsequently, we project these points onto the distance transform of the target contour to enforce an optimal alignment between both contours.

During evaluation of SSD-6D, we realized that we can compete with state-of-the-art harnessing depth data, however, our performance still significantly improved when additionally employing ICP [187] refinement. Moreover, after [9] and [10], a lot of new methods for monocular 6D pose estimation were proposed [124, 120]. Nevertheless, yet again all these methods gained significantly in performance when employing ICP.

Thus, in this work we anticipated to close the gap between RGB and RGB-D based methods. In particular, using deep learning we learn object pose refinement/tracking from RGB data alone, leveraging a new fully differentiable visual proxy loss (*c.f.* Figure 5.2). Inspired by ideas from perspective edge tracking [188, 189, 190], we directly align contours in 2D on the basis of the predicted 3D update rotation and translation. As collecting appropriate training data is very labor intensive, we decided to fully rely on synthetic samples. Therefore, we sample a random pose R, t and a random pose perturbation $\bar{R}_\Delta, \bar{t}_\Delta$. Afterwards, we render the associated image for the scene $\mathcal{S} = \mathcal{R}(R, t)$ and pose hypothesis $\mathcal{H} = \mathcal{R}(\bar{R}_\Delta^{-1}R, t - \bar{t}_\Delta)$ on top of images taken from ImageNet [53]. Our deep network is then fed with the renderings $\mathcal{S}$ and $\mathcal{H}$ and runs the first five, on ImageNet pre-trained and frozen, InceptionV4 [57] blocks to extract low level features from $\mathcal{S}$ and $\mathcal{H}$. Pre-training was crucial as the early layers are mostly responsible for extracting low-level features, suffering the most from the synthetic-to-real domain gap. The features from both images are then concatenated and fed through a very shallow network to estimate the perturbation in rotation $\hat{R}_\Delta$ and translation $\hat{t}_\Delta$. During training we then sample 2D contour points $V_\mathcal{H}$ from the rendered depth map of the hypothesis and warp them onto the distance transform $\mathcal{D}_\mathcal{S}$ of the scene contours to compute our visual proxy loss as

$$\mathcal{L}(\hat{R}_\Delta, \hat{t}_\Delta, \mathcal{D}_\mathcal{S}, V_\mathcal{H}) = \sum_{v \in V_\mathcal{H}} = \mathcal{D}_\mathcal{S}\left[\pi(\hat{R}_\Delta \pi^{-1}(v) + \hat{t}_\Delta)\right] \tag{5.2}$$

Thereby, $\pi(\cdot)$ and $\pi^{-1}(\cdot)$ denote the perspective projection and backprojection, respectively. Further, note that we utilize quaternions $\hat{q}$ to represent $\hat{R} \in SO(3)$ (*c.f.* Section 3.1). Minimizing the above loss encourages a step towards the 0-level set of the distance transform. We basically tune the network weights to rotate and translate the object in 6D so to maximize the projected contours overlap.

Noteworthy, our new formulation avoids typical pitfalls of hand-crafted methods such as tedious hyper-parameter tuning. In addition, our approach can handle ambiguities induced by symmetries and does not depend on hyptertuning the individual pose components using $\text{argmin}_{\hat{q}_\Delta,\hat{t}_\Delta} ||\bar{q}_\Delta - \frac{\hat{q}_\Delta}{||\hat{q}_\Delta||_2}||_2 + \gamma \cdot ||\bar{t}_\Delta - \hat{t}_\Delta||_2$ as in [191]. Extensive experiments on LM [131] and IC-MI [138] demonstrate that our approach can surpass all other related RGB methods [189], does not suffer from ambiguities [191], and is almost on par with ICP refinement. In fact, we outperform ICP for five degrees-of-freedom, however, due to nature of the perspective projection, estimating z is particularly challenging. Nevertheless, our method is still very applicable to the domains of AR and robotics.

Contributions. I proposed and implemented our method using deep learning to learn monocular 6D pose refinement using a projective proxy loss. I also conducted our evaluations on LM(-O), IC-MI and Choi. Wadim Kehl extended the loss formulation to sample contour points. He also implemented the related work to enable proper comparison with state-of-the-art and ran the respective experiments.

Deep Model-Based 6D Pose Refinement in RGB

Fabian Manhardt[1,*], Wadim Kehl[2,*], Nassir Navab[2], Federico Tombari[1]

[1] Technical University of Munich
[2] Toyota Research Institute, Los Altos
[*] Equal Contribution

It is the accepted but not the published version of the paper due to copyright restrictions.

Published version: https://doi.org/10.1007/978-3-030-01264-9_49

Deep Model-Based 6D Pose Refinement in RGB

Fabian Manhardt* Wadim Kehl* Nassir Navab Federico Tombari

Technical University of Munich Toyota Research Institute
{fabian.manhardt, nassir.navab}@tum.de wadim.kehl@tri.global
tombari@in.tum.de

Abstract. We present a novel approach for model-based 6D pose refinement in color data. Building on the established idea of contour-based pose tracking, we teach a deep neural network to predict a translational and rotational update. At the core, we propose a new visual loss that drives the pose update by aligning object contours, thus avoiding the definition of any explicit appearance model. In contrast to previous work our method is correspondence-free, segmentation-free, can handle occlusion and is agnostic to geometrical symmetry as well as visual ambiguities. Additionally, we observe a strong robustness towards rough initialization. The approach can run in real-time and produces pose accuracies that come close to 3D ICP without the need for depth data. Furthermore, our networks are trained from purely synthetic data and will be published together with the refinement code to ensure reproducibility.[1]

Keywords: Pose Estimation, Pose Refinement, Tracking

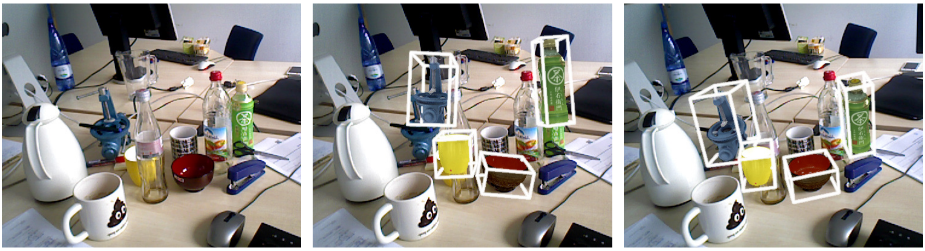
1 Introduction

The problem of tracking CAD models in images is frequently encountered in contexts such as robotics, augmented reality (AR) and medical procedures. Usually, tracking has to be carried out in the full 6D pose, i.e. one seeks to retrieve both the 3D metric translation as well as the 3D rotation of the object in each frame. Another typical scenario is pose refinement, where an object detector provides a rough 6D pose estimate, which has to be corrected in order to provide a better fit (Figure 1). The usual difficulties that arise include viewpoint ambiguities, occlusions, illumination changes and differences in appearance between the model and the object in the scene. Furthermore, for tracking applications the method should also be fast enough to cover large inter-frame motions.

Most related work based on RGB data can be roughly divided into sparse and region-based methods. The former methods try to establish local correspondences between frames [1, 2] and work well for textured objects, whereas latter

[1] http://campar.in.tum.de/Main/FabianManhardt.

* The first two authors contributed equally to this work.



(a) Input image (b) Initial pose hypotheses (c) Poses after 10 iterations

Fig. 1: Exemplary illustration of our method. While a) depicts an input RGB frame, b) shows our four initial 6D pose hypotheses. For each obtained frame we refine each pose for a better fit to the scene. In d) we show the final results after convergence. Note the rough pose initializations as well as the varying amount of occlusion the objects of interest undergo.

ones exploit more holistic information about the object such as shape, contour or color [3–6] and are usually better suited for texture-less objects. It is worth mentioning that mixtures of the two sets of methods have been proposed as well [7–10]. Recently, methods that use only depth [11] or both modalities [12–14] have shown that depth can make tracking more robust by providing more clues about occlusion and scale.

This work aims to explore how RGB information alone can be sufficient to perform visual tasks such as 3D tracking and 6-Degree-of-Freedom (6DoF) pose refinement by means of a Convolutional Neural Network (CNN). While this has already been proposed for camera pose and motion estimation [15–18], it has not been well-studied for the problem at hand.

As a major contribution we provide a differentiable formulation of a new visual loss that aligns object contours and implicitly optimizes for metric translation and rotation. While our optimization is inspired by region-based approaches, we can track objects of any texture or shape since we do not need to model global [3, 5, 13] or local appearance [19, 6]. Instead, we show that we can do away with these hand-crafted approaches by letting the network learn the object appearance implicitly. We teach the CNN to align contours between synthetic object renderings and scene images under changing illumination and occlusions and show that our approach can deal with a variety of shapes and textures. Additionally, our method allows to deal with geometrical symmetries and visual ambiguities without manual tweaking and is able to recover correct poses from very rough initializations.

Notably, our formulation is parameter-free and avoids typical pitfalls of handcrafted tracking or refinement methods (e.g. via segmentation or correspondences + RANSAC) that require tedious tuning to work well in practice. Furthermore, like with depth-based approaches such as ICP, we are robust to occlusion and produce results which come close to RGB-D methods without the need for depth data, making it thus very applicable to the domains of AR, medical and robotics.

2 Related work

Since the field of tracking and pose refinement is vast, we will only focus here on works that deal with CAD models in RGB data. Early methods in this field used either 2D-3D correspondences [20, 7] or 3D edges [21–23] and fit the model in an ICP fashion with iterative, projective update steps. Successive methods in this direction managed to obtain improved performance [8, 9]. Additionally, other works focused on tracking the contour densely via level-sets [24, 4].

Based on these works, [3] presented a new approach that follows the projected model contours to estimate the 6D pose update. In a follow-up work [25], the authors extended their method to simultaneously track and reconstruct a 3D object on a mobile phone in real-time. The authors from [5] improved the convergence behavior with a new optimization scheme and presented a real-time implementation on a GPU. Consequently, [6] showed how to improve the color segmentation by using local color histograms over time. Orthogonally, the work [13] approximates the model pose space to avoid GPU computations and enables real-time performance on a single CPU core. All these approaches share the property that they rely on hand-crafted segmentation methods that can fail in the case of sudden appearance changes or occlusion. We instead want to entirely avoid hand-crafting manual appearance descriptions.

Another set of works tries to combine learning with simultaneous detection and pose estimation in RGB. The method presented in [26] couples the SSD paradigm [27] with pose estimation to produce 6D pose pools per instance which are then refined with edge-based ICP. On the contrary, the approach from [28] uses auto-context Random Forests to regress object coordinates in the scene that are used to estimate poses. In [29] a method is presented that instead regresses the projected 3D bounding box and recovers the pose from these 2D-3D correspondences whereas the authors in [30] infer keypoint heatmaps that are then used for 6D pose computation. Similarly, the 3D Interpreter Network [31] infers heatmaps for categories and regresses projection and deformation to align synthetic with real imagery. In the work [14], a deep learning approach is used to track models in RGB-D data. Their work goes along similar grounds but we differ in multiple ways including data generation, energy formulation and their use of RGB-D data. In particular, we show that a naive formulation of pose regression does not work in the case of symmetry which is often the case for man-made objects.

We also find common ground with Spatial Transformer Networks in 2D [32] and especially 3D [33], where the employed network architecture contains a submodule to transform the 2D/3D input via a regressed affine transformation on a discrete lattice. Our network instead regresses a rigid body motion on a set of continuous 3D points to minimize the visual error.

3 Methodology

In this section we explain our approach to train a CNN to regress a 6D pose refinement from RGB information alone. We design the problem in such a way

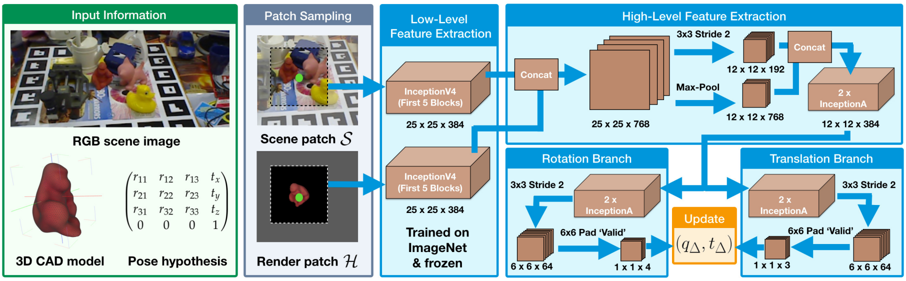


Fig. 2: Schematic overview of the full pipeline. Given input image and pose hypothesis (R, t), we render the object, compute the center of the bounding box of the hypothesis (green point) and then cut out a scene patch $\mathcal{S}$ and a render patch $\mathcal{H}$. We resize both to 224x224 and feed them separately into pre-trained InceptionV4 layers to extract low-level features. Thereafter, we concatenate and compute high-level features before diverging into separate branches. Eventually, we retrieve our pose update as 3D translation and normalized 4D quaternion.

that we supply two color patches ($\mathcal{S}$ and $\mathcal{H}$) to the network in order to infer a translational and rotational update. In Figure 2 we depict our pipeline and show a typical scenario where we have a 6D hypothesis (coming from a detector or tracker) that is not correctly aligned. We want to estimate a refinement such that eventually the updated hypothesis overlaps perfectly with the real object.

3.1 Input patch sampling

We first want to discuss our patch extraction strategy. Provided a CAD model and a 6D pose estimate (R, t) in camera space, we create a rendering and compute the center of the associated bounding box of the hypothesis around which we subsequently extract $\mathcal{S}$ and $\mathcal{H}$. Since different objects have varying sizes and shapes it is important to adapt the cropping size to the spatial properties of the specific object. The most straightforward method would be to simply crop $\mathcal{S}$ and $\mathcal{H}$ with respect to a tight 2D bounding box of the rendered mask. However, when employing such metric crops, the network loses the ability to robustly predict an update along the Z-axis: indeed, since each crop would almost entirely fill out the input patch, no estimate of the difference in depth can be drawn. Due to this, we explicitly calculate the spatial extent in pixels at a minimum metric distance (with some added padding) and use this as a fixed-size 'window' into our scene. In particular, prior to training, we render the object from various different viewpoints, compute their bounding boxes, and take the maximum width or height of all produced bounding boxes.

3.2 Training stage

To create training data we randomly sample a ground truth pose (R^*, t^*) of the object in camera coordinates and render the object with that pose onto a random background to create a scene image. To learn pose refinement, we perturb the true pose to get a noisy version (R, t) and render a hypothesis image. Given those two images, we cut out patches $\mathcal{S}$ and $\mathcal{H}$ with the strategy mentioned above.

The naive approach Provided these patches, we now want to infer a separate correction (R_Δ, t_Δ) of the perturbed pose (R, t) such that

$$R^* = R_\Delta \cdot R \quad , \quad t^* = t + t_\Delta . \tag{1}$$

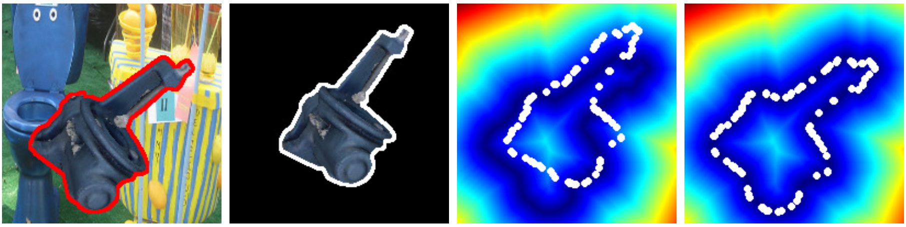
Due to the difficulty of optimizing in SO(3) we parametrize via unit quaternions q^*, q, q_Δ to define a regression problem, i.e. similar to what [34] proposed for camera localization or [14] for model pose tracking:

$$\min_{q_\Delta, t_\Delta} ||q^* - \frac{q_\Delta}{||q_\Delta||}|| + \gamma \cdot ||t^* - t_\Delta|| \tag{2}$$

In essence, this energy weighs the numerical error in rotation against the one in translation by means of the hyper-parameter γ and can be optimized correctly when solutions are unique (as is the case, e.g., of camera pose regression). Unfortunately, the above formulation only works for injective relations where an input image pair gets always mapped to the same transformation. In the case of one-to-many mappings, i.e. an image pair can have multiple correct solutions, the optimization does not converge since it is pulled into multiple directions and regresses the average instead. In the context of our task, visual ambiguity is common for most man-made objects because they are either symmetric or share the same appearance from multiple viewpoints. For these objects there is a large (sometimes infinite) set of refinement solutions that yield the same visual result. In order to regress q_Δ and t_Δ under ambiguity, we therefore propose an alternative formulation.

Proxy loss for visual alignment Instead of explicitly minimizing an ambiguous error in transformation, we strive to minimize an unambiguous error that measures similarity in appearance. We thus treat our search for the pose refinement parameters as a subproblem inside another proxy loss that optimizes for visual alignment. While there are multiple ways to define a similarity measure, we seek one that fulfills the following properties: 1) invariant to symmetric or indistinguishable object views, 2) robust to color deviation, illumination change and occlusion as well as 3) smooth and differentiable with respect to the pose.

To fulfill the first two properties we propose to align the object contours. Tracking the 6D pose of objects via projective contours has been presented before [13, 5, 3] but, to the best of our knowledge, has not so far been introduced in a deep learning framework. Contour tracking allows to reduce the difficult problem of 3D geometric alignment to a simpler task of 2D silhouette matching by moving through a distance transform, avoiding explicit correspondence search. Furthermore, a physical contour is not affected by deviations in coloring or lighting which makes it even more appealing for pure RGB methods. We refer to Figure 3 for a training example and the visualization of the contours we align.



(a) Synthetic scene input image $\mathcal{S}$ (b) 6D hypothesis rendering $\mathcal{H}$ (c) Pose estimate at initial training state (d) Refinement after convergence

Fig. 3: Visualization of our training procedure. In (a) and (b) we show the two image patches that constitute one training sample and the input to our network. We highlight for the reader the contours for which we seek the projective alignment from white to red. In (c) we see the initial state of training with no refinement together with the distance transform of the scene $\mathcal{D}_\mathcal{S}$ and the projection of 3D sample points $V_\mathcal{H}$ from the initial 6D hypothesis. Finally, in (d) we can see the refinement after convergence.

Fulfilling smoothness and differentiability is more difficult. An optimization step for this energy requires to render the object with the current pose hypothesis for contour extraction, estimate the similarity with the target contour and back-propagate the error gradient such that the refined hypothesis' projected contour is closer in the next iteration. Unfortunately, back-propagating through a rendering pipeline is non-trivial (due to, among others, z-buffering and rasterization). We therefore propose here a novel formulation to drive the network optimization successfully through the ambiguous 6D solution space. We employ an idea, introduced in [13], that allows us to use an approximate contour for optimization without iterative rendering. When creating a training sample, we use the depth map of the rendering to compute a 3D point cloud in camera space and sample a sparse point set on the contour, denoted as $V := \{v \in \mathbb{R}^3\}$. The idea is then to transform these contour points with the current refinement estimate (q_Δ, t_Δ), followed by a projection into the scene. This mimics a rendering plus contour extraction at no cost and allows for back-propagation.

For a given training sample with input patch pair $(\mathcal{S}, \mathcal{H})$, a distance transform of the scene contour $\mathcal{D}_\mathcal{S}$ and hypothesis contour points $V_\mathcal{H}$, we define the loss

$$\mathcal{L}(q_\Delta, t_\Delta, \mathcal{D}_\mathcal{S}, V_\mathcal{H}) := \sum_{v \in V_\mathcal{H}} \mathcal{D}_\mathcal{S}\left[\pi\left(q_\Delta \cdot v \cdot q_\Delta^{-1} + t_\Delta\right)\right] \tag{3}$$

with q_Δ^{-1} being the conjugate quaternion. With the formulation above we also free ourselves from any γ-balancing issue between quaternion and translation magnitudes as in a standard regression formulation.

Minimizing the above loss with a gradient descent step forces a step towards the 0-level set of the distance transform. We basically tune the network weights to rotate and translate the object in 6D to maximize the projected contour overlap. While this works well in practice, we have observed that for certain objects and stronger pose perturbations the optimization can get stuck in local minima. This occurs when our loss drives the contour points into a configuration where the distance transform allows them to settle in local valleys. To remedy this problem we introduce a bi-directional loss formulation that simultaneously aligns the contours of hypothesis as well as scene onto each other, coupled and constrained by the same pose update. We thus have an additional term that runs into the opposite direction:

$$\mathcal{L} := \mathcal{L}(q_\Delta, t_\Delta, \mathcal{D}_\mathcal{S}, V_\mathcal{H}) + \mathcal{L}(q_\Delta^{-1}, -t_\Delta, \mathcal{D}_\mathcal{H}, V_\mathcal{S}). \tag{4}$$

This final loss $\mathcal{L}$ does not only alleviate the locality problem but has also shown to lead to faster training overall. We therefore chose this energy for all experiments.

3.3 Network design and implementation

We give a schematic overview of our network structure in Figure 2 and provide here more details. In order to ensure fast inference, our network follows a fully-convolutional design. The network is fed with two $224 \times 224 \times 3$ input patches representing the cropped scene image $\mathcal{S}$ and cropped render image $\mathcal{H}$. Both patches run in separate paths through the first levels of an InceptionV4 [35] instance to extract low-level features. Thereafter we concatenate the two feature tensors, down-sample by employing max-pooling as well as a strided 3×3 convolution, and concatenate the results again. After two Inception-A blocks we branch off into two separate paths for the regression of rotation and translation. In each we employ two more Inception-A blocks before down-sampling by another strided 3×3 convolution. The resulting tensors are then convolved with either a $6 \times 6 \times 4$ kernel to regress a 4D quaternion or a $6 \times 6 \times 3$ kernel to predict a 3D update translation vector.

Initial experiments showed clearly that training the network from scratch made it impossible to bridge the domain gap between synthetic and real images. Similarly to [26, 36] we found that the network focused on specific appearance details of the rendered CAD models and the performance on real imagery collapsed drastically. Synthetic images usually possess very sharp edges and clear corners. Since the first layers learn low-level features they overfit quickly to this perfect rendered world during training. We therefore copied the first five convolutional blocks from a pre-trained model and froze their parameters. We show the improvements in terms of generalization to real data in the supplement.

Further, we initialize the final regression layers such that the bias equals identity quaternion and zero translation whereas the weights are given a small Gaussian noise level of $\sigma = 0.001$. This ensures that we start refinement from a neutral pose, which is crucial for the evaluation of the projective visual loss.

While our approach produces very good refinements in a single shot we decided to also implement an iterative version where we run the pose refinement multiple times until the regressed update falls under a threshold.

4 Evaluation

We implemented our method with TensorFlow 1.4 [37] and ran it on a i7-5820K@3.3GHz with an NVIDIA GTX 1080. For all experiments we ran the training with 100k iterations, a batch size of 16 and ADAM with a learning rate of $3 \cdot 10^{-4}$. Furthermore, we fixed the number of 3D contour points per view to $|V_{\mathcal{S}}| = |V_{\mathcal{H}}| = 100$. Additionally, our method is real-time capable since one iteration requires approximately 25ms during testing.

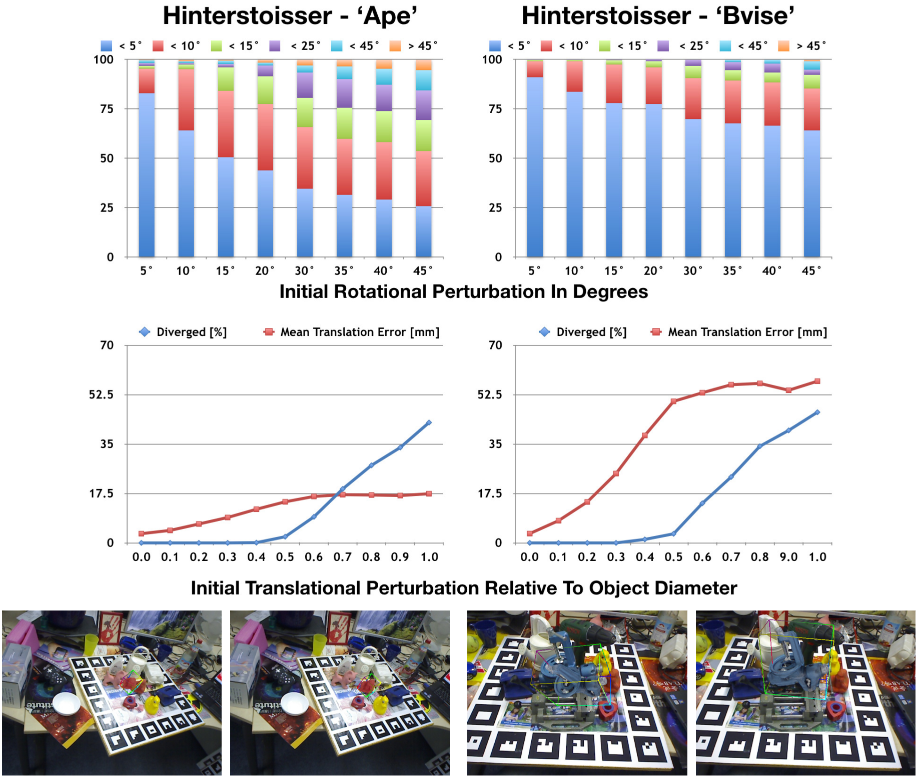
To evaluate our method, we carried out experiments on three, both synthetic and real, datasets and will convey that our method can come close to RGB-D based approaches. In particular, the first dataset, referred to as 'Hinterstoisser', was introduced in [38] and consists of 15 sequences each possessing approximately 1000 images with clutter and mild occlusion. Only 13 of these provide watertight CAD models and we therefore, like others before us, skip the other two sequences. The second one, which we refer to as 'Tejani', was proposed in [39] and consists of six mostly semi-symmetric, textured objects each undergoing different levels of occlusion. In contrast to the first two real datasets, the latter one, referred to as 'Choi' [40], consists of four synthetic tracking sequences.

In essence, we will first conduct some self-evaluation in which we illustrate our convergence properties with respect to different degrees of pose perturbation on real data. Then we show our method when applied to object tracking on 'Choi'. As a second application, we compare our approach to a variety of other state-of-the-art RGB and RGB-D methods by conducting experiments in pose refinement on 'Hinterstoisser', the 'Occlusion' dataset and 'Tejani'. Finally, we depict some failure cases and conclude with a qualitative category-level experiment.

4.1 Pose perturbation

We study the convergence behavior of our method by taking correct poses, applying a perturbation by a certain amount and measure how well we can refine back to the original pose. To this end, we use the 'Hinterstoisser' dataset since it provides a lot of variety in terms of both colors and shapes. For each frame of a particular sequence we perturb the ground truth pose either by an angle or by a translation vector. In Figure 4 we illustrate our results for the 'ape' and the 'bvise' objects and kindly refer the reader to the supplement for all graphs. In particular, we report our results for increasing degrees of angular perturbations from 5°to 45°and for increasing translation perturbations from 0 to 1 relative to the object's diameter. We define divergence if the refined rotation is above 45°in error or the refined translation larger than half of the object's diameter and we employ 10 iterative steps to maximize our possible precision.

In general, our method can recover poses very robustly even under strong perturbations. Even for the extreme case of rotating the 'bvise' with 45°we can refine back to an error less than 5°in more than 60% of all trials, and to an error less than 10°in more than 80% of all runs. Additionally, our approach only diverged for less than 1%. However, for the more difficult 'ape' object our numbers worsen. In particular, in almost 50% of the cases we were not able



Perturbation Refinement Perturbation Refinement

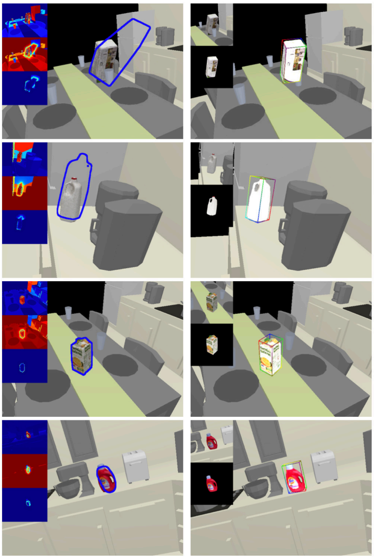
Fig. 4: Top: Perturbation results for two objects from [38] for increasing rotation and translation levels. Bottom: Qualitative results from the same experiment.

to rotate back the object to an error of less than 10%. Yet, this can be easily explained by the object's appearance. The 'ape' is a rather small object with poor texture and non-distinctive shape, which does not provide enough information to hook onto whereas the 'bvise' is large and rich in appearance. It is noteworthy that the actual divergence behavior in rotation is similar for both and that the visual alignment for the 'ape' is often very good despite the error in pose.

The translation error correlates almost linearly between initial and final pose. We also observe an interesting tendency starting from perturbation levels at around 0.6 after which the results can be divided up into two distinct sets: either the pose diverges or the error settles on a certain level. This implies that certain viewpoints are easy to align as long as they have a certain visual overlap to begin with, rather independent of how strong we perturb. Other views instead are more difficult with higher perturbations and diverge from some point on.

		PCL	C&C	Krull	Tan	Kehl	Tjaden	Ours
(a) *Kinect Box*	t_x	43.99	1.84	0.8	1.54	**0.76**	55.75	1.46
	t_y	42.51	2.23	1.67	1.90	**1.09**	70.57	2.28
	t_z	55.89	1.36	0.79	**0.34**	0.38	402.14	10.61
	α	7.62	6.41	1.11	0.42	**0.17**	42.61	1.84
	β	1.87	0.76	0.55	0.22	**0.18**	27.74	2.09
	γ	8.31	6.32	1.04	0.68	**0.20**	38.979	1.23
(b) *Milk*	t_x	13.38	0.93	**0.51**	1.23	0.64	39.21	3.89
	t_y	31.45	1.94	1.27	0.74	**0.59**	48.13	4.25
	t_z	26.09	1.09	0.62	**0.24**	**0.24**	332.11	57.68
	α	59.37	3.83	2.19	0.50	**0.41**	45.54	38.74
	β	19.58	1.41	1.44	**0.28**	0.29	26.37	27.62
	γ	75.03	3.26	1.90	0.46	**0.42**	21.72	42.68
(c) *Orange Juice*	t_x	2.53	0.96	0.52	1.10	**0.50**	2.29	0.65
	t_y	2.20	1.44	0.74	0.94	**0.69**	2.85	**0.69**
	t_z	1.91	1.17	0.63	0.18	**0.17**	48.61	6.49
	α	85.81	1.32	1.28	0.35	**0.12**	8.46	1.5
	β	42.12	0.75	1.08	0.24	**0.20**	5.95	0.68
	γ	46.37	1.39	1.20	0.37	**0.19**	2.24	0.39
(d) *Tide*	t_x	1.46	0.853	0.69	0.73	**0.34**	1.31	1.74
	t_y	2.25	1.37	0.81	0.56	**0.49**	0.83	0.74
	t_z	0.92	1.20	0.81	0.24	**0.18**	12.49	10.71
	α	5.15	1.78	2.10	0.31	**0.15**	2.03	1.78
	β	2.13	1.09	1.38	**0.25**	0.39	1.56	1.64
	γ	2.98	1.13	1.27	**0.34**	0.37	1.39	0.80

(a) Errors on 'Choi' in respect to others.



(b) Tracking quality compared to [5].

Fig. 5: Left: Translation (mm) and rotation (degrees) errors on Choi for PCL's ICP, Choi and Christensen (C&C)[40], Krull[12], Tan[11], Kehl[13], Tjaden[5] and our method. Right: Comparing [5] (left) to us (right) using only RGB.

4.2 Tracking

As a first use case we evaluated our method as a tracker on the 'Choi' benchmark [40]. This RGB-D dataset consists of four synthetic sequences and we present detailed numbers in Figure 5. Note that all other methods utilize depth information. We decided for this dataset because it is very hard for RGB-only methods: it is poor in terms of color and the objects are of (semi-)symmetric nature. To provide an interesting comparison we also qualitatively evaluated against our tracker implementation of [5]. While their method is usually robust for textureless objects it diverges on 3 sequences which we show and for which we provide reasoning[2] in Figure 5 and in the supplementary material. In essence, except for the 'Milk' sequence we can report very good results. The reason why we performed comparably bad on the 'Milk' resides in the fact that our method already treats it as a rather symmetric object. Thus, sometimes it rotates the object along its Y-axis, which has a negative impact on the overall numbers. In particular, while already being misaligned, the method still tries to completely fill the object into the scene, thus, it slightly further rotates and translates the object. Referring to the remaining objects, we can easily outperform PCL's ICP for all objects and also Choi and Christensen [40] for most of the cases. Com-

[2] The authors acknowledged our conclusions in correspondence.

	ape	bvise	cam	can	cat	driller	duck	box	glue	holep	iron	lamp	phone	total
No Refinement	0.64	0.65	0.71	0.72	0.63	0.62	0.65	0.64	0.64	0.69	0.71	0.63	0.69	0.66
2D Edge-based ICP	0.73	0.67	0.73	0.76	0.68	0.67	0.72	0.73	0.72	0.71	0.74	0.67	0.70	0.71
3D Cloud-based ICP	**0.86**	**0.88**	**0.91**	**0.87**	**0.87**	**0.85**	0.83	0.84	0.75	0.77	**0.85**	**0.84**	**0.81**	**0.84**
Ours	0.83	0.83	0.75	**0.87**	0.79	**0.85**	**0.87**	**0.88**	**0.85**	**0.82**	**0.85**	0.80	0.83	0.83

Table 1: VSS scores for each sequence of [38] with poses initialized from SSD-6D [26]. The first three rows are provided by [26]. We evidently outperform 2D-based ICP by a large margin and are on par with 3D-based ICP.

	Rot. Error [°]	Transl. Error [mm]	ADD [%]
No Ref.	27.96	9.75, 9.33, 71.09	7.4
3D ICP	17.62	10.42, 10.56, **27.31**	**90.9**
Ours	**16.17**	**4.9**, **5.87**, 42.69	34.1
[29]	–	–	43.6
[28]	–	–	50.2

	Rot. Error [°]	Transl. Error [mm]	ADD [%]
No Ref.	34.42	13.7, 13.4, 77.5	6.2
Ours	24.36	8.5, 9.0, 49.1	27.5

(a) Absolute pose errors on [38] and [42].

Sequence	Ours	MSE Loss	Kehl [13]	Tjaden [5]
Camera	**0.803**	0.562	0.493	0.385
Coffee	**0.848**	0.717	0.747	0.170
Joystick	**0.850**	0.746	0.773	0.298
Juice	**0.828**	0.613	0.523	0.205
Milk	**0.766**	0.721	0.580	0.514
Shampoo	**0.804**	0.700	0.648	0.250
Total	**0.817**	0.676	0.627	0.304

(b) VSS scores for each sequence of [39].

Table 2: Refinement scores with poses initialized from SSD-6D [26]. Left: Average ADD scores on 'Hinterstoisser' [38] (top) and 'Occlusion' [42] (bottom). Right: VSS scores on 'Tejani'. We compare our visual loss to naive pose regression as well as two state-of-the-art trackers for the case of RGB [5] and RGB-D [13].

pared to Krull [12], which is a learned RGB-D approach, we perform better for some values and worse for others. Note that our translation error along the Z-axis is quite high. Since the difference in pixels is almost nonexistent when the object is moved only a few millimeters, it is almost impossible to estimate the exact distance of the object without leveraging depth information. This has also been discussed in [41] and is especially true for CNNs due to pooling operations.

4.3 Detection refinement

This set of experiments analyzes our performance in a detection scenario where an object detector will provide rough 6D poses and the goal is to refine them. We decided to use the results from SSD-6D [26], an RGB-based detection method, that outputs 2D detections with a pool of 6D pose estimates each. The authors publicly provide their trained networks and we use them to detect and create 6D pose estimates which we feed into our system. Tables 1, 2b and 2a depict our results for the 'Hinterstoisser', 'Occlusion' and the 'Tejani' dataset using different metrics. We maximally ran 5 iterations of our method, yet, we also stopped if the last update was less than 1.5°and 7.5mm. Since our method is particularly strong at recovering from bad initializations, we employ the same RGB-verification strategy as SSD-6D. However, we apply it before conducting the refinement, since in contrast to them, we can also deal with imperfect initializations, as long

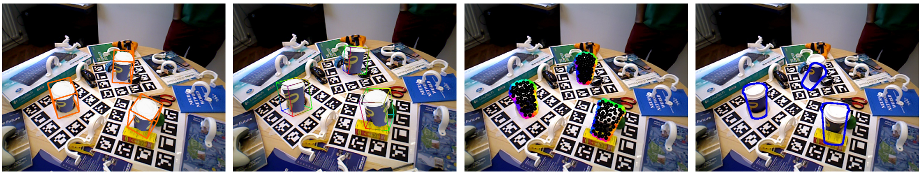


Fig. 6: Comparison on Tejani between (from left to right) our visual loss, mean squared error loss, the RGB-D tracker from [13] and the RGB tracker from [5].

as they are not completely misaligned. We report our errors with the VSS metric (which is VSD from [43] with $\tau = \infty$) that calculates a visual 2D error as the pixel-wise overlap between the renderings of ground truth pose and estimated pose. Furthermore, to compare better to related work, we also use the ADD score [38] to measure a 3D metrical error as the average point cloud deviation between real pose and inferred pose when transformed into the scene. A pose is counted as correct if the deviation is less than a $\frac{1}{10}$th of the object diameter.

Referring to 'Hinterstoisser' with the VSS metric, we can strongly improve the state-of-the-art for most objects. In particular, for the case of RGB only, we can report an average VSS score of 83%, which is an improvement of impressive and can thus successfully bridge the gap between RGB and RGB-D in terms of pose accuracy.

Except for the 'cam' and the 'cat' object our results are on par with or even better than SSD-6D + 3D refinement. ICP relies on good correspondences and robust outlier removal which in turn requires very careful parameter tuning. Furthermore, ICP is often unstable for rougher initializations. In contrast, our method learns refinement end-to-end and is more robust since it adapts to the specific properties of the object during training. However, due to this, our method requires meshes of good quality. Hence, similar to SSD-6D we have especially problems for the 'cam' object since the model appearance strongly differs from the real images which exacerbates training. Also note that their 3D refinement strategy uses ICP for each pose in the pool, followed by a verification over depth normals to decide for the best pose. Our method instead uses a simple check over image gradients to pick the best.

With respect to the ADD metric we fall slightly behind the other state-of-the-art RGB methods [28, 29]. We got the 3D-ICP refined poses from the SSD-6D authors and analyzed the errors in more detail in Table 2a. We see again that we have bigger errors along the Z-axis, but less errors along X and Y. Unfortunately, the ADD metric penalizes this deviation overly strong. Interestingly, [28, 29] have better scores and we reason this to come from two facts. The datasets are annotated via ICP with 3D models against depth data. Unfortunately, inaccurate intrinsics and the sensor registration error between RGB and D leads to an inherent mismatch where the ICP 6D pose does not always align perfectly in RGB. Purely synthetic RGB methods like ours or [26] suffer from (1) a domain gap in terms of texture/shape and (2) the dilemma that better RGB performance

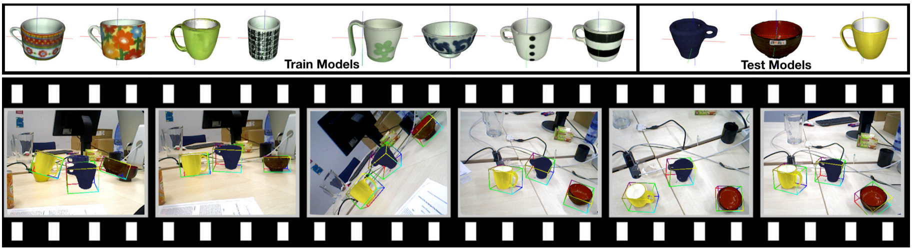


Fig. 7: Qualitative category-level experiment where we train our network on a specific set of mugs and bowls and track hitherto unseen models. The first frame depicts very rough initialization while the next frames show some intermediate refined poses throughout the sequence. The supplement shows the full video.

can worsen results when comparing to that 'true' ICP pose. We suspect that [28, 29] can learn this registration error implicitly since they train on real RGB cutouts with associated ICP pose information and thus avoid both problems. We often observe that our visually-perfect alignments in RGB fail the ADD criterion and we show examples in the supplement. Since our loss actually optimizes a form of VSS to maximize contour overlap, we can expect the ADD scores to go up only when perfect alignment in color equates perfect alignment in depth.

Eventually, referring to the 'Occlusion' dataset, we can report a strong improvement compared to the original numbers from SSD-6D, despite the presence of strong occlusion. In particular, while the rotational error decreased by approximately 8 degrees, the translational error dropped by 4mm along 'X' and 'Y' axes and by 28mm along 'Z'. Thus, we can increase ADD from 6.2% up to 28.5%, which demonstrates that we can deal with strong occlusion in the scene.

For 'Tejani' we decided to show the improvement over networks trained with a standard regression loss (MSE). Additionally, we re-implemented the RGB tracker from [5] and were kindly provided with numbers from the authors of the RGB-D tracker from [13] (see Figure 6). Since the dataset mostly consists of objects with geometric symmetry, we do not measure absolute pose errors here but instead report our numbers with the VSS metric. The MSE-trained networks constantly underperform since the dataset models are of symmetric nature which in turn leads to a large difference of 14% in comparison to our visual loss. This result stresses the importance of correct symmetry entangling during training. The RGB tracker was not able to refine well due to the fact that the color segmentation was corrupted by either occlusions or imperfect initialization. The RGB-D tracker, which builds on the same idea, performed better because it uses the additional depth channel for segmentation and optimization.

4.4 Category-level tracking

We were curious to find out whether our approach can generalize beyond a specific CAD model, given that many objects from the same category share similar

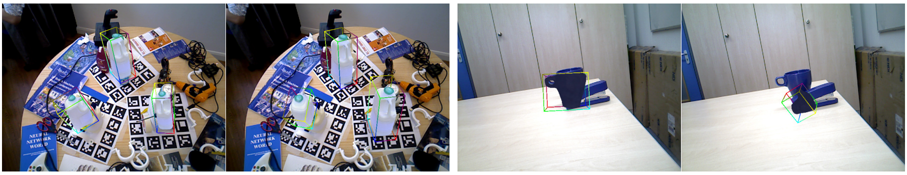


Fig. 8: Two prominent failure cases: Occlusion (left pair) and objects of very similar colors and shapes (right pair) can negatively influence the regression.

appearance and shape properties. To this end, we conducted a final qualitative experiment (see Figure 7) where we collected a total of eight CAD models of cups, mugs and a bowl and trained simultaneously on all. During testing we then used this network to track new, unseen models from the same category. We were surprised to see that the approach has indeed learned to metrically track previously unseen but nonetheless similar structures. While the poses are not as accurate as for the single-instance case, it seems that one can indeed learn the projective relation of structure and how it changes under 6D motion, provided that at least the projection functions (i.e. camera intrinsics) are constant. We show the full sequence in the supplementary material.

4.5 Failure cases

Figure 8 illustrates two known failure cases where the left image of each pair represents initialization and the right image the refined result. Although we train with occlusion certain occurrences can worsen our refinement nonetheless. While two 'milk' instances were refined well despite occlusion, the left 'milk' instance could not be recovered correctly. The network assumes the object to end at the yellow pen and only maximizes the remaining pixel-wise overlap. Besides occlusion, objects of similar color and shape can in rare cases lead to confusion. As shown in the right pair, the network mistakenly assumed the stapler, instead of the cup, to be the real object of interest.

5 Conclusion

We believe to have presented a new approach towards 6D model tracking in RGB with the help of deep learning and we demonstrated the power of our approach on multiple datasets and for the scenarios of pose refinement and for instance/category tracking. Future work will include investigation towards generalization to other domains, e.g. the suitability towards visual odometry.

Acknowledgments We would like to thank Toyota Motor Corporation for funding and supporting this work.

References

1. Vacchetti, L., Lepetit, V., Fua, P.: Stable Real-Time 3D Tracking Using Online and Offline Information. TPAMI (2004)
2. Park, Y., Lepetit, V.: Multiple 3d object tracking for augmented reality. In: ISMAR. (2008)
3. Prisacariu, V.A., Reid, I.D.: PWP3D: Real-Time Segmentation and Tracking of 3D Objects. IJCV (2012)
4. Dambreville, S., Sandhu, R., Yezzi, A., Tannenbaum, A.: A Geometric Approach to Joint 2D Region-Based Segmentation and 3D Pose Estimation Using a 3D Shape Prior. SIAM Journal on Imaging Sciences (2010)
5. Tjaden, H., Schwanecke, U., Schoemer, E.: Real-Time Monocular Segmentation and Pose Tracking of Multiple Objects. In: ECCV. (2016)
6. Tjaden, H., Schwanecke, U., Schömer, E.: Real-Time Monocular Pose Estimation of 3D Objects using Temporally Consistent Local Color Histograms. In: ICCV. (2017)
7. Schmaltz, C., Rosenhahn, B., Brox, T., Cremers, D., Weickert, J., Wietzke, L., Sommer, G.: Region-Based Pose Tracking. In: IbPRIA. (2007)
8. Brox, T., Rosenhahn, B., Gall, J., Cremers, D.: Combined Region and Motion-Based 3D Tracking of Rigid and Articulated Objects. TPAMI (2010)
9. Schmaltz, C., Rosenhahn, B., Brox, T., Weickert, J.: Region-based pose tracking with occlusions using 3D models. MVA (2012)
10. Pauwels, K., Rubio, L., Diaz, J., Ros, E.: Real-time model-based rigid object pose estimation and tracking combining dense and sparse visual cues. In: CVPR. (2013)
11. Tan, D.J., Tombari, F., Ilic, S., Navab, N.: A Versatile Learning-based 3D Temporal Tracker : Scalable , Robust , Online. In: ICCV. (2015)
12. Krull, A., Michel, F., Brachmann, E., Gumhold, S., Ihrke, S., Rother, C.: 6-DOF Model Based Tracking via Object Coordinate Regression. In: ACCV. (2014)
13. Kehl, W., Tombari, F., Ilic, S., Navab, N.: Real-Time 3D Model Tracking in Color and Depth on a Single CPU Core. In: CVPR. (2017)
14. Garon, M., Lalonde, J.F.: Deep 6-DOF Tracking. In: ISMAR. (2017)
15. Kendall, A., Cipolla, R.: Geometric loss functions for camera pose regression with deep learning. In: CVPR. (2017)
16. Zhou, T., Brown, M., Snavely, N., Lowe, D.G.: Unsupervised Learning of Depth and Ego-Motion from Video. In: CVPR. (2017)
17. Wang, S., Clark, R., Wen, H., Trigoni, N.: DeepVO: Towards End to End Visual Odometry with Deep Recurrent Convolutional Neural Networks. In: ICRA. (2017)
18. Ummenhofer, B., Zhou, H., Uhrig, J., Mayer, N., Ilg, E., Dosovitskiy, A., Brox, T.: DeMoN: Depth and Motion Network for Learning Monocular Stereo. In: CVPR. (2017)
19. Hexner, J., Hagege, R.R.: 2D-3D Pose Estimation of Heterogeneous Objects Using a Region Based Approach. IJCV (2016)
20. Rosenhahn, B., Brox, T., Cremers, D., Seidel, H.P.: A comparison of shape matching methods for contour based pose estimation. LNCS (2006)
21. Drummond, T., Cipolla, R.: Real-time visual tracking of complex structures. TPAMI (2002)
22. Tateno, K., Kotake, D., Uchiyama, S.: Model-based 3D Object Tracking with Online Texture Update. In: MVA. (2009)
23. Seo, B.K., Park, H., Park, J.I., Hinterstoisser, S., Ilic, S.: Optimal local searching for fast and robust textureless 3D object tracking in highly cluttered backgrounds. In: TVCG. (2014)

24. Bibby, C., Reid, I.: Robust Real-Time Visual Tracking using Pixel-Wise Posteriors. In: ECCV. (2008)
25. Prisacariu, V.A., Murray, D.W., Reid, I.D.: Real-Time 3D Tracking and Reconstruction on Mobile Phones. TVCG (2015)
26. Kehl, W., Manhardt, F., Ilic, S., Tombari, F., Navab, N.: SSD-6D: Making RGB-Based 3D Detection and 6D Pose Estimation Great Again. In: ICCV. (2017)
27. Liu, W., Anguelov, D., Erhan, D., Szegedy, C., Reed, S., Fu, C.y., Berg, A.C.: SSD : Single Shot MultiBox Detector. In: ECCV. (2016)
28. Brachmann, E., Michel, F., Krull, A., Yang, M.Y., Gumhold, S., Rother, C.: Uncertainty-Driven 6D Pose Estimation of Objects and Scenes from a Single RGB Image. In: CVPR. (2016)
29. Rad, M., Lepetit, V.: BB8: A scalable, accurate, robust to partial occlusion method for predicting the 3d poses of challenging objects without using depth. In: ICCV. (2017) 3848–3856
30. Pavlakos, G., Zhou, X., Chan, A., Derpanis, K.G., Daniilidis, K.: 6-DoF Object Pose from Semantic Keypoints. In: ICRA. (2017)
31. Wu, J., Xue, T., Lim, J.J., Tian, Y., Tenenbaum, J.B., Torralba, A., Freeman, W.T.: Single Image 3D Interpreter Network. In: ECCV. (2016)
32. Jaderberg, M., Simonyan, K., Zisserman, A., Kavukcuoglu, K.: Spatial Transformer Networks. In: NIPS. (2015)
33. Bhagavatula, C., Zhu, C., Luu, K., Savvides, M.: Faster Than Real-time Facial Alignment: A 3D Spatial Transformer Network Approach in Unconstrained Poses. In: ICCV. (2017)
34. Kendall, A., Grimes, M., Cipolla, R.: PoseNet: A Convolutional Network for Real-Time 6-DOF Camera Relocalization. In: ICCV. (2015)
35. Szegedy, C., Ioffe, S., Vanhoucke, V., Alemi, A.A.: Inception-v4, inception-resnet and the impact of residual connections on learning. In: ICLR Workshop. (2016)
36. Hinterstoisser, S., Lepetit, V., Wohlhart, P., Konolige, K.: On pre-trained image features and synthetic images for deep learning. CoRR **abs/1710.10710** (2017)
37. Abadi, M., Barham, P., Chen, J., Chen, Z., Davis, A., Dean, J., Devin, M., Ghemawat, S., Irving, G., Isard, M., Kudlur, M., Levenberg, J., Monga, R., Moore, S., Murray, D., Steiner, B., Tucker, P., Vasudevan, V., Warden, P., Wicke, M., Yu, Y., Zheng, X.: TensorFlow: Large-scale machine learning on heterogeneous systems. In: OSDI. (2016)
38. Hinterstoisser, S., Lepetit, V., Ilic, S., Holzer, S., Bradski, G., Konolige, K., Navab, N.: Model based training, detection and pose estimation of texture-less 3D objects in heavily cluttered scenes. In: ACCV. (2012)
39. Tejani, A., Tang, D., Kouskouridas, R., Kim, T.k.: Latent-class hough forests for 3D object detection and pose estimation. In: ECCV. (2014)
40. Choi, C., Christensen, H.: RGB-D Object Tracking: A Particle Filter Approach on GPU. In: IROS. (2013)
41. Holloway, R.L.: Registration error analysis for augmented reality. Presence: Teleoper. Virtual Environ. **6**(4) (August 1997) 413–432
42. Brachmann, E., Krull, A., Michel, F., Gumhold, S., Shotton, J., Rother, C.: Learning 6D Object Pose Estimation using 3D Object Coordinates. In: ECCV. (2014)
43. Hodan, T., Matas, J., Obdrzalek, S.: On Evaluation of 6D Object Pose Estimation. In: ECCV Workshop. (2016)

5.2 Challenges in 6D Pose Estimation

While the accuracy of 6D pose estimation from RGB data keeps increasing [8], there are still many limitations which are not well addressed in literature. Exemplary, occlusion usually significantly deteriorates performance [123]. Similarly, also ambiguities in pose [163] and other external factors such as illumination [2] can have a significant impact on the detection results. This is particularly challenging, since in contrast to other tasks such as image classification, there are no large-scale datasets explaining all the variation within these parameters.

Therefore, this section is devoted to shed light on these challenges. Thereby, Section 5.2.1 focuses on how to tackle different kinds of ambiguities in pose and Section 5.2.2 approaches the problem of lighting variations.

5.2.1 Explaining The Ambiguity of Object Detection And 6D Pose From Visual Data (ICCV 2019)

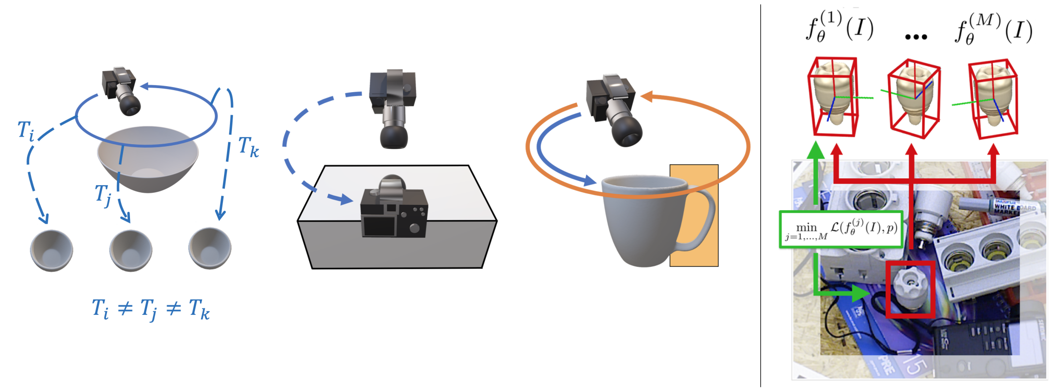


Figure 5.3. **Ambiguities in Pose.** Left: Different poses can have the same visual appearance due to symmetries and occlusion. Exemplary, the bowl is visually identical when observed from any of the three poses T_i, T_j and T_k, despite $T_i \neq T_j \neq T_k$. Learning to estimate the pose causes the network to predict the canonical mean of all poses that are identical under ambiguities. However, the canonical mean pose does not need to represent a visually correct solution. Right: Therefore, we tackle the problem by means of casting multiple pose hypotheses for a detection. Optimizing only for the *best* hypothesis enforces a Voronoi tessellation of the output space, which can be leveraged to recover a visually plausible pose despite ambiguities.

Inferring the 6D pose from monocular data can be a highly ambiguous task. As illustrated in Figure 5.3 [left], symmetries and occlusion can lead to multiple plausible poses under perspective projection. Rupprecht *et al.* [192] have recently shown that ambiguities can have a strong negative impact on the performance of a network, as the model is prone to predict the canonical mean over the underlying multi-model distribution. Similarly, naïve optimization for pose with

$$\theta^* = \operatorname*{argmin}_{\theta} \frac{1}{N} \sum_{i=0}^{N} \hat{\mathcal{L}}(\hat{T}, \bar{T}), \tag{5.3}$$

encourages to infer poses that explain all plausible poses equally well, which does not work well under ambiguities.

In this work we model the 6D pose by means of multiple hypotheses as a way to approximate the underlying distribution (*c.f.* Figure 5.3). To this end, for a given input image I, we predict N hypotheses for pose. During learning we then only backpropagate the *best* pose within all hypotheses, rewriting $\hat{\mathcal{L}}$ as

$$\mathcal{L} = \min_{i,\dots,N} \hat{\mathcal{L}}(\hat{T}_i, \bar{T}). \tag{5.4}$$

As only the best pose with respect to the ground truth is optimized, the different hypotheses spread out, fragmenting the output space similar to a Voronoi tessellation. Notice that all hypotheses collapse to a single pose in case no ambiguity is present. The new formulation enables robust pose estimation under ambiguities, as well as further reasoning about the ambiguity itself. In particular, we are capable of understanding the type of ambiguity and can exploit this information to further enhance pose quality. To this end, we apply Principle Component Analysis (PCA) to the hypotheses to infer if there is an ambiguity present and if the hypotheses span a continuous pose space. Subsequently, if an ambiguity is found, we leverage clustering together with mean shift as a robust estimator to obtain the final output pose. Otherwise, if no ambiguity could be detected, we merge all hypotheses to increase the pose robustness and additionally harness the hypotheses as a way to characterize the uncertainty in pose. In particular, we compute the standard deviation σ over the set of hypotheses $\{\hat{T}_1, \dots, \hat{T}_N\}$ and utilize it as a measurement for reliability.

Experiments on LM [131] and T-LESS [133] demonstrate that for symmetric objects our ambiguity-aware pose formulation significantly improves over the single-hypothesis baseline. Nevertheless, also when no ambiguity is present, our robust averaging can easily surpass the single-hypothesis approach. Further, while lowering the threshold for σ the pose accuracy keeps increasing, proofing the potential of using multiple hypotheses to measure uncertainty. Noteworthy, the new formulation for pose allows to robustly infer pose under the presence of ambiguity, without requiring any additional labels.

Contributions. I proposed the idea to utilize multiple hypotheses as a mean to deal with pose ambiguities and Diego Martin Arroyo implemented the first version of the method and ran some initial experiments. I extended the initial version then with our final Inception-V4 backbone and the respective heads for multiple rotation and depth regression. I also conducted all the evaluations. Christian Rupprecht contributed to the implementation of the multiple hypotheses loss. Benjamin Busam helped analyzing the ambiguities and Tolga Birdal assisted the Bingham distribution visualizations.

Explaining The Ambiguity of Object Detection and 6D Pose From Visual Data

Fabian Manhardt[1,*], Diego Martin Arroyo[1,*], Christian Rupprecht[2], Benjamin Busam[1,3], Tolga Birdal[4], Nassir Navab[1], Federico Tombari[1,5]

[1] Technical University of Munich
[2] University of Oxford
[3] Huawei
[4] Stanford University
[5] Google
* Equal Contribution

It is the accepted but not the published version of the paper due to copyright restrictions.

Published version: https://doi.org/10.1109/ICCV.2019.00694

Explaining the Ambiguity of Object Detection and 6D Pose From Visual Data

Fabian Manhardt[1,*] fabian.manhardt@tum.de
Diego Martin Arroyo[1,*] martin.arroyo@tum.de
Christian Rupprecht[2] chrisr@robots.ox.ac.uk
Benjamin Busam[1,3] benjamin.busam@huawei.com
Tolga Birdal[4] tbirdal@stanford.edu
Nassir Navab[1] nassir.navab@tum.de
Federico Tombari[1,5] tombari@in.tum.de

[1]Technical University of Munich [2]University of Oxford [3]Huawei [4]Stanford University [5]Google

Abstract

3D object detection and pose estimation from a single image are two inherently ambiguous problems. Oftentimes, objects appear similar from different viewpoints due to shape symmetries, occlusion and repetitive textures. This ambiguity in both detection and pose estimation means that an object instance can be perfectly described by several different poses and even classes. In this work we propose to explicitly deal with these ambiguities. For each object instance we predict multiple 6D pose outcomes to estimate the specific pose distribution generated by symmetries and repetitive textures. The distribution collapses to a single outcome when the visual appearance uniquely identifies just one valid pose. We show the benefits of our approach which provides not only a better explanation for pose ambiguity, but also a higher accuracy in terms of pose estimation.

1. Introduction

Driven by deep learning, image-based object detection has recently made a tremendous leap forward in both accuracy as well as efficiency [39, 16, 31, 38]. An emerging research direction in this field is the estimation of the object's pose in 3D space over the existing 6-Degrees-of-Freedom (DoF) rather than on the 2D image plane [24, 37, 46, 51, 34, 29, 49, 33]. This is motivated by a strong interest in achieving robust and accurate monocular 6D pose estimation for applications in the field of robotic grasping, scene understanding and augmented/mixed reality, where the use of a 3D sensor is not feasible [36, 26, 50, 45].

Nevertheless, 6D pose estimation from RGB is a challenging problem due to the intrinsic ambiguity caused by visual appearance of objects under different viewpoints and occlusion. Indeed, most common objects exhibit shape ambiguities and repetitive patterns that cause their appearance to be very similar under different viewpoints, thus rendering pose estimation a problem with multiple correct solutions. Furthermore, also occlusion (from the same object or from others) can cause pose ambiguity.

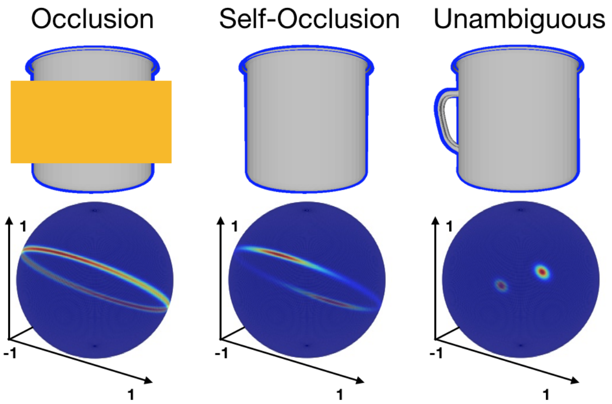


Figure 1: **Pose ambiguities**. External or self-occlusion can cause the 6DoF pose of an object to become ambiguous. Our method is able to detect and predict these ambiguities automatically without additional supervision. The antipodally symmetric Bingham distributions show that the model has understood the full range of valid poses.

For example, as illustrated in Figure 1, the cup is identical from every viewpoint in which the handle is not visible. Thus, from a single image, it is impossible to univocally estimate the current object pose. Moreover, object symmetry can also induce visual ambiguities leading to multiple poses with the same visual appearance. However, most datasets do not reflect this ambiguity, as the ground truth pose annotations are mostly uniquely defined at each frame. This is problematic for a proper optimization of the rotation, since a visually correct pose still results in a high loss. Thus, many recent 3D detectors avoid regressing the rotation directly and, instead, explicitly model the solution space in an unambiguous fashion [37, 24].

Essentially, in [24], the authors train their convolutional

[*] The first two authors contributed equally to this work.

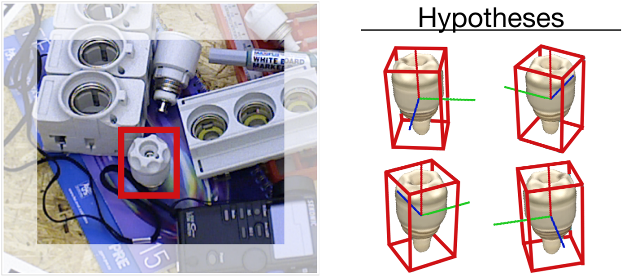


Figure 2: **Overview.** We predict M hypotheses for the pose to approximate the distribution in the solution space. Each hypothesis is visually identical from the current viewpoint.

neural network (CNN) by mapping all possible pose solutions for a certain viewpoint onto an unambiguous arc on the view sphere. Rad *et al.* [37] employ a separate CNN solely trained to classify the symmetry in order to resolve these ambiguities. However, this simplification exhibits several downsides, such as the explicit inclusion of information about certain symmetries in each trained object. Moreover, this is not always easy to model, as *e.g.* in the case of partial view ambiguity. Further, all these approaches rely on prior knowledge and annotation of the object symmetries and aim to solve the ambiguity by providing a single outcome in terms of estimated pose and object. Added to this, these methods are also unable to deal with ambiguities generated by other common factors such as occlusion.

On the contrary, Sundermeyer *et al.* [42] and Corona *et al.* [7] recently proposed novel methods to conduct pose estimation in an ambiguity-free manner. In the core, both learn a feature embedding solely based on visual appearance. Nonetheless, although [42] is able to deal with ambiguities implicitly, it does not model their detection and description explicitly. In contrast, [7] also learns to classify the order of rotational symmetry, in particular the number of equivalent views around an axis of rotation. However, they require explicit hand-annotated labels and, in addition, cannot deal with ambiguities aside from these symmetry classes such as (self-) occlusion.

In this paper we propose to model the ambiguity of the object detection and pose estimation tasks directly by allowing our learned model to predict multiple solutions, or *hypotheses*, for a given object's visual appearance (Fig 2). Inspired by Rupprecht *et al.* [40] we propose a novel architecture and loss function for monocular 6D pose estimation by means of multiple predictions. Essentially, each predicted hypothesis itself corresponds to a 3D translation and rotation. When the visual appearance is ambiguous, the model predicts a point estimate of the distribution in 3D pose space. Conversely, when the object's appearance is unique, the hypotheses will collapse into the same solution. Importantly, our model is capable of learning the distribution of these 6D hypotheses from one single ground truth pose per sample, without further supervision.

Besides providing more insight and a better explanation for the task at hand, the additional knowledge gained from rotation distributions can be exploited to improve the accuracy of the pose estimates. In essence, analyzing the distribution of the hypotheses enables us to classify if the current perceived viewpoint is ambiguous and to compute the axis of ambiguity for that specific object and viewpoint. Subsequently, when ambiguity is detected, we can employ mean shift [6] clustering over the hypotheses in quaternion space to find the main modes for the current pose. A robust averaging in 3D rotation space for each mode then yields a highly accurate pose estimate. When the view is ambiguity-free, we can improve our pose estimates by robustly averaging over all 6D hypotheses, and by taking advantage of the predicted pose distribution as a confidence measure.

Our contributions are threefold:

- We propose a novel method for 6DoF pose estimation, which can deal with the inherent ambiguities in pose by means of multiple hypotheses.
- Explicit detection of rotational ambiguities and characterization of the uncertainty in the problem without further annotation or supervision.
- A mechanism to measure the reliability and to increase the robustness of the unambiguous 6D pose prediction.

2. Related Work

We first review recent work in object detection and pose estimation from 2D and 3D data. Afterwards, we discuss common grounds and main differences with approaches aimed at symmetry detection for 3D shapes.

Object Detection and Pose Estimation. Almost all current research focus on deep learning-based methods.

[48, 25, 7] employ CNNs to learn an embedding space for the pose and class from RGB-D data, which can subsequently be utilized for retrieval. Notably, the majority of most recent deep learning based methods focus on RGB as input [24, 37, 8, 46, 51, 42]. Since utilizing pre-trained networks often accelerates convergence and leads to better local minima, these methods are usually grounded on state-of-the-art backbones for 2D object detection, such as Inception [44] or ResNet [16]. In particular, Kehl *et al.* [24] employ SSD [31] with an InceptionV4 [43] backbone and extend it to also classify viewpoint and in-plane rotation. Similarly, Sundermeyer *et al.* [42] also use SSD for localization, but employ an augmented auto-encoder for the unambiguous retrieval of the associated 6D pose. Rad *et al.* [37] utilize VGG [41] and augment it to provide the 2D projections of the 3D bounding box corners. A similar approach is chosen by [46], based on YOLO [38]. Afterwards, both apply PnP to fit the associated 3D bounding box into the regressed 2D projections, in order to estimate the 3D pose of the detection. In [51], Xiang *et al.* compute a shared feature embedding for subsequent object instance segmentation paired with pose estimation. Finally, Do *et al.* [8] extend Mask-RCNN [15] with a third branch, which provides the 3D rotation and the distance to the camera for each prediction.

Object Symmetry Detection Oftentimes, object pose ambiguity arises from symmetric shapes. We review relevant methods that extract symmetry from 3D models to outline commonalities and differences with our approach.

To our knowledge, [7] is the only method which estimates both: the 6D pose, and the symmetry of the perceived object. In particular, the network is trained to also predict the rotational order (*i.e.* the number of identical views), posing it as a classification task.

Generally, most methods for symmetry detection are found in the shape analysis community. Among the different kinds of symmetries, axial symmetries are of particular interest, and multiple approaches have been proposed. Most methods rely on feature matching or spectral analysis: [9] treat the problem as a correspondence matching task between a series of keypoints on an object, determining the reflection symmetry hyperplane as an optimization problem. Elawady *et al.* [10] rely on edge features extracted using a Log-Gabor filter in different scales and orientations coupled with a voting procedure on the computed histogram of local texture and color information. In addition, [5] and [35] are also grounded on wavelet-based approaches. Recently, neural network approaches have also been proposed. Ke *et al.* [23] adapt an edge-detection architecture with multiple residual units and successfully apply it to symmetry detection using real-world images.

Notably, all these approaches aim at detecting symmetries of 3D shapes alone, while our focus is to model the ambiguity arising from objects under specific viewpoints with the goal of improving and explaining pose estimation.

3. Methodology

In this section we describe our method for handling symmetries and other ambiguities for object detection and pose estimation in detail. We will first define what we understand as an ambiguity.

3.1. Ambiguity in Object Detection and Pose Estimation

We describe the rigid body transformations $SE(3)$ via the semi-direct product of $SO(3)$ and $\mathbb{R}^3$. While for the latter, we use Euclidean 3-vectors, the algebra $\mathbb{H}_1$ of unit quaternions is used to model the spatial rotations in $SO(3)$. A quaternion is given by

$$\mathbf{q} = q_1\mathbf{1} + q_2\mathbf{i} + q_3\mathbf{j} + q_4\mathbf{k} = (q_1, q_2, q_3, q_4)\,, \qquad (1)$$

with $(q_1, q_2, q_3, q_4) \in \mathbb{R}^4$ and $\mathbf{i}^2 = \mathbf{j}^2 = \mathbf{k}^2 = \mathbf{ijk} = -\mathbf{1}$. We regress quaternions above the $q_1 = 0$ hyperplane and, thus, omit the southern hemisphere, such that any possible 3D rotation can be expressed by only one single quaternion.

Under ambiguities, a direct naive regression of the rotation as a quaternion will lead to poor results, as the network will learn to predict a rotation that is closest to all results in the symmetry group. This prediction can be seen as the (conditional) mean rotation. More formally, in a typical supervised setting we associate images I_i with poses p_i in a dataset (I_i, p_i) where $i \in \{1, \ldots, N\}$. To describe symmetries, we define for a given image I_i, the set $\mathcal{S}(I_i)$ of poses p that all have an identical image

$$\mathcal{S}(I_i) = \{p_j | I_j = I_i\}\,. \qquad (2)$$

Note that in the case of non-discrete symmetries the set $\mathcal{S}$ will contain infinitely many poses, which in turn transforms the sums of S in the following to integrals. For the sake of a simpler notation and a finite training set in practice, we chose to continue with a notion of a finite $|S|$. The naive model $f(I, \theta)$, that directly regresses a pose p' from I, optimizes a loss $\mathcal{L}(p, p')$ by minimizing

$$\theta^* = \operatorname*{argmin}_{\theta} \sum_{i=1}^{N} \mathcal{L}(f_\theta(I_i), p_i) \qquad (3)$$

over the training set. However, due to symmetry, the mapping from I to p is not well defined and cannot be modeled as a function. By minimizing Equation 3, f is learned to predict a pose $\tilde{p}$ approximating all possible poses for this image equally well.

$$f(I_i, \theta^*) = \tilde{p} = \min_{p} \sum_{j=1}^{|\mathcal{S}(I_i)|} \mathcal{L}(p, p_j) \qquad (4)$$

This is an unfavorable result since $\tilde{p}$ is chosen to minimize the sum of all losses towards the different symmetries. In the following section, we will describe how we model these ambiguities inside our method.

3.1.1 Multiple Pose Hypotheses

The key idea behind the proposed method is to model the ambiguity by allowing multiple pose predictions from the network. In order to predict M pose hypotheses from f, we extend the notation to $f_\theta(I) = (f_\theta^{(1)}(I), \ldots, f_\theta^{(M)}(I))$ where f now returns M pose hypotheses for each image I.

For training, the idea is not to punish all hypotheses given the current pose annotation, since they might be correct under ambiguities. Thus, we use a loss that optimizes only one of the M hypotheses for each annotation. The most intuitive choice is to pick the closest one. We adapt the meta loss $\mathcal{M}$ from [40] that operates on f,

$$\theta^* = \underset{\theta}{\operatorname{argmin}} \sum_{i=1}^{N} \mathcal{M}(f_\theta(I_i), p_i), \tag{5}$$

while we use the original pose loss $\mathcal{L}$ for each $f^{(j)}$

$$\hat{\mathcal{M}}(f_\theta(I), p) = \min_{j=1,\ldots,M} \mathcal{L}(f_\theta^{(j)}(I), p). \tag{6}$$

However, the hard selection of the minimum in equation 6 does not work in practice as some of the hypothesis functions $f_\theta^{(j)}(I)$ might never be updated if they are initialized far from the target values. We relax $\hat{\mathcal{M}}$ to $\mathcal{M}$ by adding the average error for all hypotheses with an epsilon weight:

$$\mathcal{M}(f_\theta(I), p) = \left(1 - \epsilon \frac{M}{M-1}\right) \hat{\mathcal{M}}(f_\theta(I), p) + \frac{\epsilon}{M-1} \sum_{j=1}^{M} \mathcal{L}(f_\theta^{(j)}(I), p). \tag{7}$$

The normalization constants before the two components are designed to give a weight of $(1-\epsilon)$ to $\hat{\mathcal{M}}$ and ϵ to the gradient distributed over all other hypotheses. When $\epsilon \to 0$, $\mathcal{M} \to \hat{\mathcal{M}}$. This is necessary since the average in the second term already contains the minimum from the first one.

3.2. Architecture

We employ SSD-300 [31] with an extended InceptionV4 [43] backbone and adjust it to also provide the 6D pose along with each detection. In particular, we append two more 'Reduction-B' blocks to the backbone. Essentially, we branch off after each dimensionality reduction block and place in total 6.099 anchor boxes to cover objects at different scales. Moreover, to include the unambiguous regression of the 6D pose, we modify the prediction kernel such that it provides $C + M \cdot P$ outputs for each anchor box. Thereby, C denotes the number of classes, M denotes the number of hypotheses, and P denotes the number of parameters to describe the 6D pose. In our case, for each of the M predicted hypotheses, we regress $P = 5$ values to characterize the 6D pose, composed of an explicitly normalized 4D quaternion for the 3D rotation and the object's distance towards the camera. We can estimate the remaining two degrees-of-freedom by back-projecting the center of the 2D bounding box using the inferred depth.

Additionally, in line with [32, 24] we conduct hard negative mining to deal with foreground-background imbalances. Thus, given a set of positive boxes *Pos* and hard-mined negative boxes *Neg* for a training image, we minimize the following energy function:

$$\mathcal{L}(\text{Pos}, \text{Neg}) := \sum_{b \in \text{Neg}} \mathcal{L}_{class} + \sum_{b \in \text{Pos}} (\mathcal{L}_{class} + \alpha \mathcal{L}_{fit} + \beta \mathcal{M}(f_\theta(I), p)). \tag{8}$$

For the class and the refinement of the anchor boxes, we employ the cross-entropy loss $\mathcal{L}_{class}$ and the smooth L1-norm $\mathcal{L}_{fit}$, respectively. In order to compare the similarity of two quaternions, we compute the angle between the estimated rotation and the ground truth rotation according to

$$\mathcal{L}_{rotation}(q, q') = \arccos\left(2\langle q, q'\rangle^2 - 1\right). \tag{9}$$

Additionally, we employ the smooth L1-norm as loss for the depth component $\mathcal{L}_{depth}$. Altogether, we define the final loss for each hypothesis j and input image I as follows

$$\mathcal{L}(f_\theta^{(j)}(I)) = \mathcal{L}_{rotation}(q^{(j)}, q') + \lambda \mathcal{L}_{depth}(d^{(j)}, d'). \tag{10}$$

3.3. Processing Multiple Hypotheses

During inference we further analyze the predicted multiple hypotheses in order to determine whether the pose of the object is ambiguous. Notice that prior to this, we first map all hypotheses to reside on the upper hemisphere. If we detect an ambiguity, we additionally exploit the multiple hypotheses to estimate the view-dependent axes of ambiguity.

Detection of Visual Ambiguities in Scenes. We analyze the distribution of predicted hypotheses in quaternion space to determine whether the pose exhibits an ambiguity. To this end, Principal Component Analysis (PCA) is performed on the quaternion hypotheses $\mathbf{q}_i$. The singular value decomposition of the data matrix indicates the ambiguity: if the dominant singular values $\sigma_{1/2} \gg 0$ ($\sigma_i > \sigma_{i+1}\ \forall i$), an ambiguity in the pose prediction is likely, while small singular values imply a collapse to a single unambiguous solution.

We determine the existence of ambiguity by thresholding the value of σ_2. Empirically, we find the criteria $\sigma_2 > 0.8$ to offer good estimations for ambiguity. It is noteworthy that we can learn to detect ambiguities without further supervision, directly from standard datasets.

Estimation of the Axis of Ambiguity. As mentioned, very prominent representatives for visual ambiguities are symmetries in the objects of interest, as illustrated in Fig. 3 (left) and (mid). Nevertheless, for other objects such as

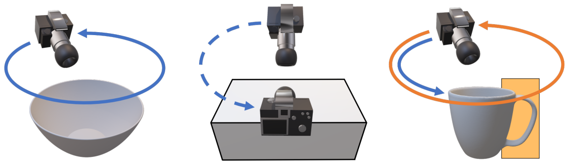


Figure 3: **Examples of pose ambiguity.** Left: Rotational ambiguity. Mid: Two different possible poses for each side. Right: Ambiguity around an arc through (self-) occlusion.

cups, also (self-) occlusion can induce ambiguities in appearance (right).

To calculate a viewpoint dependant ambiguity axis, we take a closer look at the following scenario. A rotation $\mathbf{q}_i = (q_{i1}, q_{i2}, q_{i3}, q_{i4})$ rotates the camera c_0 to c_i around the rotation axis

$$a_i = (q_{i2}, q_{i3}, q_{i4}) \,/\sqrt{q_{i2}^2 + q_{i3}^2 + q_{i4}^2}. \tag{11}$$

All these rotation axes lie in the same plane which is perpendicular to the ambiguity axis $s \perp a_i \; \forall i$. Thus, if we stack the rotation axes $A = \left(a_1^T, a_2^T, \cdots, a_n^T\right)$, we can formulate the overdetermined linear equation system $A^T s = 0$. The ambiguity axis can be found as the solution to the optimization problem

$$\min_{s \in \mathbb{R}^3} \left\| A^T s \right\|_p, \tag{12}$$

which we solve for $p = 2$ using SVD.

3.4. From Multiple Hypotheses to 6D Pose

After analyzing the distribution of the hypotheses, we can robustly compute the associated 6D pose for each case.

Unambiguous Object Pose. In case of an unambiguous object pose, we utilize the multiple hypotheses as an input for a geometric median (geodesic L_1-mean [14]) to improve robustness of the overall estimation

$$\mathbf{q}_{\text{gm}} = \underset{\mathbf{q} \in \mathbb{H}_1}{\operatorname{argmin}} \sum_i \mathrm{d}_{\text{geo}} \left(\mathbf{q}_i, \mathbf{q}\right). \tag{13}$$

The iterative calculation follows the Weiszfeld algorithm [47, 13] in the tangent spaces to the quaternion hypersphere [4]. From a statistical perspective, our rotation measures are treated as inputs for an L_1-estimator to robustly detect the geometric median where d_{geo} gives the geodesic distance on the quaternion hypersphere. Note that Gramkow [12] showed that locally, using the Euclidean distance in the ambient, quaternion space well approximates the Riemannian one. In addition, we compute the median depth of all hypotheses. Afterwards, we utilize the center of the 2D detection and backproject it into 3D to obtain the translation and therewith the full 6D pose of the detection.

Ambiguous Object Pose. As the number of possible 3D rotations is finite yet unknown, we employ mean shift [6] to cluster the hypotheses in quaternion space. Specifically, we use the the angular distance of the quaternion vectors to measure similarity and the Weiszfeld algorithm to merge clusters inside mean shift. This yields either one cluster (if the poses are connected) or multiple (if they are unconnected) as illustrated in Fig. 3. For each cluster we compute a median rotation and the median depth to retrieve the associated 3D translation. Note that we only consider the depths of the hypotheses, which contributed to the corresponding cluster. We apply simple contour checks [24] to find the best fitting cluster from which we extract the final 6D pose.

Synthetic Data. As noted in [19], domain adaptation between synthetically generated data samples and real-world images trivializes the collection of training data. We render CAD models in random poses and add a series of augmentations, such as illumination changes, shadows and blur, as well as background images taken from the MS COCO [30].

4. Evaluation

In this section, we first introduce our experimental setup. Following that, we clearly demonstrate the benefits of our method compared to typical pose estimation systems on a toy dataset. Next, we show robustness in determining whether a view exhibits an ambiguity. Fourth, we report our 6D pose estimation accuracy for the unambiguous and the ambiguous case on common benchmark datasets. Finally, we demonstrate how we can model reliability in pose estimation by analyzing the variance across hypotheses.

4.1. Experimental Setup

Evaluation metrics. In order to properly assess the 6D pose performance, we distinguish between potentially ambiguous and non-ambiguous objects. When dealing with non-ambiguous objects, we report the absolute error for the 3D rotation in degrees and 3D translation in millimeters. We also show our accuracy using the Average Distance of Distinguishable Model Points (ADD) metric from [18], which measures if the average deviation of the transformed model points is less than 10% of the object's diameter.

For 'ambiguous' objects we rely on the Average Distance of Indistinguishable Model Points (ADI) metric, which extends ADD for ambiguity, measuring error as the average distance to the *closest* model point [21, 17].

We also show our results for the Visual Surface Similarity (VSS) metric. As [24], we define VSS similar to the Visual Surface Discrepancy (VSD) [21], however, set $\tau = \infty$. Hence, we measure the pixel-wise overlap of the rendered ground truth pose and the rendered prediction, which is not subject to ambiguities.

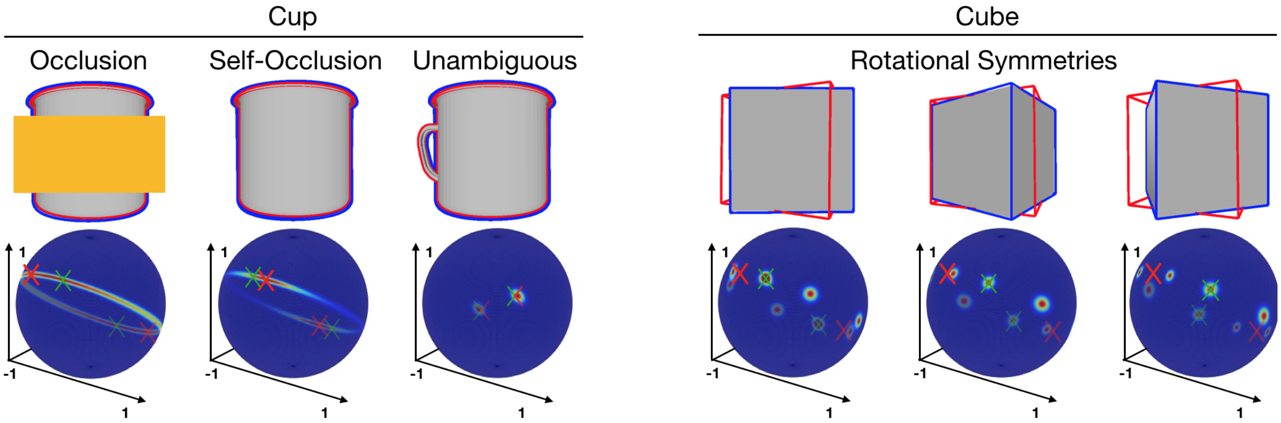


Figure 4: **Synthetic toy dataset.** Top: Contours of the rendered poses for the naive SH (M=1) model in red and our MH (M=30) model in blue. Bottom: Bingham distributions for each pose cluster, together with the ground truth quaternion in green and the SH predicted quaternion in red. Our model is not only correct in both cases but can also predict the full range of valid poses. SH fails on the cube example.

Object	Ambiguity	SH		MH	
		VSS [%]	ADI [%]	VSS [%]	ADI [%]
Cup	(Self-) Occlusion	97.0	**100**	**98.1**	**100**
Cube	Plane Symmetries	87.4	15.6	**98.6**	**100**

Table 1: **Synthetic results.** for the naive SH ($M = 1$) and our MH ($M = 30$) model on the synthetic toy dataset.

Bingham Distributions. In order to visually analyze the multi-hypotheses output of our network, we inspect the underlying rotation distributions. A *Bingham distribution* [1] (BD) is a special equivalent to a Gaussian distribution on a hypersphere. BDs represent a probability distribution on S^d with antipodal symmetry well suited to study poses parametrized by quaternions, where q and $-q \in \mathbb{H}_1$ represent the same element in $SO(3)$. In line with previous works [28, 11, 2], we visualize an equatorial projection of the closest distribution to our pose output using BDs.

4.2. Synthetic Ambiguity Evaluation

We render a simple synthetic dataset of a rotating cup and cube. We compare the baseline with $M = 1$ hypothesis and our method with $M = 30$ hypotheses. The results are shown in Fig. 4, Tab.1, and the supplement. For the cup, both methods yield an ADI score of 100%. The single hypothesis approach SH is indeed able to compute visually correct poses even though it cannot model the pose distribution along an arc. It has learned the conditional mean pose where the handle is exactly opposite of the camera. Nonetheless, this is only one of the infinitely many possible solutions. In contrast, our method is able to predict the whole distribution as seen in the Bingham plots. This is essential for tasks such as next-best-view prediction or robotic manipulation. When there is no ambiguity, both methods predict only the one correct pose.

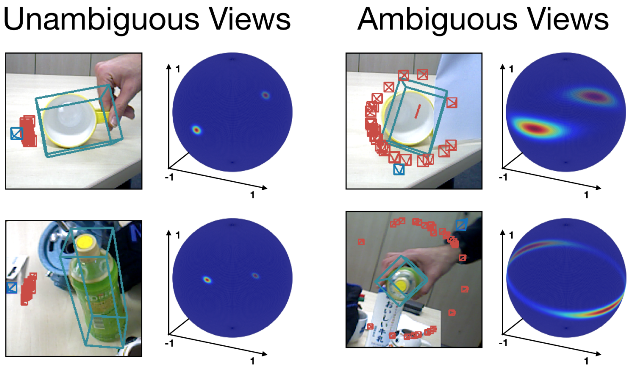


Figure 5: **Real data.** The red frustums visualize ($M = 30$) pose hypotheses. The blue frustum constitutes the median, which determines the predicted 3D bounding box. In the unambiguous case (left) the hypotheses agree. However, partial symmetries and occlusion lead to multiple possible outcomes on the right, which meaningfully reflect to the Bingham distribution of hypotheses.

For the cube object, SH fails (red outline) with an ADI of only 15.6%. Here, the conditional mean is not inside the set of correct poses. Our method is again able to estimate the underlying distribution and can correctly estimate all four modes of correct poses. This yields a perfect ADI of 100%.

When applying our method to real data (Fig. 5), we achieve similar results. If there is a unique solution, the method is able to robustly estimate the correct pose. For ambiguous views, we retrieve the governing distribution as depicted by the viewpoint frustums and spherical plots.

4.3. Real World Datasets

To conduct evaluations on real data, we build two datasets addressing both *unambiguous* and *ambiguous*

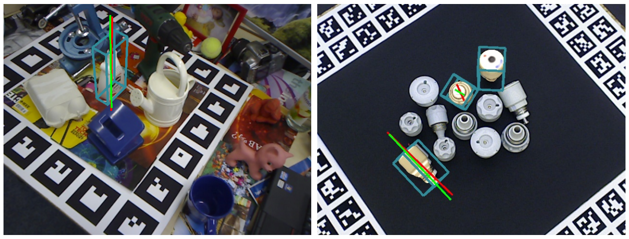


Figure 6: **Ambiguity detection.** Symmetry axis (green line) estimation. Notice that one screw was classified to be unambiguous (*i.e.* no axis), because the ambiguity could be resolved through the texture.

cases. In particular, for the former, we use the popular 'LineMOD' [18] and 'LineMOD Occlusion' dataset [27]. The authors of [27] selected one sequence from the original 'LineMOD' dataset and labeled eight additional objects. Nevertheless, we moved the 'glue' and 'eggbox' object to the *ambiguous dataset*, since both exhibit several views (mostly from the top), which are not unique. Additionally, following [24, 37] we removed the 'cup' and 'bowl' objects, because no watertight CAD models are provided for them. We also discard the 'lamp' since the CAD model does not possess correct normal vectors for proper rendering. To the latter, the *ambiguous* dataset, besides the 'glue' and 'bowl' objects, we added several models from T-LESS [20] to cover different types of ambiguities. In essence, T-LESS mostly consists of symmetric and textureless industrial objects. For our experiments we choose a subset that covers both cases: complete rotational symmetry along an axis (object 4) and objects with more than one rotational symmetry (object 5, 9, 10).

4.4. Ambiguity Detection Analysis

To evaluate the ability of our model to learn pose distributions, we manually labeled for each validation image of the *ambiguous* dataset, whether the current object view exhibits ambiguity based on the visible object texture and shape. This ground truth is used to quantitatively assess our capability of detecting pose ambiguity. Additionally, we compute the ground truth symmetry axis for each object. It is important to note that we do not conduct object symmetry detection, instead, we describe the perceived pose ambiguity in terms of a symmetry axis. These annotations are only used for evaluation and not during training.

For each detected ambiguity, we compute the average discrepancy of the computed symmetry axis from the ground truth annotation. For the ambiguity-free case, we achieve to report an accuracy of more than 99%, while for the ambiguous case we can also state a high accuracy of 82% correctly classified views. Furthermore, the mean axis only deviates by 24°, which shows that our formulation is able to precisely explain the perceived ambiguity.

	Rot. [°]	Trans. [mm]	VSS [%]	ADD [%]	F1
SSD-6D [24]	28.0	72.4	67.4	9.4	88.8
[42]	–	–	–	22.1	–
SH ($\mathbf{M} = \mathbf{1}$)	17.9	45.6	76.8	31.2	91.6
MH ($\mathbf{M} = \mathbf{5}$)	**17.4**	**39.5**	**78.2**	**35.3**	**93.4**

Table 2: **Pose errors of unambiguous objects with synthetic training data**. Comparison with [42], [24]. Results of [24] from their released models and code.

	ape	can	cat	dril	duck	holep	mean
Tekin [46]	2.5	17.5	0.7	7.7	1.1	5.5	5.8
MH ($\mathbf{M} = \mathbf{5}$)	**5.9**	**22.4**	**4.2**	**32.0**	**12.2**	**17.0**	**15.6**

	BB-8 [37]	Tekin [46]	MH ($\mathbf{M} = \mathbf{5}$)
ADD [%]	45.9	**47.9**	44.4

Table 3: **Pose errors of unambiguous objects with real training data split from [3]**. Top: Comparison with [46] on LineMOD Occlusion. Bottom: Comparison with [37] and [46] on LineMOD. Results of [46] from their released models and code.

In Fig. 6, we respectively show one sample of estimated ambiguity axis from 'LineMOD' and 'T-LESS'. For each detection, we draw the estimated axis in red, while the green line denotes the hand-annotated groundtruth axis.

4.5. Comparison to State-of-the-Art

Unambiguous Pose Estimation. In Tab 2 and Tab 3, we report our results for the *unambiguous* subset for training with synthetic data and with the train data split from [3]. Since the number of predicted hypotheses $\mathbf{M}$ is a hyperparameter, we will show an ablation in the supplement and only report our best results with $\mathbf{M} = \mathbf{5}$ here.

For the case of synthetic training only, even for the single hypothesis case, our approach outperforms SSD-6D by more than 35% of relative error while also being more robust in terms of 2D detection. Comparing with Sundermeyer *et al.* [42] we can report a relative improvement of approximately 50% referring to ADD. In addition, our averaging over all hypotheses leads to more robustness towards outliers and, thus, another improvement of all metrics.

When also employing real data, we can improve our results by approximately 9% to 44.4% and are on par with the state-of-the-art methods from [37] and [46], even though we employ no crop and paste augmentations. Further, when using the more challenging 'LineMOD Occlusion' dataset, we can exceed Tekin *et al.* [46] for all objects and overall almost triple their ADD score from 5.8% to 15.6%.

Ambiguous Pose Estimation. Referring to Tab 4, for the ambiguous 'LineMOD' objects, we attain a VSS score of 79% and an ADI score of 55%, which is a relative improvement of approximately 13% and 145% compared to

	VSS [%]			ADI [%]			F1		
	MH	SH	[24]	MH	SH	[24]	MH	SH	[24]
eggbox	**83.1**	78.5	76.3	55.7	**56.0**	26.3	**98.0**	83.0	93.7
glue	**74.6**	74.0	65.1	54.6	**58.7**	17.6	**90.1**	74.0	76.8
mean	**78,9**	76.3	70.7	55.2	**57.4**	22.0	**94.1**	78.5	85.5

		VSS [%]			ADI [%]		
	Scene	MH	SH	[42]	MH	SH	[42]
obj_04	5, 9	70.8	68.6	**78.5**	**19.7**	14.1	15.2
obj_05	2, 3, 4	87.6	82.8	**88.8**	**78.0**	48.3	76.3
obj_09	5, 11	84.4	79.1	**86.5**	69.9	54.5	**77.3**
obj_10	5, 11	82.0	78.5	**82.3**	**57.9**	29.4	31.9
mean		81.2	77.3	**84.0**	**56.4**	36.6	50.6

Table 4: **Ambiguous dataset.** (top: 'LineMOD') (bottom: T-LESS). We compare our multiple hypotheses MH ($M = 30$) results and the same predictor trained to output a single hypothesis SH ($M = 1$) with [42][1] and SSD-6D [24].

SSD-6D. In the 6D setting, the multiple hypothesis detector overall achieves similar performance as the single hypothesis predictor. However, for the 2D detection case, we are able to increase the accuracy from 79% to 94%. As constituted, only a few views are ambiguous for these objects. Investigating the results, we discovered that the single hypothesis predictor is not able to understand exactly these views and tends to simply discard them. In contrast, the multiple hypotheses predictor is indeed able to understand these views and yields reliable pose predictions.

For all ambiguous 'T-LESS' objects (Tab 4), our multiple hypotheses approach surpasses the single hypothesis estimator, which, when trained and evaluated under the same conditions, is not able to capture the ambiguities in pose. Thus, the single hypothesis predictor is not able to produce equally accurate results, being only capable of computing precise poses for unambiguous views.Comparing with [42], we report similar performance in pose. Our ADI improves with 56.4% compared to 50.6% while VSS falls slightly behind by 2.8%. For fairness, we only compare the 6D pose accuracy for correctly detected objects (*i.e.* $IoU \leq 0.5$) since [42] trained their 2D detector for T-LESS on real data.

4.6. Measuring Reliability

To the best of our knowledge, there is no prior work capable of modelling the confidence in the continuous pose estimate. Yet, this information can highly improve the overall robustness and accuracy. In our case, we can utilize the different hypotheses to first determine whether the current view is unambiguous and subsequently employ them as a confidence measurement in the unambiguous 6D pose. To quantify the effect of this, we report our test results on the unambiguous subset of 'LineMOD' in Fig. 7 (top), where we compute a confidence measure via the standard deviation with respect to the Karcher mean [22].

STD σ	Rot. [°]	Trans. [mm]	VSS [%]	ADD [%]	Rejects [%]
< 0.05	11.8	39.4	80.0	37.7	32.6
< 0.075	13.8	41.3	79.1	35.5	18.2
< 0.10	15.5	43.0	78.3	34.3	10.5
< 0.15	17.3	44.0	77.7	33.4	4.0
$< \infty$	19.2	44.8	77.3	32.7	0.0

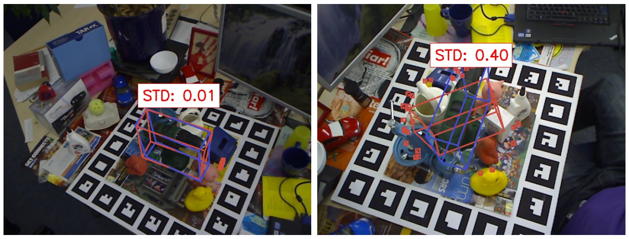


Figure 7: **Reliability.** Top: results for different bins for the standard deviation over all hypotheses for the poses. Bottom: pose with the lowest (left) and the highest (right) standard deviation in the hypotheses. GT pose in blue, predicted pose in red. The red frustums illustrate the hypotheses.

Naturally, a lower standard deviation means more accurate poses. By only allowing poses with $\sigma < 0.1$, all metrics improve, while only losing about 10.5% of all estimates. The rotational error decreases by approximately 20% and the translation error drops from 44.8mm to 43.0mm. Accordingly, using an even lower threshold (*e.g.* $\sigma < 0.05$) gives another significant improvement for pose (especially in rotation), however, at the cost of rejecting more estimates. The qualitative example image in Fig. 7 also confirms these results. The pose with the lowest standard deviation for the 'driller' is very accurate, and the one with the highest is rather imprecise. We experience the same behavior for all *unambiguous* 'LineMOD' objects.

5. Conclusion

We propose a new approach for pose estimation that implicitly models ambiguities without requiring any input preprocessing as well as the feasibility of domain adaptation between synthetic and real data. In addition, we can estimate the axis of rotational ambiguity and perform pose refinement based on clustering without knowing the number of clusters in advance. Our experiments show that our method is suitable for detecting both challenging objects with multiple rotational symmetries and datasets with little ambiguity. Lastly, we argue that our method constitutes a metric of reliability for the 6D pose.

In conclusion, we believe that the new formulation of the pose detection problem from images as an ambiguous task paves the way towards interesting applications in the domain of robotic interactions and automation.

Acknowledgments We would like to thank Toyota Motor Corporation for funding and supporting this work and NVIDIA for the donation of a GPU.

References

[1] Christopher Bingham. An antipodally symmetric distribution on the sphere. *The Annals of Statistics*, pages 1201–1225, 1974.

[2] Tolga Birdal, Umut Simsekli, Mustafa Onur Eken, and Slobodan Ilic. Bayesian pose graph optimization via bingham distributions and tempered geodesic mcmc. In *NeurIPS*, 2018.

[3] Eric Brachmann, Frank Michel, Alexander Krull, Michael Ying Yang, Stefan Gumhold, et al. Uncertainty-driven 6d pose estimation of objects and scenes from a single rgb image. In *CVPR*, 2016.

[4] Benjamin Busam, Tolga Birdal, and Nassir Navab. Camera pose filtering with local regression geodesics on the riemannian manifold of dual quaternions. In *ICCV Workshop*, 2017.

[5] Marcelo Cicconet, Vighnesh Birodkar, Mads Lund, Michael Werman, and Davi Geiger. A convolutional approach to reflection symmetry. *PRL*, 95(1):44–50, 2017.

[6] Dorin Comaniciu, Peter Meer, and Senior Member. Mean shift: A robust approach toward feature space analysis. *TPAMI*, 24:603–619, 2002.

[7] Enric Corona, Kaustav Kundu, and Sanja Fidler. Pose estimation for objects with rotational symmetry. In *IROS*, 2018.

[8] Thanh-Toan Do, Ming Cai, Trung Pham, and Ian D. Reid. Deep-6dpose: Recovering 6d object pose from a single RGB image. *CoRR*, abs/1802.10367, 2018.

[9] Frederik Eaton and Zoubin Ghahramani. Choosing a variable to clamp. In *Artificial Intelligence and Statistics*, 2009.

[10] Mohamed Elawady, Christophe Ducottet, Olivier Alata, Cécile Barat, and Philippe Colantoni. Wavelet-based reflection symmetry detection via textural and color histograms. *ICCV Workshop*, 2017.

[11] Jared Glover and Leslie Pack Kaelbling. Tracking the spin on a ping pong ball with the quaternion bingham filter. In *ICRA*, 2014.

[12] Claus Gramkow. On averaging rotations. *Journal of Mathematical Imaging and Vision*, 15(1-2):7–16, 2001.

[13] Richard Hartley, Khurrum Aftab, and Jochen Trumpf. L1 rotation averaging using the weiszfeld algorithm. In *CVPR*, 2011.

[14] Richard Hartley, Jochen Trumpf, Yuchao Dai, and Hongdong Li. Rotation averaging. *IJCVn*, 103(3):267–305, 2013.

[15] Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross B. Girshick. Mask R-CNN. In *ICCV*, 2017.

[16] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *CVPR*, 2016.

[17] Stefan Hinterstoisser, Stefan Holzer, Cedric Cagniart, Slobodan Ilic, Kurt Konolige, Nassir Navab, and Vincent Lepetit. Multimodal templates for real-time detection of texture-less objects in heavily cluttered scenes. In *ICCV*, 2011.

[18] Stefan Hinterstoisser, Vincent Lepetit, Slobodan Ilic, Stefan Holzer, Gary Bradski, Kurt Konolige, and Nassir Navab. Model based training, detection and pose estimation of texture-less 3d objects in heavily cluttered scenes. In *ACCV*, 2013.

[19] Stefan Hinterstoisser, Vincent Lepetit, Paul Wohlhart, and Kurt Konolige. On pre-trained image features and synthetic images for deep learning. In *ECCV*, 2018.

[20] Tomáš Hodan, Pavel Haluza, Štěpán Obdrzalek, Jií Matas, Manolis Lourakis, and Xenophon Zabulis. T-LESS: An RGB-D dataset for 6D pose estimation of texture-less objects. In *WACV*, 2017.

[21] Tomas Hodan, Jiri Matas, and Stepan Obdrzalek. On Evaluation of 6D Object Pose Estimation. In *ECCV Workshop*, 2016.

[22] Hermann Karcher. Riemannian center of mass and mollifier smoothing. *Communications on pure and applied mathematics*, 30(5):509–541, 1977.

[23] Wei Ke, Jie Chen, Jianbin Jiao, Guoying Zhao, and Qixiang Ye. Srn: Side-output residual network for object symmetry detection in the wild. In *CVPR*, 2017.

[24] Wadim Kehl, Fabian Manhardt, Slobodan Ilic, Federico Tombari, and Nassir Navab. SSD-6D: Making RGB-Based 3D Detection and 6D Pose Estimation Great Again. In *ICCV*, 2017.

[25] Wadim Kehl, Fausto Milletari, Federico Tombari, Slobodan Ilic, and Nassir Navab. Deep Learning of Local RGB-D Patches for 3D Object Detection and 6D Pose Estimation. In *ECCV*, 2016.

[26] Iasonas Kokkinos. Ubernet: Training a 'universal' convolutional neural network for low-, mid-, and high-level vision using diverse datasets and limited memory. In *CVPR*, 2017.

[27] Alexander Krull, Eric Brachmann, Frank Michel, Michael Ying Yang, Stefan Gumhold, and Carsten Rother. Learning Analysis-by-Synthesis for 6D Pose Estimation in RGB-D Images. In *ICCV*, 2015.

[28] Gerhard Kurz, Igor Gilitschenski, Simon Julier, and Uwe D Hanebeck. Recursive estimation of orientation based on the bingham distribution. In *FUSION*, 2013.

[29] Yi Li, Gu Wang, Xiangyang Ji, Yu Xiang, and Dieter Fox. Deepim: Deep iterative matching for 6d pose estimation. In *ECCV*, 2018.

[30] Tsung Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. Microsoft COCO: Common objects in context. In *ECCV*, 2014.

[31] Wei Liu, Dragomir Anguelov, Dumitru Erhan, Christian Szegedy, Scott Reed, Cheng-yang Fu, and Alexander C Berg. SSD : Single Shot MultiBox Detector. In *ECCV*, 2016.

[32] Yuanliu Liu, Zejian Yuan, Badong Chen, Jianru Xue, and Nanning Zheng. Illumination Robust Color Naming via Label Propagation. In *ICCV*, 2015.

[33] Fabian Manhardt, Wadim Kehl, and Adrien Gaidon. Roi-10d: Monocular lifting of 2d detection to 6d pose and metric shape. In *CVPR*, 2019.

[34] Fabian Manhardt, Wadim Kehl, Nassir Navab, and Federico Tombari. Deep model-based 6d pose refinement in rgb. In *ECCV*, 2018.

[35] Maks Ovsjanikov, Jian Sun, and Leonidas Guibas. Global intrinsic symmetries of shapes. *Eurographics Symposium on Geometry Processing*, 27(5):1341–1348, 2008.

[36] Lerrel Pinto and Abhinav Gupta. Supersizing self-supervision: Learning to grasp from 50k tries and 700 robot hours. In *ICRA*, 2016.

[37] Mahdi Rad and Vincent Lepetit. Bb8: A scalable, accurate, robust to partial occlusion method for predicting the 3d poses of challenging objects without using depth. In *ICCV*, 2017.

[38] Joseph Redmon, Santosh Divvala, Ross Girshick, and Ali Farhadi. You only look once: Unified, real-time object detection. In *CVPR*, 2016.

[39] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster R-CNN: towards real-time object detection with region proposal networks. In *NeurIPS*, 2015.

[40] Christian Rupprecht, Iro Laina, Robert DiPietro, Maximilian Baust, Federico Tombari, Nassir Navab, and Gregory D Hager. Learning in an uncertain world: Representing ambiguity through multiple hypotheses. In *ICCV*, 2017.

[41] Karen Simonyan and Andrew Zisserman. Very Deep Convolutional Networks for Large-Scale Image Recognition. *CoRR*, abs/1409.1, 2014.

[42] Martin Sundermeyer, Zoltan-Csaba Marton, Maximilian Durner, Manuel Brucker, and Rudolph Triebel. Implicit 3d orientation learning for 6d object detection from rgb images. In *ECCV*, 2018.

[43] Christian Szegedy, Sergey Ioffe, Vincent Vanhoucke, and Alex A. Alemi. Inception-v4, inception-resnet and the impact of residual connections on learning. In *ICLR Workshop*, 2016.

[44] Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. Going Deeper with Convolutions. In *CVPR*, 2015.

[45] Keisuke Tateno, Federico Tombari, Iro Laina, and Nassir Navab. Cnn-slam: Real-time dense monocular slam with learned depth prediction. In *CVPR*, 2017.

[46] Bugra Tekin, Sudipta N. Sinha, and Pascal Fua. Real-time seamless single shot 6d object pose prediction. In *CVPR*, 2018.

[47] Endre Weiszfeld. Sur le point pour lequel la somme des distances de n points donnés est minimum. *Tohoku Mathematical Journal, First Series*, 43:355–386, 1937.

[48] Paul Wohlhart and Vincent Lepetit. Learning Descriptors for Object Recognition and 3D Pose Estimation. In *CVPR*, 2015.

[49] Bin Xu and Zhenzhong Chen. Multi-level fusion based 3d object detection from monocular images. In *CVPR*, 2018.

[50] Jian Yao, Sanja Fidler, and Raquel Urtasun. Describing the scene as a whole: Joint object detection, scene classification and semantic segmentation. In *CVPR*, 2012.

[51] Xiang Yu, Schmidt Tanner, Narayanan Venkatraman, and Fox Dieter. Posecnn: A convolutional neural network for 6d object pose estimation in cluttered scenes. In *RSS*, 2018.

5.2.2 DB-GAN: Boosting Object Recognition Under Strong Lighting Conditions (WACV 2021)

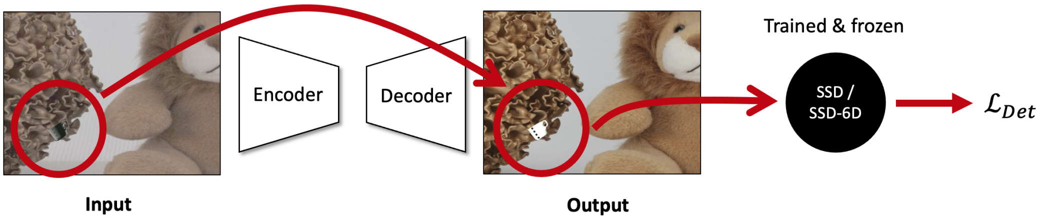


Figure 5.4. **Image Normalization for Object Detection.** Our model is trained to normalize the input image in terms of lighting. To this end, we leverage recent advances in GANs to generate a high quality reconstruction of the input, whilst removing any variations in light. To further improve the normalization, we tailor the reconstruction to work particularly well for the task at hand. To this end, we add a detection term in which we propagate the loss, with respect to the given ground truth, from a pre-trained SSD(-6D) [59, 9] instance to the image normalization network. Due to this, the network is forced to reconstruct the image such that detection is optimized.

The real world has a vast amount of different illumination conditions, which can have a significant impact on object detection. For example, strong directional light from the side can easily lead to misdetections or misclassifications. Moreover, most methods and benchmark datasets only evaluate in well lit setups, which is unrealistic in the real world [131, 120]. While collecting appropriate large-scale datasets is difficult and time-consuming, training on synthetic samples alone introduces a significant domain gap as modeling real light is challenging and requires a lot of computation. In this work, we investigate how an image can be normalized in terms lighting that training on synthetic data is possible without suffering from the domain gap.

There have been several works proposed, attempting at normalizing lighting conditions. Difference of Gaussians based approaches proved to be effective, however, remove most textural information in the process [193]. Other works harnessing GANs either require prior knowledge of the input image [194] or normalize lighting by means of color constancy and image enhancement [195, 196]. Noteworthy, none of these approaches consider the final application to tailor image manipulation. Therefore, we instead propose a novel pipeline which takes an unnormalized RGB image as input and returns a respective normalized image which is optimized to particularly work in the domain of object detection and pose estimation.

In the core, we leverage a GAN-based architecture (*c.f.* Section 2.4) to normalize the input image I to possess uniform lighting $\bar{\mathrm{I}}$. Our generator G follows an encoder-decoder architecture transforming any input image into the lighting normalized domain with $\hat{\mathrm{I}} = \mathrm{G}(\mathrm{I})$. To learn the mapping from the unnormalized to the normalized lighting space, we employ an L1-loss $\mathcal{L}_{recons}$ for reconstruction and harness two discriminators $\mathcal{L}_{gan}$ to ensure high quality and proper domain transfer [159]. We additionally employ a perceptual loss $\mathcal{L}_{perceptual}$ to enforce feature level similarity between the prediction and the ground truth [197]. Nevertheless, while this allows proper domain transfer, the images are not yet tailored towards the actual task. As the goal is to improve object detection and pose estimation, we harness an associated detector and pre-train it on synthetic images without exposing the detector to any variations in lighting. Afterwards, during training of the GAN, we feed the detector with the reconstructed images $\widehat{I}$ and backpropagate the loss of the detector through G. This encourages the network to produce images from the lighting normalized domain in order to re-enable detection

$$\mathcal{L}(\widehat{I}, \overline{I}) = \mathcal{L}_{recons}(\widehat{I}, \overline{I}) + \lambda_1 \mathcal{L}_{perceptual}(\widehat{I}, \overline{I}) + \lambda_2 \mathcal{L}_{gan}(\widehat{I}, \overline{I}) + \lambda_3 \mathcal{L}_{det}(\widehat{I}). \tag{5.5}$$

In practice, we paired our GAN with SSD [59] for 2D object detection and SSD-6D for 6D pose estimation [9] (*c.f.* Section 5.1.1), replacing $\mathcal{L}_{det}$ with the respective objective function.

Another contribution of our work is the release of Toyota TrueBlue, a novel dataset for object detection under white balance variations. Our dataset comes with 11 scenes, each of which recorded under 11 different and quantified color temperatures. We provide annotations with a precise CAD model for 5 objects. We made TrueBlue publicly available at https://forms.gle/5LYFrQAqzAzFKVwj9. We tested an adaptation of DB-GAN on TrueBlue showing the effectiveness of our approach at dealing with color temperature variations.

We performed experiments on the TYO-L and TUD-L datasets from the BOP challenge 2019 [8]. Thereby, our method outperforms all existing works from lighting normalization, most often by a large margin. In addition, the loss study confirms the effectiveness of our detection loss. Finally, we explored how the method can cope with respect to different contrast ratios. It turns out that while the accuracy without normalization varies significantly, our method produces similar results across all contrast ratios, always outperforming the baseline.

Contributions. I proposed and implemented the idea to tailor illumination normalization towards the actual task of 2D object detection by extending a GAN with a 2D object detection loss. Luca Minciullo then extended the initial implementation for the task of 6D pose estimation. Luca Minciullo, Kei Yoshikawa and Sven Meier recorded the TrueBlue dataset and conducted the experiments on TYO-L and TUD-L.

DB-GAN: Boosting Object Recognition Under Strong Lighting Conditions

Fabian Manhardt[2,*], Luca Minciullo[1,*], Kei Yoshikawa[1], Sven Meier[1], Federico Tombari[2], Norimasa Kobori[3]

[2] Toyota Motor Europe
[2] Technical University of Munich
[3] Woven CORE, Inc.
* Equal Contribution

It is the accepted but not the published version of the paper due to copyright restrictions.

Published version: https://doi.org/10.1109/WACV48630.2021.00298

DB-GAN: Boosting Object Recognition Under Strong Lighting Conditions

Luca Minciullo [*]
Toyota Motor Europe
luca.minciullo@toyota-europe.com

Fabian Manhardt[*]
Technical University of Munich
fabian.manhardt@tum.de

Kei Yoshikawa
Toyota Motor Europe
kei.yoshikawa@toyota-europe.com

Sven Meier
Toyota Motor Europe
sven.meier@toyota-europe.com

Federico Tombari[†]
Technical University of Munich
tombari@in.tum.de

Norimasa Kobori
Woven CORE, Inc.
norimasa.kobori@tri-ad.global

Abstract

Driven by deep learning, object recognition has recently made a tremendous leap forward. Nonetheless, its accuracy often still suffers from several sources of variation that can be found in real-world images. Some of the most challenging variations are induced by changing lighting conditions. This paper presents a novel approach for tackling brightness variation in the domain of 2D object detection and 6D object pose estimation. Existing works aiming at improving robustness towards different lighting conditions are often grounded on classical computer vision contrast normalisation techniques or the acquisition of large amounts of annotated data in order to achieve invariance during training. While the former cannot generalise well to a wide range of illumination conditions, the latter is neither practical nor scalable. Hence, We propose the usage of Generative Adversarial Networks in order to learn how to normalise the illumination of an input image. Thereby, the generator is explicitly designed to normalise illumination in images so to enhance the object recognition performance. Extensive evaluations demonstrate that leveraging the generated data can significantly enhance the detection performance, outperforming all other state-of-the-art methods. We further constitute a natural extension focusing on white balance variations and introduce a new dataset for evaluation.

1. Introduction

Due to its wide range of applications, localising objects in natural images is one of the most studied fields in computer vision [41, 55, 26, 20, 52, 12, 13]. Recently, driven by deep learning and the accessibility of large-scale datasets such as ImageNet [6] or Open Images [22], there has been tremendous improvement in terms of detection accuracy as well as the number of objects that can be recognized simultaneously [25, 27, 36, 10, 20, 33, 24]

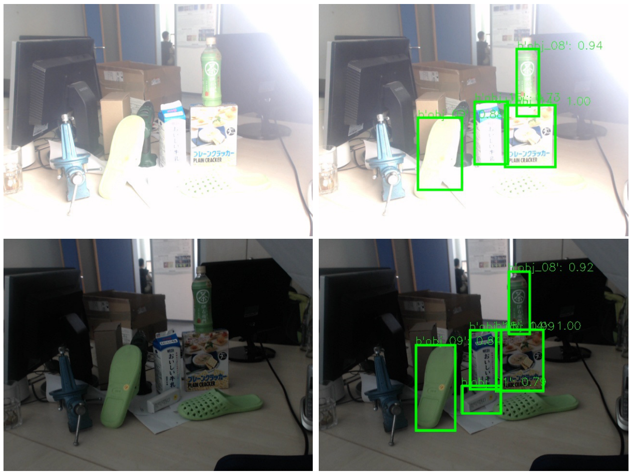


Figure 1. **Detection under strong lighting variations.** Although the input image is subject to strong light from the side, we can still detect almost all objects taken from Toyota Light [12] (top). Similarly, we are able to robustly detect these objects when little light is available (bottom). In each row the input image is shown on the left, while SSD detections are shown on the right.

Despite the undeniable advances, several open challenges still remain to be solved. Some of the most prominent being robustness towards illumination [17, 35, 51], viewpoint changes [28], occlusion, as well as handling the

[*] These authors contributed equally to this work
[†] Federico Tombari is now working at Google

synthetic-to-real domain gap [13, 46].

Real-world environments commonly possess a large variation of illumination conditions. For instance, applications involving outdoor scenes are often exposed to strong changes in illumination. In autonomous driving, cars oftentimes operate in extreme scenarios such as direct strong sunlight during the day or almost no light at night. Similarly, indoor vision systems often suffer from challenging lighting conditions. Noteworthy, nearby windows or inside refrigerators the contrast ratio be 1000:1 or higher. These challenges commonly go unnoticed when training on large scale datasets. However, many practical applications deal with objects categories or instances that are not part of benchmark datasets. Therefore, training data needs to be collected from scratch and the acquisition of data with the required variation is problematic. This is particularly true for 6D pose estimation, since annotating the 6D pose of an object is very difficult, time consuming, and error-prone [13]. For this reason, increasing the capability of models leveraging only synthetic data is of high interest [20, 46, 13, 44]. As a consequence, in this work we focus on robustness towards brightness and color with a particular focus on synthetic data.

In this paper we introduce our novel method to improve 2D object detection and 6D object pose estimation, which we call Detector-Boost GAN(DB-GAN) - a GAN-based architecture for illumination normalisation (*c.f*. Figure 1). Our method is essentially trained to perform illumination normalisation by means of generating images tailored to the capabilities of the object detector. By back-propagating the detection loss, DB-GAN learns to eradicate the weaknesses of the detector and strengthen its performance. Our method does not need prior information on the input image and is able to automatically recover normalised texture under dark, bright as well as non-uniform light conditions. DB-GAN is capable of outperforming all related state-of-the-art methods on two standard benchmark datasets, TUD Light and Toyota Light [12]. We also introduce a new dataset named Toyota TrueBlue, aimed at assessing robustness to white balance changes. Our approach is able to achieve significant mAP improvements on all datasets compared to our baseline detectors and other existing works. Noteworthy, despite focusing on improving detectors, our method can be potentially leveraged to enhance performance of various computer vision tasks.

In summary, we make the following contributions. i) We propose a novel architecture which learns to generate images in order to facilitate further detection under strong illumination changes. ii) We introduce Toyota TrueBlue, a new dataset focusing explicitly on robustness to change in white balance and iii) experimentally demonstrate that DB-GAN significantly enhances performance both in 2D and 3D, outperforming all related methods.

2. Related Work

In this section we provide an overview on previous works in illumination normalisation. Since we employ Generative Adversarial Networks (GANs) to normalize images, we also briefly outline the most important works in the GAN literature.

2.1. Generative Adversarial Networks

Generative Adversarial Networks(GANs) [8] are one of most important recent advances in generative models. GANs train in alternation two deep learning architectures: a generator and a discriminator. While the generator produces realistically looking images, the discriminator attempts to distinguish images coming from the generator from images sampled from the true distribution. The networks are trained jointly in a min-max game fashion, converging in an equilibrium in which the discriminator is not capable of distinguishing real from fake. Inspired by [8], Isola *et al.* employ Conditional GANs [30] for image translation between two domains [14]. Here the generated samples are also conditioned on the input sample, meaning that the discriminator always receives a pair of images. Accordingly, the discriminator is required to distinguish whether the generated output is consistent with the input and correctly translates to the target domain. Similarly, Cycle-GAN, proposed in [59] also carries out domain translation, but without the need for paired data. SINGAN [39] leverages a sequence of generators learning to reconstruct texture at different resolutions and can be trained using a single high resolution training image.

Some existing GAN based works have been introduced in the context of object detection. While, Wang *et al.* [49] leverage GANs for knowledge distillation, Bai *et al.* [2] focuses on improving the detection of small objects. In [58], the authors propose to use weakly supervised object discovery for the detection of vehicles in high resolution remote sensing images. Wang *et al.* [50] propose an adversarial mask generation approach to improve occlusion and deformation robustness in object detection. Finally, other works [7, 15] use GAN generators to produce instance level segmentation masks for either weakly supervised [7] or unpaired data based object detection [15].

Nevertheless, to the best of our knowledge, none of the mentioned works have been used to improve illumination robustness.

2.2. Illumination Normalisation

In this section we specifically cover illumination normalization, image enhancement and color constancy approaches with a special focus on GAN-based solutions.

Local Contrast Normalization [16], was introduced as a pre-processing step to mimic the behaviour of the V1 cells in the cortical area of the brain. A few deep-learning approaches for robustness towards illumination changes have also been proposed. Krizhevsky *et al*. [21] introduce Local Response Normalization as a brightness normalisation module to be applied after non-linearities in deep architectures. Rad *et al*. [35] propose to learn the parameters of a generalization of the Difference-of-Gaussians(DoG) method using CNNs. Thereby, the DoG parameters are learned end-to-end with respect to object detection and 6D object pose estimation. Nonetheless, this method is inherently restricted by the capacity of DoGs for normalisation. Other works [23, 48] perform illumination estimation for modality fusion of thermal and color inputs [23] and image enhancement [48]. Several additional approaches have been introduced for general image enhancement [53, 54, 9, 31].

GAN-based approaches In [42], the authors leverage GANs (*i.e*. Angular-GAN) to remove light and shadows from RGB images. Their method is fully-supervised and uses synthetic training samples generated with GTA-V. Jiang *et al*. propose EnlightenGAN[17] for transforming dark into bright images and vice-versa. The architecture is inspired by Cycle-GAN[59], hence, eradicating the need for paired images during training. However, prior knowledge on whether the input image is *too dark* or *too bright* is required. Furthermore, this method assumes that the input image is acquired under uniform lighting, which is rarely the case in practical scenarios. Wei *et al*. recently introduce Retinex-Net[51], an end-to-end trainable architecture for low-light image enhancement. [51] decomposes the image into reflectance and illumination, prior to adjusting illumination. Nonetheless, they require paired low-light/normal-light data for training. Zhang *et al*. [57] propose a GAN base architecture to deal with illumination robustness in face recognition. They learn a illumination invariant latent space by means of adversarial training. Sakkos *et al*. [38] use two GAN generators to produce both low-light and bright images and then perform semantic segmentation on the difference image in a multi-task setting. Finally, Chen *et al*. [4] propose a GAN-based image enhancement approach.

3. Methodology

In this work, we propose a novel method for illumination normalisation in RGB images. The network is grounded on an Encoder-Decoder architecture, leveraging recent advances in GANs to further enhance the reconstruction quality. The core novelty of this work lies in the additional back-propagation of a detection loss, while training the GAN. This implicitly forces the network to generate images, which simplify latter object detection despite contrary conditions such as very strong illumination. Unlike previous works [17, 51] our method does not require prior knowledge of the input image as well as any real data for training.

In this section, we explain the technical details of our proposed method.

3.1. DB-GAN for Detection-Driven Reconstruction

Let $\mathcal{I}$ be any image space and $\bar{\mathcal{I}}$ be the subset of $\mathcal{I}$ whose elements possess uniform lighting. Assumed an acquired set of image pairs (I, $\bar{I}$), where $I \in \mathcal{I}$, and $\bar{I} \in \bar{\mathcal{I}}$, the illumination normalised version of $\mathcal{I}$. In addition, we assume that all objects of interest are annotated in the form of either bounding boxes or 6D poses. In the following sections we describe how we construct a dataset with these characteristics without any human labelling.

We want to learn a mapping from the domain $\mathcal{I}$ to the illumination-free domain $\bar{\mathcal{I}}$. To this end, we employ a GAN based architecture, following recent success of adversarial models at image generation tasks[59, 39, 14, 42]. To avoid losing details in the reconstructed image [14, 17, 3, 19], our generator $\mathcal{G}$ is based on an encoder-decoder architecture with skip connections [37]. Given an image pair (I, $\bar{I}$) the generator has to learn to normalise the input image according to

$$\hat{I} = \mathcal{G}(I). \tag{1}$$

Since we assume pairs of images, we can learn the mapping from $\mathcal{I}$ to $\bar{\mathcal{I}}$ in a fully supervised fashion, using a reconstruction loss on the target $\bar{I}$ and the prediction $\hat{I}$ with

$$\mathcal{L}_{recons} := ||\hat{I} - \bar{I}||_1. \tag{2}$$

To prevent the generator from predicting blurry outputs we adopt the perceptual loss [18]. In particular, to ensure high and low-level similarity, we extract features ϕ^l at multiple levels L from a VGG16 [43] network trained on ImageNet. We employ the first five ($|L| = 5$) different layers and calculate the perceptual loss using

$$\mathcal{L}_{perceptual} := \frac{1}{|L|} \sum_{l \in L} ||\phi^l(\hat{I}) - \phi^l(\bar{I})||_1. \tag{3}$$

We additionally use an adversarial loss to improve fine-grained reconstruction and ensure proper domain transfer. In particular, we use a discriminator which assesses if a sample in fact originates from the illumination-free domain. In our implementation we use both a global D and a local discriminator LD as proposed in [14]. While the global discriminator encourages better translation to the target domain, the local discriminator operates on small patches in order to enforce the preservation of details. We use binary cross entropy loss for both discriminators. Following common practice [14, 30, 32] we condition the output on input according to $I \oplus \hat{I}$ or $I \oplus \bar{I}$, where $\oplus$ denotes horizontal concatenation [30].

During generator training, we feed the conditioned images to both discriminators for teaching the generator to

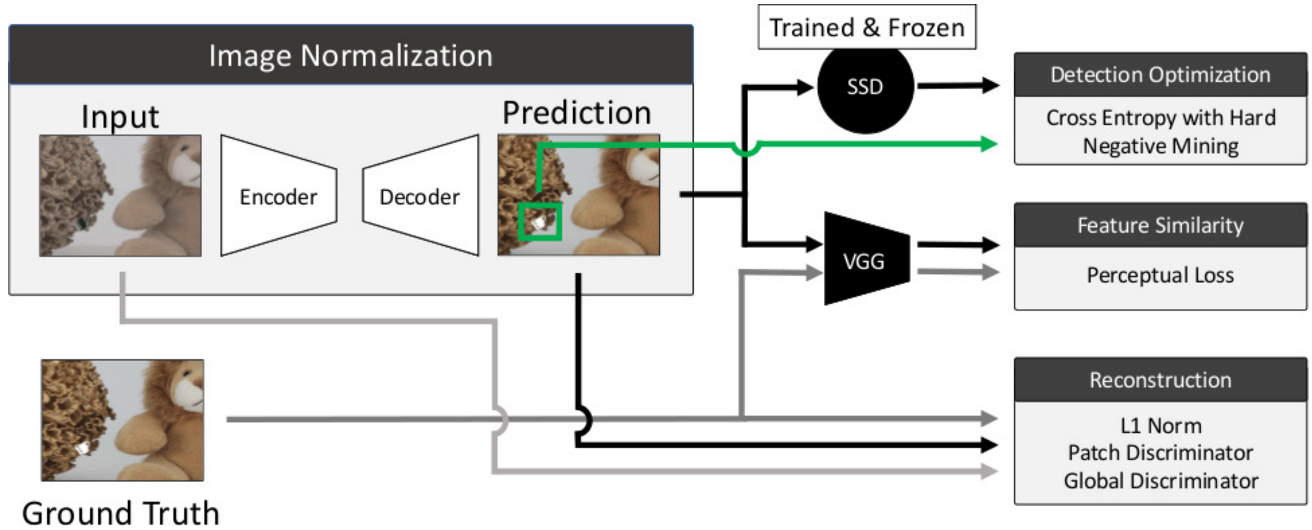


Figure 2. **Training scheme of DB-GAN for Object Detection.** Our loss is based on three different blocks, all intended to optimize detection under high lighting variations. First, a reconstruction term for high quality reconstruction of the normalized target scene $\bar{I}$. Thereby, we incorporate two discriminators ensuring consistency at different scales. Second, a perceptual term to enforce feature similarity between the prediction $\hat{I}$ and the target $\bar{I}$. Finally, a detection term in which we propagate the loss, with respect to the given ground truth (green arrow), from a pre-trained SSD instance to the image normalization network. Due to this, the network is forced to reconstruct the image $\bar{I}$ such that detection is optimized.

produce realistic images that seem to originate from the uniform lighting domain. Again, we use binary cross entropy loss for optimization. We denote these two loss term as $\mathcal{L}_{fool_D}$ and $\mathcal{L}_{fool_{LD}}$.

Unique to this work is the training of the generator with a additional detection loss (*Detection Optimization* as depicted in Fig 2). In essence, we encourage the GAN to not only create realistic illumination normalised images, but to also optimize the image for detection. To this end, we pretrain the detector on synthetic data without any illumination changes and freeze its weights. When training DB-GAN, we additionally back-propagate the loss with respect to the trained detector. DB-GAN is consequently required to adequately adjust lighting in order to optimize detection. To test out the proposed architecture, we use SSD [27] for 2D object detection and SSD6D [20] for 6D object pose estimation. For both detectors we use the original loss terms $\mathcal{L}_{Det}$, as reported in the corresponding papers. Given a set of positive Pos and hard-mined negative Neg anchor boxes, we minimize the following

$$\mathcal{L}_{Det}(\mathrm{Pos}, \mathrm{Neg}) := \sum_{b \in Pos} (L_{class} + \alpha L_{fit}) + \sum_{b \in Neg} L_{class}. \tag{4}$$

with respect to SSD, and for SSD6D according to

$$\mathcal{L}_{Det}(\mathrm{Pos}, \mathrm{Neg}) := \sum_{b \in Neg} L_{class} + \sum_{b \in Pos} (L_{class} + \alpha L_{fit} + \beta L_{view} + \gamma L_{inplane}). \tag{5}$$

Thereby L_{class} denotes the cross-entropy loss applied to each anchor and L_{fit} denotes the L1 loss which measures the misalignment of the corners in order to provide a tight fit. Further, SSD6D decouples 3D rotation into viewpoint and in-plane rotation. Thereby viewpoint describes the perceived surface and inplane rotation describes how this surface is rotated on the image-plane. To increase stability, SSD6D bins viewpoint and in-plane rotation and conducts classification referring again to the cross-entropy loss for L_{view} and $L_{inplane}$.

The final loss for the generator is then comprised of a weighted sum over all individual contributions

$$\mathcal{L} := \mathcal{L}_{recons} + \lambda_1 \mathcal{L}_{GAN} + \lambda_2 \mathcal{L}_{perceptual} + \lambda_3 \mathcal{L}_{fool_D} + \lambda_4 \mathcal{L}_{fool_{LD}} + \lambda_5 \mathcal{L}_{Det} \tag{6}$$

We empirically found that good choices for the above hyper-parameters are: $\lambda_1 = \lambda_2 = \lambda_3 = 1$, $\lambda_4 = 0.5$ and $\lambda_5 = 0.01$.

3.2. Image Enhancement Using DB-GAN

The aim of our approach is to use the trained DB-GAN to generate a new training set enhancing the detector's robustness towards different lighting conditions.

PHOS Dataset. In line with [35], we use the PHOS dataset [47] to train our DB-GAN for illumination robustness. Contrary to [35], we only use PHOS to extract background images. The PHOS dataset [47], contains 15 real scenes, captured under 15 different lighting conditions: one *correct* exposure, 8 images under uniform lighting (*i.e.* 4 underexposed samples and 4 overexposed samples) and 6 samples with non-uniform lighting.

Baseline Detector Data Generation. As we want to back-propagate the detector loss, we need to first train a detector instance capable of detecting all objects of interest. Since we focus on training with synthetic data, we follow standard procedure [20, 29] and render 3D object models with random poses on top of random backgrounds, drawn from the Microsoft Coco dataset [26, 11]. Afterwards, we use the generated data to train the initial detector.

DB-GAN Data Generation. While the background variability in PHOS is limited, it exposes a very high per image resolution of (4256×2832). Considering the input resolution of modern deep learning architectures, this enables the sampling of numerous diverse patches. We use 256×256 as sample size, since it correlates to the input resolution of the DB-GAN generator. We use Laplacian checks to ensure only patches with sufficient textural variation are used. To generate our DB-GAN training data, we render the object models on these PHOS patches. Therefore, we randomly sample an image I from any lighting condition and utilize the matching image with the correct exposure as ground truth $\bar{I}$. We apply several light perturbations on the object model with respect to different OpenGL functionalities and render the result on I. We then re-render the same objects and poses onto the target image, however, without employing any perturbations. We demonstrate two example training examples in the supplementary material. Once the detector baseline and DB-GAN are trained, the detector training data is passed to the generator. The resulting output images form the new, normalised, training data. Finally, a new detector instance is trained on the normalised data.

3.3. Toyota TrueBlue dataset

TrueBlue is a new dataset which specifically targets to assess object detection robustness to white balance errors. Existing image datasets focusing on color temperature [5, 40, 34] do not quantify the illuminant and do not contain household objects with ground truth bounding boxes and 3D object models. We believe this dataset to be the first to be acquired with known light source color temperature and camera settings and, thus, enables quantification of detector performance under erroneous white balance conditions[1].

Toyota TrueBlue (see Figure 3) consists of 11 image sets of 3 different scenes with daily household objects with 3D model, distractor objects and also the MacBeth Color Checker chart. Each scene was illuminated from above by a set of three lights of different types, e.g. LED, incandescent, compact fluorescent, daylight and mixture of different light sources. 11 images were acquired of each scene using a Nikon D750 with Nikon 24-70mm f/2.8 lens with 11 different white balance settings, ranging from 2500K to 10000K. More details on how the dataset was acquired can be found in the supplementary material.

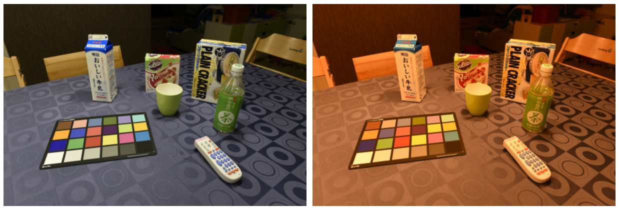


Figure 3. **Example of two color images from the Toyota TrueBlue dataset.** The image on the left has a 2500K color temperature, while the image on the right depicts the same scene at 10000K. More examples can be found in the supplementary material.

4. Evaluation

In this section, we introduce the implementation details, and the datasets used for evaluation. Then we demonstrate the results of our experiments.

4.1. Implementation Details

We generate 50000 training images for both the GAN and the two SSD instances and 100000 training images for SSD6D since it is a more complex task. We train all detectors for 50 epochs. Due to the different number of objects we trained DB-GAN for 10 epochs on the TUD Light objects and for 30 epochs on the Toyota Light objects, with a batch size of 1. The initial learning rate is set to 0.0003 with an exponential update rule. To stabilise training, we do not back-propagate the detector loss until the reconstructions are fairly realistic. Empirically, we found that 30000 iterations are sufficient for this. The experiments were implemented with Tensorflow [1] and run on a single Nvidia TitanXp GPU.

Generator implementation. The generator follows an encoder-decoder architecture using skip connections similar to [37]. We use a 5×5 filter size and leaky ReLU(LReLU) [56], with a 0.2 slope on the negative side, as activation function. The generator consists of eight convolutional layers with stride equal to 2 in the encoder as well as eight deconvolutional layers in the decoder. Each convolutional layer of the encoder is followed by batch normalisation. Further, the encoder has an input image size of 256×256. We use unpooling with zero padding for up-

[1] Toyota Trueblue can be downloaded free-of-charge for non-commercial use by filling the form at https://forms.gle/ZX1aWPiu9HoetKcG9.

sampling. The final up-sampling layer is followed by a hyperbolic tangent activation function to squeeze the output between 0-1 for all channels.

Discriminators implementation. The global discriminator is composed of four convolutional layers. Each convolutional layer is followed by a batch normalisation layer with LReLU activation. Finally, a fully connected layer with sigmoid activation is applied.

The local discriminator first applies a convolutional layer. Afterwards, we extract 64 non-overlapping patches of size 32×32. Each of them is processed by two more convolutional layers followed by a fully connected layer with sigmoid activation. This enforces the output to be also locally consistent within each patch.

Detectors. Our SSD and SSD6D detectors work at 299 × 299 resolution with an InceptionV4 backbone [45] using 6099 anchor boxes. For viewpoint classification in 6D we use 89 view vertices and 36 inplane angles.

4.2. Evaluation Protocol

To assess the performance of our method, we performed experiments on the Benchmark for 6D Object Pose Estimation (BOP) 2019 challenge version [12] of both the Toyota Light and the TUDLight datasets. For the 6D experiments on the Toyota Light dataset we trained all detectors on 4 objects, namely objects 6,9,14 and 15 that we believe well represent the dataset in terms of shape and appearance variation. Note that for all experiments we do not use any of the datasets images during training, but rather train our networks fully from synthetic renderings of the 3D model data. We compare the performance of our approach against the SSD or SSD6D baselines. We additionally compare against three illumination normalisation/image enhancement approaches: the Difference of Gaussians (DoG), EnlightenGAN[17], RetinexNet[51] and Deep Upe[48]. Among classical computer vision approaches DoG still provides top performance on image normalisation for object detection and 6D object pose estimation [35]. In our experiments we used two Gaussian kernels of size 5 and 3 pixels. For all DoG, EnlightenGAN, RetinexNet and Deep Upe we pre-processed the training dataset as well as the input images at inference time.

Finally, we show the effectiveness of our approach at increasing object detection robustness against white balance variation. To achieve this we manually perturbe the hue value of the GAN training images. The hue range $[-15, 15]$ was divided into 4 intervals of equal length. Then a random hue value was sampled in each interval and added uniformly to each image pixel producing 4 new GAN training images. The task of DB-GAN was to reconstruct the original images.

Toyota Light dataset. The Toyota Light dataset [12] contains 21 rigid household objects, captured under 5 different lighting conditions. Noteworthy, the annotation for each input sample includes the actual light conditions at the acquisition time. Two lighting levels are reported. The first is ambient light which is a diffuse overhead light source. The intensity of the incident light on the object was kept constant at 200lx for all samples. The second is the intensity of a directional light source oriented at 90 degrees to the scene. This feature makes this dataset suitable to evaluate non-uniform lighting robustness.

TUD Light Dataset. The TU Dresden Light dataset contains training and test image sequences that show three moving objects under 8 different lighting conditions. The object poses were annotated by manually aligning the 3D object model with the first frame of the sequence and propagating the initial pose through the sequence using ICP.

Metrics. All 2D experiments are evaluated following the standard metric for 2D detection, *i.e.* mean Average Precision(mAP) with a 0.5 IOU threshold. The 6D experiments are evaluated using the BOP 19 challenge toolkit. We report the average recall and report the recall according to all individual BOP metrics in the supplementary materials.

4.3. Qualitative Results

Figure 4 shows qualitative results of our approach for 2D object detection. Both in challenging dark and bright conditions DB-GAN is able to recover images that look almost identical. The SSD trained on DB-GAN generated images can detect a larger number of object instances compared to the SSD baseline. Qualitative comparisons among the different approaches are presented in the supplementary material.

4.4. Quantitative Results

Here we provide a quantitative evaluation of our detection boosting approach compared with existing works.

4.4.1 2D Object Detection

Toyota Light & TUD Light. Table 1 shows the 2D results on the Toyota Light and TUD Light datasets. Our method achieves a mAP of 0.72 on the Toyota Light dataset and 0.66 on the TUD Light dataset, outperforming the SSD baseline as well as all the other approaches. In more detail, we surpass the best existing approach by 0.43 (Deep Upe and EnlightenGAN) on the Toyota Light and by 0.04 (RetinexNet) on TUD Light.

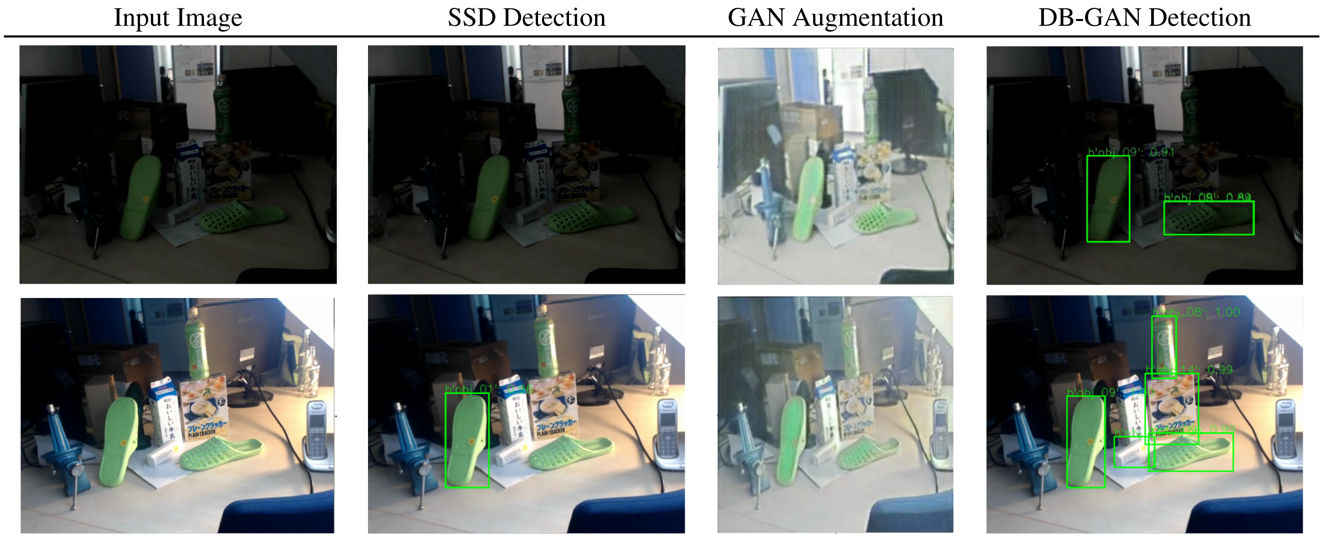


Figure 4. **Comparison of the SSD baseline with our GAN optimized SSD on objects taken from Toyota Light.** Thereby, the second column depicts the results using only SSD and the fourth column shows the corresponding detection employing DB-GAN. It can be easily deduced that our approach significantly improves detection even under difficult lightning conditions. Further, notice that almost all directional light is canceled by the GAN, as illustrated in the intermediate DB-GAN representations (3rd column).

SSD with	Toyota Light mAP ↑	TUD Light mAP ↑
DoG	0.20	0.36
enlightenGAN[17]	0.29	0.43
Retinex-Net[51]	0.28	0.62
Deep Upe [48]	0.29	0.47
baseline	0.27	0.18
DB-GAN	**0.72**	**0.66**

Table 1. **DB-GAN 2D Object Detection results on the Toyota Light and TUD Light datasets.** Our method outperforms the SSD baseline as well all other state-of-the-art approaches for illumination normalisation.

Losses used	mAP ↑
L1	0.55
+ Perceptual	0.67
+ Global Discriminator	0.66
+ Local Discriminator	0.60
+ SSD Loss	**0.72**

Table 2. **DB-GAN loss ablation study on Toyota Light..** These results show that the best performance is achieved when using the proposed combination of loss terms.

SSD with	Toyota TrueBlue mAP ↑
baseline	0.39
baseline w/ augmentation	0.54
DB-GAN	**0.73**

Table 3. **DB-GAN results on the Toyota TrueBlue dataset.** Our method outperforms the baseline as well as SSD when leveraging color augmentations.

Ablation Study. The ablation study was performed on the Toyota Light dataset for 2D object detection. We added the loss terms one by one and report the corresponding mAP. Table 2 shows the results of our ablation study with respect to each loss contribution. Noteworthy, each loss term helps to improve the overall detection performance. Importantly, our main contribution, *i.e.* the back-propagation of the detector loss, constitutes again a significant leap forward in performance, overall giving the best results.

Toyota TrueBlue. Table 3 shows our results on color robustness. We compared our method against the SSD baseline. We additionally compared with standard color augmentation by training a SSD instance on perturbed images. In practice, we perturbed the hue channel of the training images by sampling a random value in the interval $[-15, 15]$ and adding that amount. The results show that our approach achieves a mAP of 0.73, improving on the SSD baseline by almost a factor of two and performing 0.19 better than color augmentation. The supplementary material provides visual examples of detection results for each color temperature.

4.4.2 6D Object Pose Estimation

Toyota Light & TUD Light Table 4 reports the results of our DB-GAN experiments for 6D object pose estimation.

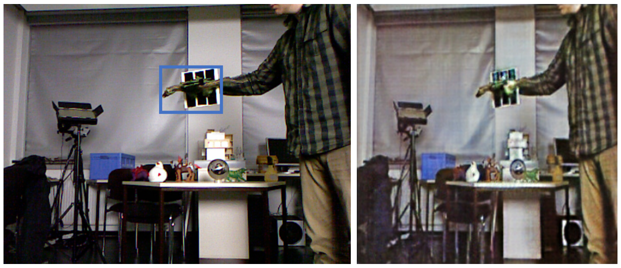


Figure 5. **Examples from the evaluation on TUD Light.** The pair shows a TUD Light image with the corresponding GAN augmentation. Notice how the GAN especially focused on the objects of interest. Nevertheless, the method is also capable of recovering structure in the background, which was almost completely lost due to bad illumination.

SSD6D with	Toyota Light w\o ICP	 w\ICP	TUD Light w\o ICP	 w\ICP
DoG	0.35	0.37	0.14	0.19
enlightenGAN[17]	0.30	0.34	0.157	0.21
Retinex-Net[51]	0.32	0.36	0.13	0.19
Deep Upe [48]	0.34	0.38	0.12	0.18
baseline	0.23	0.32	0.159	0.155
DB-GAN	**0.42**	**0.44**	**0.164**	**0.25**

Table 4. **Results for DB-GAN for 6D object pose estimation on Toyota Light and TUD Light.** Our method outperforms the SSD6D baseline as well all other state-of-the-art approaches for illumination normalisation.

Similar to 2D object detection, our approach improves on the baseline detector as well as all alternative approaches, by a margin of 7.2% on the Toyota Light and 0.5% on the TUD Light. Furthermore, our approach is the only one that significantly improves performance over the baseline on the TUD Light dataset. Additionally, we applied an ICP step to refine the predicted poses. Notice that our approach stays the most competitive. Furthermore, with ICP the gap between our approach and existing methods on TUD Light significantly increases.

4.4.3 Additional Experiments.

Figure 6 shows the performance of the SSD baseline and our boosted SSD on the entire Toyota Light dataset (both train and test sets) as a function of the contrast ratio. Here, contrast ratio is defined as the ratio of the intensity of incident light from the directional light source with respect to the overhead diffuse light source mentioned previously. We observe that the SSD baseline particularly struggles to detect objects in low and high contrast, while after boosting, SSD has become more light invariant, showing roughly the same level of performance for each setting. This shows that our approach is able to improve detection accuracy for both uniform lighting (Contrast Ratio=0) as well as non-uniform lighting (Contrast Ratio=1-10).

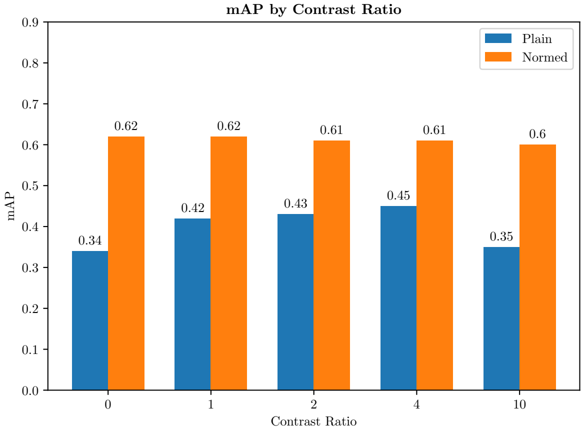


Figure 6. **Comparison between the SSD baseline and the boosted SSD with respect to different contrast ratios in the images.** We report mAP for each contrast ratio value in the Toyota Light dataset. Our approach can deal with both uniform (Contrast Ratio=0) as well as non-uniform lighting (Contrast Ratio=1-10).

DB-GAN pre-processing during inference. While preprocessing training images greatly improves performance, we also want to investigate the use of DB-GAN for preprocessing input images prior to inference. From our experiments we found that in almost all cases this further enhanced the models' capabilities. Nonetheless, when referring to Toyota Light, we surprisingly reveal a small drop in performance. Repetitive textural patterns as well as large flat areas oftentimes degrade the domain transfer capabilities of the GAN, since these samples are eminently different to our training distribution. A qualitative example is shown in Figure 5.

5. Conclusion & Future Work

We presented DB-GAN, a GAN based approach which is able to boost object detection and 6D object pose estimation performance under challenging lighting conditions. The evaluation shows that our method clearly outperforms both the baseline detectors as well as all other state-of-the-art approaches. Further, our method for image normalisation is fully data-driven and neither requires large manually annotated datasets, nor prior knowledge of the input image. Furthermore, our approach is able to deal with non-uniform lighting and does not need prior knowledge of the input image. In the future we want to explore how to expand our methods towards a more diverse set of tasks.

References

[1] Martín Abadi, Ashish Agarwal, Paul Barham, Eugene Brevdo, Zhifeng Chen, Craig Citro, Greg S. Corrado, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Ian Goodfellow, Andrew Harp, Geoffrey Irving, Michael Isard, Yangqing Jia, Rafal Jozefowicz, Lukasz Kaiser, Manjunath Kudlur, Josh Levenberg, Dandelion Mané, Rajat Monga, Sherry Moore, Derek Murray, Chris Olah, Mike Schuster, Jonathon Shlens, Benoit Steiner, Ilya Sutskever, Kunal Talwar, Paul Tucker, Vincent Vanhoucke, Vijay Vasudevan, Fernanda Viégas, Oriol Vinyals, Pete Warden, Martin Wattenberg, Martin Wicke, Yuan Yu, and Xiaoqiang Zheng. TensorFlow: Large-scale machine learning on heterogeneous systems, 2015. Software available from tensorflow.org.

[2] Yancheng Bai, Yongqiang Zhang, Mingli Ding, and Bernard Ghanem. Sod-mtgan: Small object detection via multi-task generative adversarial network. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 206–221, 2018.

[3] Andrew Brock, Jeff Donahue, and Karen Simonyan. Large scale gan training for high fidelity natural image synthesis. *arXiv preprint arXiv:1809.11096*, 2018.

[4] Yu-Sheng Chen, Yu-Ching Wang, Man-Hsin Kao, and Yung-Yu Chuang. Deep photo enhancer: Unpaired learning for image enhancement from photographs with gans. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 6306–6314, 2018.

[5] F. Ciurea and B. Funt. A large image database for color constancy research. In *Proceedings of the Imaging Science and Technology Eleventh Color Imaging Conference*, 2003.

[6] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kehui Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. *2009 IEEE Conference on Computer Vision and Pattern Recognition*, pages 248–255, 2009.

[7] Ali Diba, Vivek Sharma, Rainer Stiefelhagen, and Luc Van Gool. Weakly supervised object discovery by generative adversarial & ranking networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, pages 0–0, 2019.

[8] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In *Advances in neural information processing systems*, pages 2672–2680, 2014.

[9] Chunle Guo, Chongyi Li, Jichang Guo, Chen Change Loy, Junhui Hou, Sam Kwong, and Runmin Cong. Zero-reference deep curve estimation for low-light image enhancement. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 1780–1789, 2020.

[10] Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross Girshick. Mask r-cnn. In *Proceedings of the IEEE international conference on computer vision*, pages 2961–2969, 2017.

[11] Stefan Hinterstoisser, Vincent Lepetit, Paul Wohlhart, and Kurt Konolige. On pre-trained image features and synthetic images for deep learning. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 0–0, 2018.

[12] Tomas Hodan, Frank Michel, Eric Brachmann, Wadim Kehl, Anders GlentBuch, Dirk Kraft, Bertram Drost, Joel Vidal, Stephan Ihrke, Xenophon Zabulis, et al. Bop: Benchmark for 6d object pose estimation. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 19–34, 2018.

[13] Tomáš Hodaň, Vibhav Vineet, Ran Gal, Emanuel Shalev, Jon Hanzelka, Treb Connell, Pedro Urbina, Sudipta N Sinha, and Brian Guenter. Photorealistic image synthesis for object instance detection. In *2019 IEEE International Conference on Image Processing (ICIP)*, pages 66–70. IEEE, 2019.

[14] Phillip Isola, Jun-Yan Zhu, Tinghui Zhou, and Alexei A Efros. Image-to-image translation with conditional adversarial networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1125–1134, 2017.

[15] Heeoh Jang, Dongkyu Kim, Wonhyuk Ahn, and Heung-Kyu Lee. Generative object detection: Erasing the boundary via adversarial learning with mask. In *2019 IEEE 2nd International Conference on Information Communication and Signal Processing (ICICSP)*, pages 495–499. IEEE, 2019.

[16] Kevin Jarrett, Koray Kavukcuoglu, Yann LeCun, et al. What is the best multi-stage architecture for object recognition? In *2009 IEEE 12th international conference on computer vision*, pages 2146–2153. IEEE.

[17] Yifan Jiang, Xinyu Gong, Ding Liu, Yu Cheng, Chen Fang, Xiaohui Shen, Jianchao Yang, Pan Zhou, and Zhangyang Wang. Enlightengan: Deep light enhancement without paired supervision. *arXiv preprint arXiv:1906.06972*, 2019.

[18] Justin Johnson, Alexandre Alahi, and Li Fei-Fei. Perceptual losses for real-time style transfer and super-resolution. In *European conference on computer vision*, pages 694–711. Springer, 2016.

[19] Tero Karras, Samuli Laine, and Timo Aila. A style-based generator architecture for generative adversarial networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4401–4410, 2019.

[20] Wadim Kehl, Fabian Manhardt, Federico Tombari, Slobodan Ilic, and Nassir Navab. Ssd-6d: Making rgb-based 3d detection and 6d pose estimation great again. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 1521–1529, 2017.

[21] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pages 1097–1105, 2012.

[22] Alina Kuznetsova, Hassan Rom, Neil Alldrin, Jasper Uijlings, Ivan Krasin, Jordi Pont-Tuset, Shahab Kamali, Stefan Popov, Matteo Malloci, Tom Duerig, and Vittorio Ferrari. The open images dataset v4: Unified image classification, object detection, and visual relationship detection at scale. *arXiv:1811.00982*, 2018.

[23] Chengyang Li, Dan Song, Ruofeng Tong, and Min Tang. Illumination-aware faster r-cnn for robust multispectral pedestrian detection. *Pattern Recognition*, 85:161–171, 2019.

[24] Zhigang Li, Gu Wang, and Xiangyang Ji. Cdpn: Coordinates-based disentangled pose network for real-time

rgb-based 6-dof object pose estimation. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 7678–7687, 2019.

[25] Tsung-Yi Lin, Priya Goyal, Ross Girshick, Kaiming He, and Piotr Dollár. Focal loss for dense object detection. In *Proceedings of the IEEE international conference on computer vision*, pages 2980–2988, 2017.

[26] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. Microsoft coco: Common objects in context. In *European conference on computer vision*, pages 740–755. Springer, 2014.

[27] Wei Liu, Dragomir Anguelov, Dumitru Erhan, Christian Szegedy, Scott Reed, Cheng-Yang Fu, and Alexander C. Berg. SSD: Single shot multibox detector. In *ECCV*, 2016.

[28] Fabian Manhardt, Diego Martin Arroyo, Christian Rupprecht, Benjamin Busam, Tolga Birdal, Nassir Navab, and Federico Tombari. Explaining the ambiguity of object detection and 6d pose from visual data. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 6841–6850, 2019.

[29] Fabian Manhardt, Wadim Kehl, Nassir Navab, and Federico Tombari. Deep model-based 6d pose refinement in rgb. In *The European Conference on Computer Vision (ECCV)*, September 2018.

[30] Mehdi Mirza and Simon Osindero. Conditional generative adversarial nets. *arXiv preprint arXiv:1411.1784*, 2014.

[31] Sean Moran, Pierre Marza, Steven McDonagh, Sarah Parisot, and Gregory Slabaugh. Deeplpf: Deep local parametric filters for image enhancement. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 12826–12835, 2020.

[32] Augustus Odena, Christopher Olah, and Jonathon Shlens. Conditional image synthesis with auxiliary classifier gans. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, pages 2642–2651, 2017.

[33] Kiru Park, Timothy Patten, and Markus Vincze. Pix2pose: Pixel-wise coordinate regression of objects for 6d pose estimation. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 7668–7677, 2019.

[34] Andrew Blake Tom Minka Toby Sharp Peter Gehler, Carsten Rother. Bayesian color constancy revisited. In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2008.

[35] Mahdi Rad, Peter M. Roth, and Vincent Lepetit. Alcn: Adaptive local contrast normalization for robust object detection and 3d pose estimation. In *BMVC 2017*, 2017.

[36] Joseph Redmon and Ali Farhadi. Yolov3: An incremental improvement. *arXiv preprint arXiv:1804.02767*, 2018.

[37] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. In *International Conference on Medical image computing and computer-assisted intervention*, pages 234–241. Springer, 2015.

[38] Dimitrios Sakkos, Edmond SL Ho, and Hubert PH Shum. Illumination-aware multi-task gans for foreground segmentation. *IEEE Access*, 7:10976–10986, 2019.

[39] Tamar Rott Shaham, Tali Dekel, and Tomer Michaeli. Singan: Learning a generative model from a single natural image. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 4570–4580, 2019.

[40] Lilong Shi and Brian Funt. Re-processed version of the gehler color constancy dataset of 568 images, 2010.

[41] Xuepeng Shi, Shiguang Shan, Meina Kan, Shuzhe Wu, and Xilin Chen. Real-time rotation-invariant face detection with progressive calibration networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2295–2303, 2018.

[42] Oleksii Sidorov. Conditional gans for multi-illuminant color constancy: Revolution or yet another approach? In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, pages 0–0, 2019.

[43] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.

[44] Martin Sundermeyer, Zoltan-Csaba Marton, Maximilian Durner, Manuel Brucker, and Rudolph Triebel. Implicit 3d orientation learning for 6d object detection from rgb images. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 699–715, 2018.

[45] Christian Szegedy, Sergey Ioffe, Vincent Vanhoucke, and Alexander A Alemi. Inception-v4, inception-resnet and the impact of residual connections on learning. In *Thirty-First AAAI Conference on Artificial Intelligence*, 2017.

[46] Jonathan Tremblay, Thang To, and Stan Birchfield. Falling things: A synthetic dataset for 3d object detection and pose estimation. In *IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, pages 2038–2041, 2018.

[47] Vasillios Vonikakis, Dimitrios Chrysostomou, Rigas Kouskouridas, and Antonios Gasteratos. A biologically inspired scale-space for illumination invariant feature detection. *Measurement Science and Technology*, 24(7):074024, 2013.

[48] Ruixing Wang, Qing Zhang, Chi-Wing Fu, Xiaoyong Shen, Wei-Shi Zheng, and Jiaya Jia. Underexposed photo enhancement using deep illumination estimation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 6849–6857, 2019.

[49] Wanwei Wang, Wei Hong, Feng Wang, and Jinke Yu. Gan-knowledge distillation for one-stage object detection. *IEEE Access*, 8:60719–60727, 2020.

[50] Xiaolong Wang, Abhinav Shrivastava, and Abhinav Gupta. A-fast-rcnn: Hard positive generation via adversary for object detection. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2606–2615, 2017.

[51] Chen Wei, Wenjing Wang, Wenhan Yang, and Jiaying Liu. Deep retinex decomposition for low-light enhancement. *arXiv preprint arXiv:1808.04560*, 2018.

[52] Yu Xiang, Tanner Schmidt, Venkatraman Narayanan, and Dieter Fox. Posecnn: A convolutional neural network for 6d object pose estimation in cluttered scenes. *arXiv preprint arXiv:1711.00199*, 2017.

[53] Kai-Fu Yang, Xian-Shi Zhang, and Yong-Jie Li. A biological vision inspired framework for image enhancement in poor

visibility conditions. *IEEE Transactions on Image Processing*, 29:1493–1506, 2019.

[54] Qing Zhang, Yongwei Nie, Lei Zhu, Chunxia Xiao, and Wei-Shi Zheng. Enhancing underexposed photos using perceptually bidirectional similarity. *IEEE Transactions on Multimedia*, 2020.

[55] Shanshan Zhang, Jian Yang, and Bernt Schiele. Occluded pedestrian detection through guided attention in cnns. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 6995–7003, 2018.

[56] Xiaohu Zhang, Yuexian Zou, and Wei Shi. Dilated convolution neural network with leakyrelu for environmental sound classification. In *2017 22nd International Conference on Digital Signal Processing (DSP)*, pages 1–5. IEEE, 2017.

[57] Yang Zhang, Changhui Hu, and Xiaobo Lu. Il-gan: Illumination-invariant representation learning for single sample face recognition. *Journal of Visual Communication and Image Representation*, 59:501–513, 2019.

[58] Kun Zheng, Mengfei Wei, Guangmin Sun, Bilal Anas, and Yu Li. Using vehicle synthesis generative adversarial networks to improve vehicle detection in remote sensing images. *ISPRS International Journal of Geo-Information*, 8(9):390, 2019.

[59] Jun-Yan Zhu, Taesung Park, Phillip Isola, and Alexei A Efros. Unpaired image-to-image translation using cycle-consistent adversarial networks. In *Proceedings of the IEEE international conference on computer vision*, pages 2223–2232, 2017.

5.3 Category-level 6D Object Pose Estimation

Whilst the performance of 6D pose estimation keeps steadily increasing, contemporary methods can only cope with a handful of specific object instances. In fact, most methods even train separate networks for each individual object [120, 9], or greatly relax the problem as for instance estimating pose only up-to-scale [129]. This naturally hampers possible applications as, for instance, robots seamlessly integrated into everyday processes necessarily require the ability to work with hundreds of different objects in metric scale.

5.3.1 ROI-10D: Monocular Lifting of 2D Detection to 6D Pose And Metric Shape (CVPR 2019)

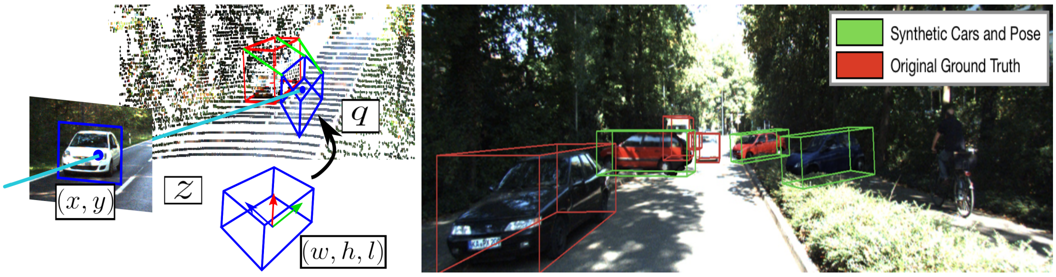


Figure 5.5. **Lifting 2D Objects to 6D Pose and Metric Scale.** Left: After lifting our 2D detections to 6D pose and metric scale, we instantiate the associated 3D bounding box and directly compare it against the associated groundtruth box. This avoids common pitfalls due to weighting different loss terms and directly optimizes for the desired goal, the best alignment in 3D. Right: After learning a shape space of cars using a 3D Convolutional AutoEncoder, we autolabel all images from the KITTI3D dataset and leverage the annotations for coherent synthesis of new training images.

For autonomous driving it is crucial to reliably estimate the pose of all actors without relying on any given mesh. To accomplish this, most works leverage Lidar sensors to provide a 3D representation of the world as additional input to the pose estimator [182, 198]. Unfortunately, as illustrated in Section 1.1, Lidar sensors are expensive and do not work well for certain scenarios (*e.g.* snow can reflect the outgoing laser). Thus, in this work we want to explore the potential of estimating the 7D pose from monocular data alone. We further want to infer a full 3D representation of all actors in the scene by additionally reconstructing the shape of each object.

Our 2D backbone is based on RetinaNet using a Feature Pyramid Network together with Focal Loss [75]. Grounded on the 2D detections $\mathcal{X}$, parameterized as the 4 corners of the detection bounding box, we propose a novel fully differentiable lifting function $\mathcal{F} : \mathbb{R}^{4x2} \rightarrow \mathbb{R}^{8x3}$, which directly maps from 2D Region of Interest (RoI) to the corresponding 3D bounding box $\mathcal{B} := \{\mathcal{B}_1, ..., \mathcal{B}_8\}$. Given RoI $\mathcal{X}$, our lifting module $\mathcal{F}(\mathcal{X})$ applies RoIAlign [76] at $\mathcal{X}$, followed by individual heads to retrieve the allocentric rotation q_a, described as 4D quaternion, 2D centroid $(x, y)^T$, depth z and metric extents $(w, h, l)^T$. Afterwards we calculate the associated 8 corners $\mathcal{B}_i$ according to

$$\mathcal{B}_i := q \cdot \begin{pmatrix} \pm w/2 \\ \pm h/2 \\ \pm l/2 \end{pmatrix} \cdot q^{-1} + K^{-1} \begin{pmatrix} x \cdot z \\ y \cdot z \\ z \end{pmatrix}, \tag{5.6}$$

with K being the camera intrinsic matrix and q denoting the egocentric rotation after conversion as described in Section 3.2. Our differentiable lifting thus allows to directly measure the misalignment in 3D with respect to the ground truth box $\bar{\mathcal{B}}$ as following $\mathcal{L}(\hat{\mathcal{B}}, \bar{\mathcal{B}}) = \frac{1}{8}\sum_{i\in\{1,\dots,8\}} ||\hat{\mathcal{B}}_i - \bar{\mathcal{B}}_i||$. The 3D alignment loss is visualized on the left of Figure 5.5. Our ablative study on the KITTI3D validation set [30, 118] suggest that this new formulation, leads to more robust results than classical weighting of the individual loss terms, even when additionally learning these hyperparameters [199]. Noteworthy, a couple of follow-up works are grounded on our bounding box loss [200, 117]. Exemplary, Simonelli *et al*. extend this paradigm proposing a disentangled formulation, which further stabilizes learning [117].

A second major contribution is the simultaneous retrieval of textured 3D meshes, enabling further data augmentation by coherent synthesis of new images. Given a set of 3D CAD models, we compute their associated TSDF representations ϕ of size $128 \times 128 \times 256$. We subsequently train a 3D Convolutional AutoEncoder on ϕ to learn a low-dimensional shape space using $\mathcal{L}(E, D, \phi) = |D(E(\phi)) - \phi| + |(||E(\phi)|| - 1)| + |\nabla D(E(\phi))|$, with E denoting the Encoder and D describing the Decoder of the network, and ∇ penalizing jumps on the output level set via total variation. In contrast to other methods relying on PCA [129], our results advocate that an AutoEncoder is capable of capturing more details of the models. Eventually, the shape groundtruth of KITTI3D [30] is labeled in a separate offline stage minimizing a reprojection loss. Therefore, the shape prediction head can be trained in a fully supervised fashion. As aforementioned annotating real data is very costly and additionally leveraging synthetic data comes with a significant loss in performance due to the domain shift. Hence, we propose to harness our previously extracted meshes, which we colorize by projecting their vertices onto the image plane, and render them on top of images extracted from KITTI. As the meshes and scenes are obtained from the real domain, we keep the domain gap as small as possible. To generate new unseen poses and decrease overfitting, we move the meshes in 3D without manipulating their viewpoint and apply minor rotational perturbations in 3D. An example of a synthesized training image is illustrated on the right of Figure 5.5.

Eventually, leveraging our proposed strategy for synthesizing new training data together with our 3D bounding box loss, we were able to double the accuracy of current state-of-the-art methods on the hidden KITTI3D test set, increasing the AP with respect to the 3D IoU metric from 7.08%, 5.18%, 4.68% to 12.30%, 10.30%, 9.39% for easy, medium and hard. Moreover, our qualitative results show that our predicted shapes well align with the corresponding object.

Contributions. Wadim Kehl implemented the ResNet-34 backbone for 2D object detection and the 3D auto-encoder for learning of a latent shape space. I extended the 2D detector with branches for pose and shape and implemented our proposed 3D bounding box loss. I also automatically labeled the training data with shape annotations. I ran all ablations and evaluated on KITTI3D together with the aid from Wadim Kehl.

ROI-10D: Monocular Lifting of 2D Detection to 6D Pose And Metric Shape

Fabian Manhardt[1,*], Wadim Kehl[2,*], Adrien Gaidon[2]

[1] Technical University of Munich
[2] Toyota Research Institute, Los Altos
[*] Equal Contribution

It is the accepted but not the published version of the paper due to copyright restrictions.

Published version: https://doi.org/10.1109/CVPR.2019.00217

ROI-10D: Monocular Lifting of 2D Detection to 6D Pose and Metric Shape

Fabian Manhardt*
TU Munich
fabian.manhardt@tum.de

Wadim Kehl*
Toyota Research Institute
wadim.kehl@tri.global

Adrien Gaidon
Toyota Research Institute
adrien.gaidon@tri.global

Abstract

We present a deep learning method for end-to-end monocular 3D object detection and metric shape retrieval. We propose a novel loss formulation by lifting 2D detection, orientation, and scale estimation into 3D space. Instead of optimizing these quantities separately, the 3D instantiation allows to properly measure the metric misalignment of boxes. We experimentally show that our 10D lifting of sparse 2D Regions of Interests (RoIs) achieves great results both for 6D pose and recovery of the textured metric geometry of instances. This further enables 3D synthetic data augmentation via inpainting recovered meshes directly onto the 2D scenes. We evaluate on KITTI3D against other strong monocular methods and demonstrate that our approach doubles the AP on the 3D pose metrics on the official test set, defining the new state of the art.

1. Introduction

How much can one understand a scene from a single color image? Using large annotated datasets and deep neural networks, the Computer Vision community has steadily pushed the envelope of what was thought possible, not just for semantic understanding but also in terms of 3D properties of scenes and objects. In particular, Deep learning methods on monocular imagery have proven competitive with multi-sensor approaches for important ill-posed inverse problems like 3D object detection ([3, 31, 20, 34, 24], 6D pose tracking [30, 40], depth prediction [9, 11, 13, 42, 33], or shape recovery [18, 23]. These improvements have been mainly accomplished by incorporating strong implicit or explicit priors that regularize the underconstrained output space towards geometrically-coherent solutions. Furthermore, these models benefit directly from being end-to-end trainable in general. This leads to increased accuracy, since networks are discriminatively tuned towards the target objective instead of intermediate outputs followed by non-trainable post-processing heuristics. The main challenge, though, is to design a model and differentiable loss function that lend itself to well-behaved minimization.

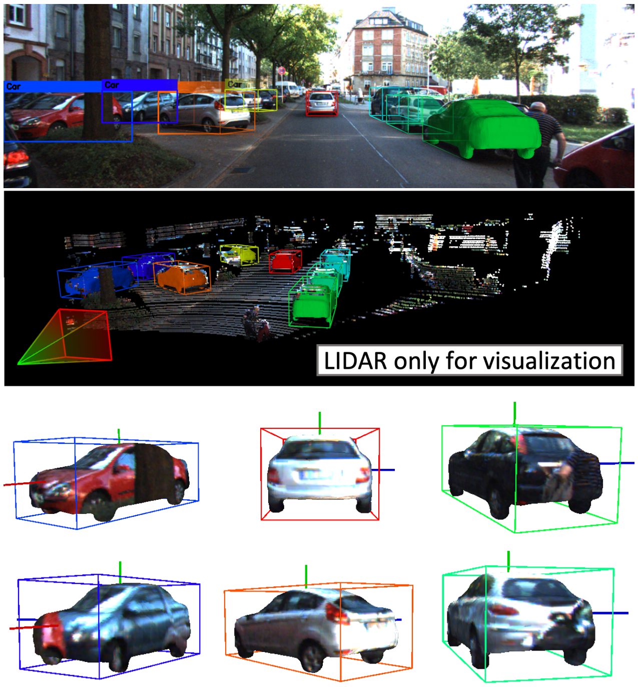


Figure 1. Top (from left to right): our 2D detections, 3D boxes, and meshed shapes inferred from a single monocular image in one forward pass. Middle: our predictions on top of a LIDAR point cloud, demonstrating metric accuracy. Bottom: example well-localized, metrically-accurate, textured meshes predicted by our network.

In this work we introduce a new end-to-end method for metrically accurate monocular 3D object detection, *i.e.* the task of predicting the location and extent of objects in 3D using a single RGB image as input. Our key idea is to regress oriented 3D bounding boxes by lifting predicted 2D Regions of Interest (RoIs) using a monocular depth network. Our main contributions are:

- an end-to-end multi-scale deep network for monocular 3D object detection, including a differentiable 2D to 3D RoI lifting map that internally regresses all required components for 3D box instantiation;

[*] Equal contribution. This work was part of an internship stay at TRI.

- a loss function that aligns those 3D boxes in metric space, directly minimizing their error with respect to ground truth 3D boxes;
- an extension of our model to predict metric textured meshes, enabling further 3D reasoning, including 3D-coherent synthetic data augmentation.

We call our method ”ROI-10D”, as it lifts 2D regions of interests to 3D for prediction of 6 degrees of freedom pose (rotation and translation), 3 DoF spatial extents, and 1 or more DoF shape. Experiments on the KITTI3D [12] benchmarks show that our approach enables accurate predictions from a single RGB image. Furthermore, we show that our monocular 3D poses are competitive or better than the state of the art.

2. Related Work

Since the amount of work on object detection has expanded significantly over the last years, we will narrow our focus to recent advances among RGB-based methods for 3D object detection. 3DOP from Chen *et al*. [4] use KITTI [12] stereo data and additional scene priors to create 3D object proposals followed by a CNN-based scoring. In their follow-up work Mono3D [3], the authors replace the stereo-based priors by exploiting various monocular counterparts such as shape, segmentation, location, and spatial context. Mousavian *et al*. [31] propose to couple single-shot 2D detection with an additional binning of azimuth orientations plus offset regression. Similarly, SSD-6D from Kehl *et al*. [20] introduces a structured discretization of the full rotational space for single-shot 6D pose estimation. The work from Xu *et al*. [41] incorporates a monocular depth module to further boost the accuracy of inferred poses on KITTI.

Instead of discretizing $SO(3)$, [34, 37] formulate the 6D estimation problem as a regression of the 2D projections of the 3D bounding box. These methods assume the scale of the objects to be known and can therefore use a perspective-n-point (PnP) variant to recover poses from 2D-3D correspondences. Grabner *et al*. [14] present a mixed approach where they regress 2D control points and absolute scale to recover pose and, subsequently, the object category. In addition, Rad *et al*. [34] empirically show the superiority of this proxy loss over standard regression of the 6 degrees of freedom. In contrast, [40, 24, 30] directly encode the 6D pose. In particular, Xiang *et al*. [40] first regress the rotation as Euler angles and the 3D translation as the backprojected 2D centroid. Thereafter, they transform the 3D mesh into the camera frame and measure the average distance of the model points [16] towards the ground truth. Similarly, [24] also minimizes the average distance of model points for 6D pose refinement. Manhardt *et al*. [30] also conduct 6D pose refinement but regress a 4D update quaternion to describe the 3D rotation. Their proxy loss samples and transforms 3D contour points to maximize projective alignment.

Notably, all these direct encoding methods require knowledge of the precise 3D model. However, when working at a category-level the 3D models are usually not available, and these approaches are not designed to handle intra-class 3D shape variations (for instance between different types of cars). We therefore propose a more robust way of lifting to 3D that only requires bounding boxes. Thereby, the extents of these bounding boxes can also be of variable size. Similar to us, [7] use RoIs to lift 2D detections, but their pipeline is not trained end-to-end and reliant on RGB-D input for 3D box instantiation.

In terms of monocular shape recovery, 3D-RCNN from Kundu *et al*. [23] uses an RPN to estimate the orientation and shape of cars on KITTI with a render-and-compare loss. Kanazawa *et al*. [18] predict instance shape, texture, and camera pose using a differentiable mesh renderer [19]. While these methods show very impressive results as part of their synthesis error minimization, they recover shapes only up to scale. Furthermore, our approach does not require differentiable rendering or approximations thereof.

3. Monocular lifting to 10D for pose and shape

In this section we describe our method of detecting objects in 2D space and consequently, computing their 6D pose and metric shape from a single monocular image. First, we give an overview of our network architecture. Second, we explain how we lift the loss computation to 3D in order to improve pose accuracy. Third, we describe our learned metric shape space and its use for 3D reconstruction from estimated shape parameters. Finally, we describe how our shape estimation enables 3D-coherent data augmentation to improve detection.

3.1. End-to-end Monocular Architecture

Our architecture (Figure 2) follows a two-stage approach, similar to Faster R-CNN [36], where we first produce 2D region proposals and then run subsequent predictions for each. For the first stage we employ a RetinaNet [26] that uses a ResNet-34 backbone with FPN structure [25] and focal loss weighting. For each detected and precise 2D object proposal, we then use the RoIAlign operation [15] to extract localized features for each region.

In contrast to the aforementioned related works, we do not directly regress 3D information independently for each proposal from these localized features. Predicting this information from monocular data, in particular absolute translation, is ill-posed due to scale and reprojection ambiguities, which the lack of context exacerbates. In contrast, networks that aim to predict global depth information over the whole scene can overcome these ambiguities by leveraging geometric constraints as supervision [11]. Consequently, we use a parallel stream based on the state-of-the-art Su-

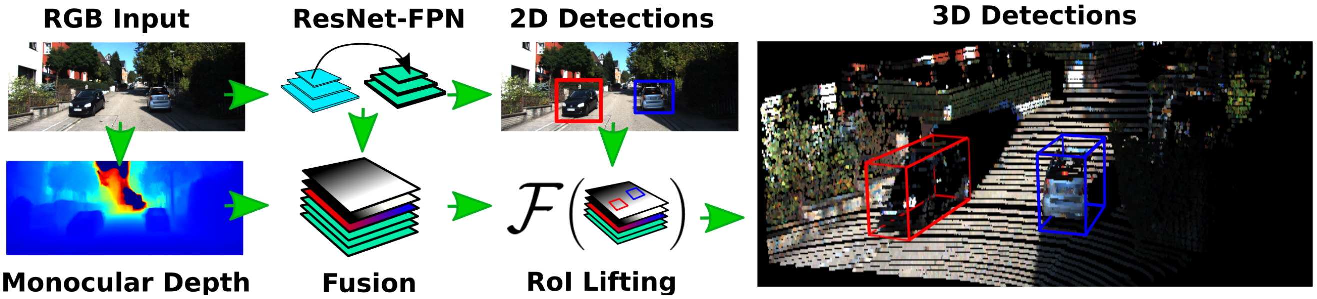


Figure 2. We process our input image with a ResNet-FPN architecture for 2D detection and a monocular depth prediction network. We use the predicted Regions of Interest (RoI) to extract fused feature maps from the ResNet-FPN and depth network via a RoIAlign operation before regressing 3D bounding boxes, a process we call RoI lifting.

perDepth network [33], which predicts per-pixel depth from the same monocular image.

We use these predicted depth maps to support distance reasoning in the subsequent 3D lifting part of our network. Besides the aforementioned localized feature maps from our 2D RPN, we also want to include the corresponding regions in the predicted depth map. For better localization accuracy, we furthermore decided to include a 2D coordinates map [27]. We thus propagate all the information to our fusion module, which consists of two convolutional layers with Group Normalization [39] for each input modality. Finally, we concatenate all features, use RoIAlign and run into separate branches for the regression of 3D rotation, translation, absolute (metric) extents, and object shape, as described in the following sections.

3.2. From Monocular 2D Instance to 6D Pose

Formally, our approach towards the problem is to define a fully-differentiable lifting mapping $\mathcal{F} : \mathbb{R}^4 \rightarrow \mathbb{R}^{8\times3}$ from a 2D RoI $\mathcal{X}$ to a 3D box $\mathcal{B} := \{B_1, ..., B_8\}$ of eight ordered 3D points. We chose to encode the rotation as a 4D quaternion and the translation as the projective 2D object centroid (similar to [31, 20, 40]) together with the associated depth. In addition, we describe the 3D extents as the deviation from the mean extents over the whole data set.

Given RoI $\mathcal{X}$, our lifting $\mathcal{F}(\mathcal{X})$ runs RoIAlign at that position, followed by separate prediction heads to recover rotation q, RoI-relative 2D centroid (x, y), depth z and metric extents (w, h, l). From this we build the 8 corners B_i:

$$B_i := q \cdot \begin{pmatrix} \pm w/2 \\ \pm h/2 \\ \pm l/2 \end{pmatrix} \cdot q^{-1} + K^{-1} \begin{pmatrix} x \cdot z \\ y \cdot z \\ z \end{pmatrix} \qquad (1)$$

with K^{-1} being the inverse camera intrinsics. We build the points B_i in a defined order to preserve absolute orientation. We depict the instantiation in Figure 3.

Our formulation is reminiscent of 3D anchoring (as MV3D [5], AVOD [22]). However, our 2D instantiation of those 3D anchors is sparse and works over the whole image plane. While such 3D anchors explicitly provide the object's 3D location, our additional degree of freedom also requires the estimation of the depth.

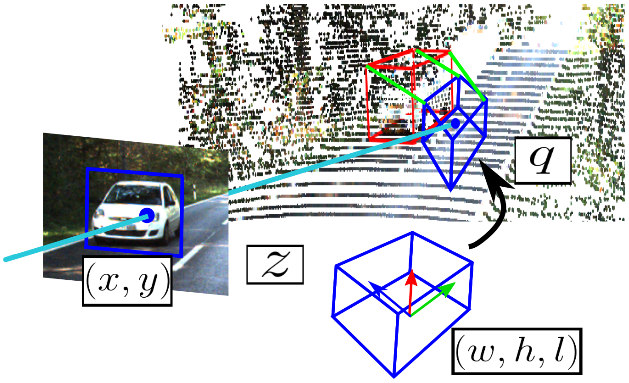


Figure 3. Our lifting $\mathcal{F}$ regresses all components to estimate a 3D box $\mathcal{B}$ (blue). From here, our loss minimizes the pointwise distances towards the ground truth $\mathcal{B}^*$ (red). We visualize three of the eight correspondences in green.

Lifting Pose Error Estimation to 3D When estimating the pose from monocular data only, little deviations in pixel-space can induce big errors in 3D. Additionally, penalizing each term individually can lead to volatile optimization and is prone to suboptimal local minima. We propose to lift the problem to 3D and employ a proxy loss describing the full 6D pose. Consequently, we do not force to optimize all terms equally at the same time, but let the network decide its focus during training. Given a ground truth 3D box $\mathcal{B}^* := \{B_1^*, ..., B_8^*\}$ and its associated 2D detection $\mathcal{X}$ in the image, we run our lifting map to retrieve the 3D prediction $\mathcal{F}(\mathcal{X}) = \mathcal{B}$. The loss itself is the mean over the eight corner distances in metric space:

$$\mathcal{L}(\mathcal{F}(\mathcal{X}), \mathcal{B}^*) = \frac{1}{8} \sum_{i \in \{1..8\}} ||\mathcal{F}(\mathcal{X})_i - \mathcal{B}^*{}_i||. \qquad (2)$$

We depict some of the 3D-3D correspondences that the loss is aligning as green lines in Figure 3.

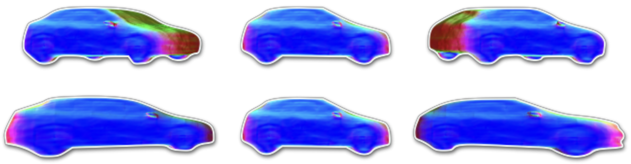


Figure 4. Comparison between egocentric (top) and allocentric (bottom) poses. While egocentric poses undergo viewpoint changes towards the camera when translated, allocentric poses always exhibit the same view, independent of the object's location.

When deriving the loss, the chain rule leads to

$$\left[\frac{\nabla\mathcal{F}(\mathcal{X})}{\nabla q}, \frac{\nabla\mathcal{F}(\mathcal{X})}{\nabla(x,y)}, \frac{\nabla\mathcal{F}(\mathcal{X})}{\nabla z}, \frac{\nabla\mathcal{F}(\mathcal{X})}{\nabla(w,l,h)}\right]\frac{\nabla\mathcal{L}(\cdot)}{\nabla\mathcal{F}(\mathcal{X})}\mathcal{L}(\cdot) \tag{3}$$

and shows clearly the individual impact that each lifting component contributes towards 3D alignment. Similar to work that employ projective or geometric constraints [29, 30], we observe that we require a warm-up period to bring regression into proper numerical regimes. We therefore train with separate terms until we reach a stable 3D box instantiation and switch then to our lifting loss.

We also want to stress that our parametrization allows for general 6D pose regression. Although the object annotations in KITTI3D exhibit only changing azimuths, many driving scenarios and most robotic use cases require solving for all 6 degrees of freedom.

Allocentric Regression and Egocentric Lifting Multiple works [31, 23] emphasize the importance of estimating the allocentric pose for monocular data, especially for larger fields of view. The difference is depicted in Figure 4 where the relative object translation with respect to the camera changes the observed viewpoint. Accordingly, we follow the same principle since RoIs lose the global context. Therefore, rotations q are considered allocentric during regression inside $\mathcal{F}$ and then corrected with the inferred translation to build the egocentric 3D boxes.

3.3. Object Shape Learning & Retrieval

In this section we explain how we extend our end-to-end monocular 3D object detection method to additionally predict meshes and how to use them for data augmentation.

Learning of a Smooth Shape Space Given a set of 50 commercially available CAD models of cars, we created projective truncated signed distances fields (TSDF) ϕ_i of size $128 \times 128 \times 256$. We initially used PCA to learn a low-dimensional shape, similar to [23]. During experimentation we found the shape space to be quickly discontinuous away from the mean, inducing degenerated meshes. Using PCA to generate proper shapes requires to evaluate each dimension according to its standard deviation. To avoid this tedious process, we instead trained a 3D convolutional autoencoder, consisting of encoder E as well as decoder $\mathcal{D}$, and enforced different constraints on the output TSDF. In particular, we employed 4 convolutional layers with filter sizes of 1,8,16,32 for both E and $\mathcal{D}$. In addition, we used a fully-connected layer of 6 to represent the latent space. During training we further map all latent representations on the unit hypersphere to ensure smoothness within the embedding. Furthermore, we penalize jumps in the output level set via total variation, which regularizes towards smoother surfaces. The final loss is the sum of all these components:

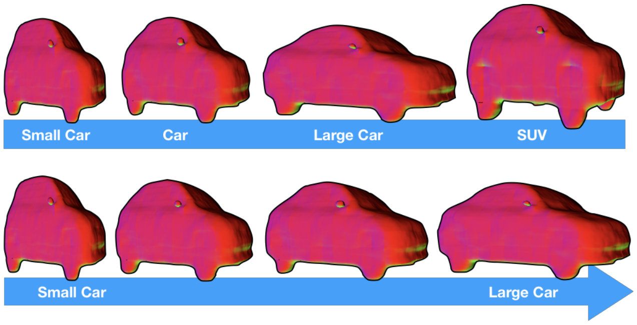


Figure 5. Top: Median of each category in the learned shape space. Bottom: Smooth interpolation on the latent hypersphere between two categories.

$$\begin{gathered}\mathcal{L}_{tsdf}(E,\mathcal{D},\phi) = \\ |\mathcal{D}(E(\phi)) - \phi| + |(||E(\phi)|| - 1)| + |\nabla\mathcal{D}(E(\phi))|\end{gathered} \tag{4}$$

We additionally classified each CAD model as either 'Small Car', 'Car', 'Large Car' or 'SUV'. Afterwards, we computed the median shape over each class, and all cars together, using the Weiszfeld algorithm [38], as illustrated in Fig. 5 (top). Below, we show our ability to smoothly interpolate between the median shapes in the embedding. We observed that we could safely traverse all intermediate points on the embedding without degenerate shapes and found a six-dimensonsal latent space to be a good compromise between smoothness and detail.

Ground truth shape annotation. To avoid gradient approximation through central differences as [23], we labeled the KITTI3D car instances offline. Running greedy search initialized from every median, we seek for the minimal projective discrepancy in LIDAR and segmentation from [15].

For the shape branch of our 3D lifter, we measure the similarity between predicted shape s and ground truth shape s^* as the angle between the two points on the hypersphere.

$$\mathcal{L}_{shape}(s,s^*) = \arccos\left(2\langle s,s^*\rangle^2 - 1\right) \tag{5}$$

During inference we predict the low-dimensional latent vector and feed it to the decoder to obtain its TSDF representation. We can also compute the 3D mesh from the TSDF employing marching cubes [28].

Simple mesh texturing. Since our method computes absolute scale and 6D pose, we conduct projective texturing of the retrieved 3D mesh. To this end, we project each vertex that faces towards the camera onto the image plane and assign the corresponding pixel value. Afterwards, we mirror the colors along the symmetry axis for completion.

3.4. Synthetic 3D data augmentation

Since annotating monocular data with metrically accurate 3D annotations is usually costly and difficult, many recent works leverage synthetic data [10, 8, 2, 1] to train their methods [20, 17, 35]. Nevertheless, this often comes with a significant drop in performance due to the domain gap. This is especially true for KITTI3D, since it is a very small dataset with only around 7k images (or 3.5k images for train and val respectively with the split from [3]). This can easily result in severe overfitting to the training data distribution.

An interesting solution to this domain gap, proposed by Alhaija et al. [1], consists in extending the dataset by inpainting 3D synthetic renderings of objects onto real-world image backgrounds. Inspired by this Augmented Reality type of approach, we propose to utilize our previously extracted meshes in order to produce realistic renderings. This allows for increased realism and diversity, in contrast to using a small set of fixed CAD models as in [1]. Furthermore, we do not use strong manual or map priors to place the synthetic objects in the scene. Instead, we employ the allocentric pose to move the object in 3D without changing the viewpoint. We apply some rotational perturbations in 3D to generate new unseen poses and decrease overfitting. Fig. 6 illustrates one synthetically generated training sample. While the red bounding boxes show the original ground truth annotations, the green bounding boxes depict the synthetically added cars and their sampled 6D pose.

3.5. Implementation details

The method was implemented in PyTorch [32] and we employed AWS p3.16xlarge instances for training. We used SGD with momentum, a batch size of 8 and a learning rate of 0.001 with linear warm-up. We ran a total of 200k iterations and decayed the learning rate after 120k and 180k steps by 0.1. We employed both scale-jittering and horizontal flipping to augment the dataset. For the synthetic car augmentations, we extracted in total 140 meshes from the training sequences, which we textured using the corresponding ground truth poses. We then augmented each input sample with up to 3 different cars by shooting rays in random directions and sampling a 3D translation along the ray. Additionally, we employed the original allocentric rotation to avoid textural artifacts, however, perturbed the rotations up to 10 degrees in order to always produce new unseen 6D poses. Our shape space is six-dimensional although smaller dimensionality can lead to well-behaving spaces, too. We show qualitative results in the supplement. During testing, we resize the shorter side of the image to 600 and run 2D detection. We filter the detections with 2D-NMS at 0.65 before RoI-lifting. The resulting 3D boxes are then processed by a very strict Bird's Eye View-NMS at 0.05 that prevents physical intersections.

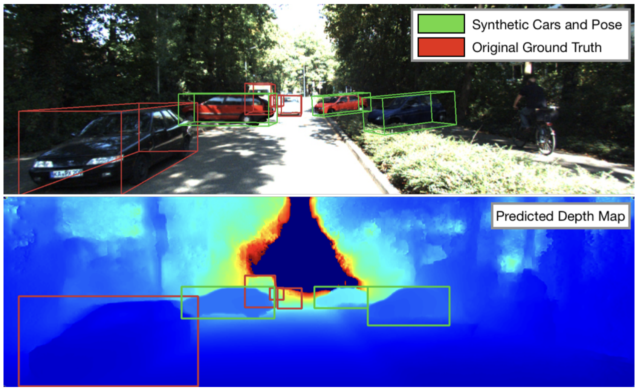


Figure 6. Synthetically generated training sample. Top: Green bounding boxes show original ground truth cars and poses. In contrast, red boxes illustrate the rendered meshes from a sampled 6D pose. Bottom: Augmented depth map from SuperDepth [33]. Notice that we utilized the annotated meshes, which we colored using the ground truth pose and our projective texturing.

4. Evaluation

In this section, we describe our evaluation protocol, compare to the state of the art for RGB-based approaches, and provide an ablative analysis discussing the merits of our individual contributions.

4.1. Evaluation Protocol

We use the standard KITTI3D benchmark [12] and its official evaluation metrics. We evaluate our method on three different difficulties: easy, moderate, hard. Furthermore, as suggested we also set the IoU threshold to 0.7 for both 2D and 3D. For the pose, we compute the average precision (AP) in the Bird's eye view, which measures the overlap of the 3D bounding boxes projected on the ground plane. We also compute the AP for the full 3D bounding box.

4.2. Comparison to Related Work

We compare ourselves on the train/validation split from [3] and on the official test set against state-of-the-art RGB-based methods on KITTI3D, namely (stereo-based) 3DOP [4], Mono3D [3], and Xu *et al.* [41] which also uses a depth module for better reasoning. Note that, although slightly lower in 2D AP, our model using synthetic data provides the best pose accuracy among our trained networks and we chose this model to compete against the others. As can be seen in Table 1 and 2, our method performs worse in 2D

Method	Type	Bird Eye View AP [val / test]			3D Detection AP [val / test]		
		Easy	Moderate	Hard	Easy	Moderate	Hard
Mono3D [3]	Mono	5.22 / –	5.19 / –	4.13 / –	2.53 / –	2.31 / –	2.31 / –
3DOP [4]	Stereo	12.63 / –	9.49 / –	7.59 / –	6.55 / –	5.07 / –	4.10 / –
Xu *et al.* [41]	Mono	**22.03** / 13.73	**13.63** / 9.62	**11.60** / 8.22	**10.53** / 7.08	5.69 / 5.18	5.39 / 4.68
ROI-10D	Mono	10.74 / –	7.46 / –	7.06 / –	7.79 / –	5.16 / –	3.95 / –
ROI-10D (Syn.)	Mono	14.50 / **16.77**	9.91 / **12.40**	8.73 / **11.39**	9.61 / **12.30**	**6.63 / 10.30**	**6.29 / 9.39**

Table 1. 3D detection performance on KITTI3D validation [3] and official KITTI3D test set. We report our AP for Bird's eye view and 3D IoU at the official IoU threshold of 0.7 for each metric. Note that we only evaluated the synthetic ROI-10D version on the online test set.

Method	2D Detection AP [val /test]		
	Easy	Moderate	Hard
Mono3D [3]	**93.89** / 92.33	**88.67 / 88.66**	**79.68** / 78.96
3DOP [4]	93.08 / **93.04**	88.07 / 88.64	79.39 / **79.10**
Xu *et al.* [41]	– / 90.43	– / 87.33	– / 76.78
ROI-10D	89.04 / –	88.39 / –	78.77 / –
ROI-10D (Syn.)	85.32 / 75.33	77.32 / 69.64	69.70 /61.18

Table 2. 2D AP performance on KITTI3D validation [3] and official test set at official IoU threshold of 0.7.

due to our strict 3D-NMS, but we are by far the strongest in the Bird's Eye View and the 3D AP. This underlines the important aspect of of proper data analysis to counteract overfitting. On the official test set, we get around twice the 3D AP of our closest monocular competitor. It is noteworthy that [41] trained their depth module on both KITTI3D and Cityscapes [6] for better generalization whereas the SuperDepth model we use has been pre-trained on KITTI data only. Interestingly, they have a strong drop in numbers when moving from the validation set onto the test set (e.g. from 22.03% to 13.73% or 10.53% to 7.08%), which suggests aggressive tuning towards the validation set with known ground truth. We want to mention that the evaluation protocol forces the 3D AP and Bird's eye view AP to be bounded from above by the 2D detection AP since missed detections in 2D always reflect negatively on the pose metrics. This strengthens our case further since our pose metric numbers would be even higher if we were to correct them with a 2D AP normalization.

4.3. Ablative Analysis

In the ablative analysis we want to first investigate how our new loss specifically minimizes the alignment problem. Additionally, we will identify where and why certain poses in KITTI3D are so much more difficult to estimate right. Finally, we analyze our method in respect to different inputs and how well our loss affects the quality of the poses.

Lifting Loss We run a controlled experiment where, isolating one instance with ground truth RoI $\mathcal{X}$ and 3D box B^*, we solely optimize the lifting module $\mathcal{F}$ with randomly initialized parameters. The step-wise improvement in alignment between $\mathcal{F}(\mathcal{X})$ and B^* is depicted in Figure 7 and we refer to the supplementary material for the full animations. Independent of initialization, we can observe that our loss always converges smoothly towards the global optimum. We also show the magnitude of each Jacobian component from Eq. 3 and can see that the loss focuses strongly on depth while steadily increasing importance towards rotation and 2D centroid position. Since our scale regression recovers deviation from the average car size, it was mostly neglected during optimization since the original error in extents was minimal. Without manually enforcing any principal direction during optimization or scaling the magnitudes, the loss steers the impact of each component quite well.

Pose Recall vs. Training Data To better understand our strengths and weaknesses, in Fig. 8 we show our recall for different bins for depth and rotation on the train/val split from [3]. We accept a detection if the Bird's Eye View IoU is larger than 0.5. Note that we followed the KITTI convention, such that an angle of 0 degrees corresponds to an object facing to the right. Since the dataset is rather small for deep learning methods, we also plot the training data distribution to understand if there is a correlation between sample frequency and pose quality.

For translation we did not discover any connection between the number of occurrences in the training data and the pose results. Nevertheless, closer objects are in general significantly better localized in 3D than objects further away. This can be explained by the fact that the network strongly relies on the predicted depth map to estimate the distance. However, the uncertainty of our monocular depth estimation also grows with distance. Very interestingly, utilizing our synthetic data generation improves the results across all bins. This confirms that, since the variety of scenes is limited, the network learns biases quickly and risks over-fitting without our proposed augmentation.

Our synthetic approach also clearly leads to better rotation estimates. In contrast to translation, we can find a strong correlation between the training data distribution and pose quality. While our method achieves good results on frequent viewpoints, the recall naturally drops when objects are seen from an underrepresented angle.

Method	2D Detection AP [0.7]			Bird's Eye View AP [0.5 / 0.7]			3D Detection AP [0.5 / 0.7]		
	Easy	Moderate	Hard	Easy	Moderate	Hard	Easy	Moderate	Hard
No Weighting	88.95	87.54	78.68	40.17 / 11.85	27.85 / 7.32	24.49 / 7.22	33.95 / 7.47	22.53 / 4.83	21.78 / 3.76
Multi-Task Weighting [21]	88.20	83.81	74.87	36.22 / 10.00	26.82 / 6.60	23.02 / 5.84	31.40 / 6.70	21.04 / 4.64	17.32 / 3.63
ROI-10D (w/o depth)	78.57	73.44	63.69	36.21 / 14.04	24.90 / 3.69	21.03 / 3.56	29.38/ **10.12**	19.80 / 1.76	18.04 / 1.30
ROI-10D	**89.04**	**88.39**	**78.77**	42.65 / 10.74	29.80 / 7.46	25.03 / 7.06	36.25 / 7.79	23.00 / 5.16	**22.06** / 3.95
ROI-10D (Syn.)	85.32	77.32	69.70	**46.85 / 14.50**	**34.05 / 9.91**	**30.46 / 8.73**	**37.59** / 9.61	**25.14 / 6.63**	21.83 / **6.29**

Table 3. Different weighting strategies and input modalities on the train/validation split from [3]. We report our AP referring to 2D detection, the bird's eye view challenge and 3D IoU. Besides the official IoU threshold of 0.7, we also report for a softer threshold of 0.5.

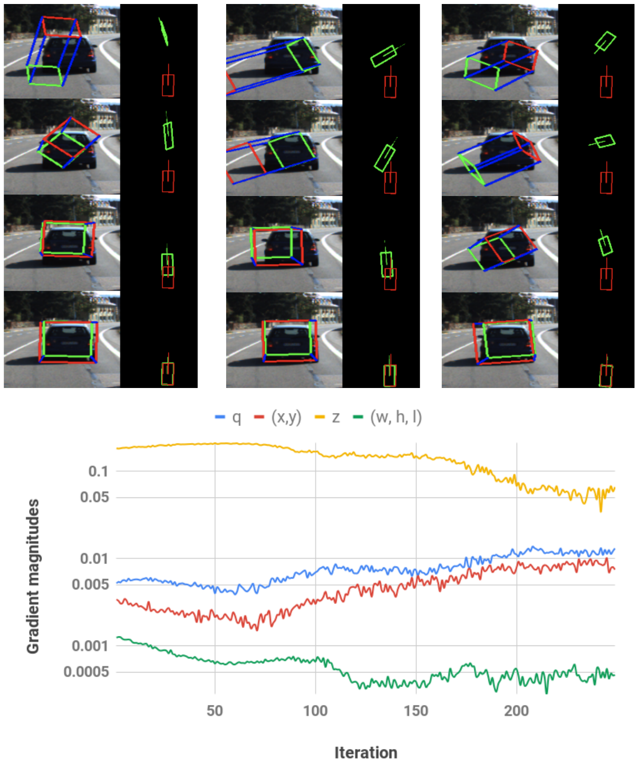


Figure 7. Controlled lifting loss experiment with given 2D RoI $\mathcal{X}$ over multiple runs with different seeding. Top: Visualizing $\mathcal{F}(\mathcal{X})$ during optimization in camera and bird's eye view. Bottom: Gradient magnitudes of each lifting component, averaged over all runs. We refer to the supplement for the full animations.

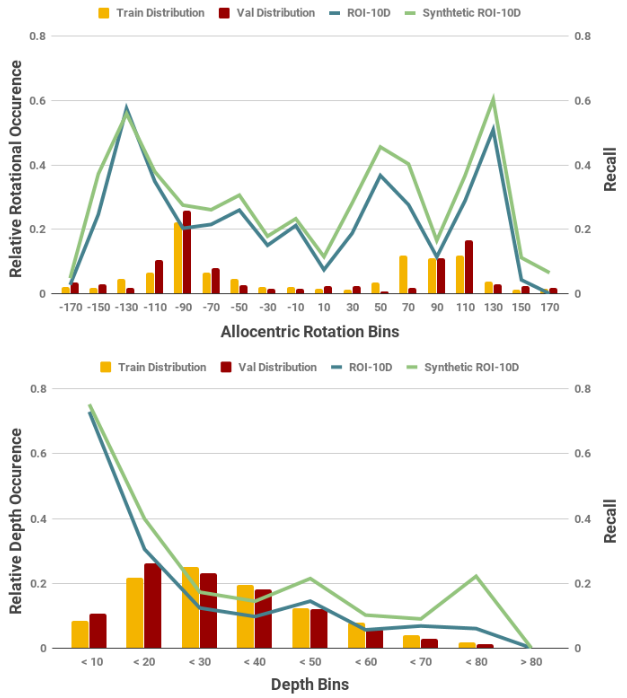


Figure 8. Recall of orientation and depth against the ground truth split distributions. Evidently, there exists a strong correlation between model performance and sample distribution. Synthetically augmenting underrepresented bins leads to overall better results.

Loss and input data We trained networks with different loss and data configurations on the train/validation [3] split to incrementally highlight our contributions. In the first two rows of Table 3 we ran training with separate regression terms instead of our lifting loss. While the first row shows the results with uniform weighting of all terms of $\mathcal{F}$ (similar to the approach of Xu *et al.* [41]), the second row shows training with the adaptive multi-task weighting from Kendall *et al.* [21]. Interestingly, we were not able to see an improvement with the adaptive weighting. We believe it comes from the fact that each term's magnitude is not at all comparable: while the (x, y) centroid moves in RoI-normalized image coordinates, the depth z is metric, the extents (w, h, l) are multiples of standard deviation from the mean extent, and the rotation q moves on a 4D unit sphere. Any uninformed weighting about the actual 3D instantiation has no means to properly assess the relative importance apart from numerical magnitude, thus comparing apples to oranges. Our formulation (row 4) avoids these problems and is either equal or better across all metrics.

Table 3 also presents results of a trained variant without monocular depth (row 3) and results for our method using depth without (rows 4) and with (row 5) synthetic augmentation. The results without depth cues are clearly worse, but we nonetheless get respectable numbers for the Bird's eye view and 3D AP. Unfortunately, our aggressive 3D-NMS discarded some correct solutions because of wrongly-regressed overlapping z-values, reducing our 2D AP significantly. Our synthetic data training shows strong im-

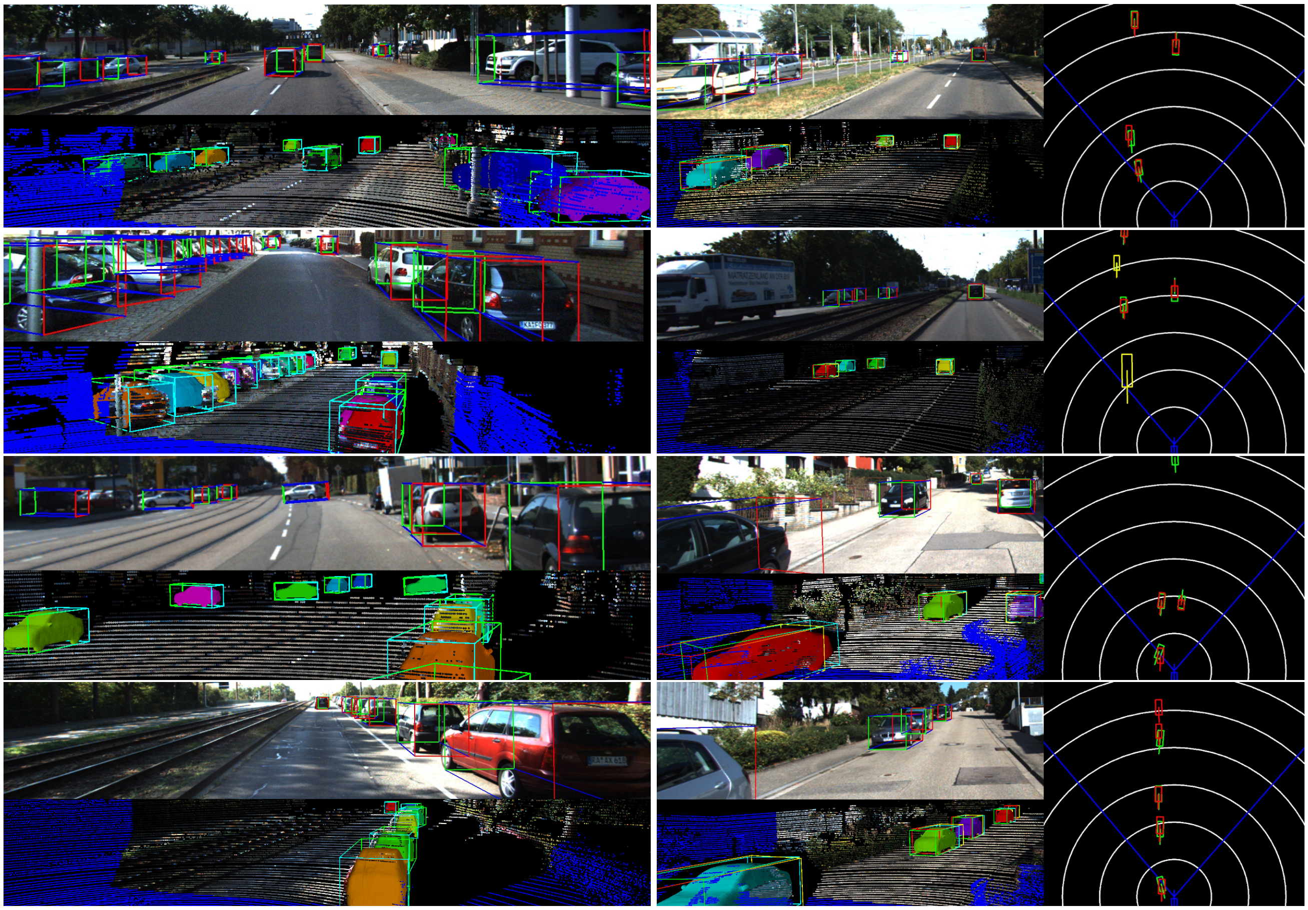


Figure 9. Qualitative results on the test (left) and validation (right) set. Noteworthy, we only trained on the train split to ensure that we never saw any of these images. For the validation samples, we additionally depict the ground truth poses in red. To get a proper estimate of the accuracy of the poses, we also plot the Bird's eye view (right) where we show clearly that we can recover accurate poses and proper metric shapes for unseen data, even at a distance.

provement on the pose metrics since we reduced the rotational data sample imbalance. By inspecting the drop in 2D AP, we realized that we designed our augmentations to be occlusion-free to avoid unrealistic intersections with the environment. In turn, this led to a weaker representation of strongly-occluded instances and to another introduced bias. We also show some qualitative results in Figure 9.

5. Conclusion

We proposed a monocular deep network that can lift 2D detections in 3D for metrically accurate pose estimation and shape recovery, optimizing directly a novel 3D loss formulation. We found that maximizing 3D alignment end-to-end for 6D pose estimation leads to very good results since we optimize for exactly the quantity we seek. We provided some insightful analysis on pose distributions in KITTI3D and how to leverage this information with recovered meshes for synthetic data augmentation. We found this reflection to be very helpful and quite important to improve the pose recall. Non-maximum-suppression in 2D and 3D is, however, a major influence on the final results and should to be addressed in future work, too.

References

[1] Hassan Abu Alhaija, Siva Karthik Mustikovela, Lars Mescheder, Andreas Geiger, and Carsten Rother. Augmented reality meets computer vision: Efficient data generation for urban driving scenes. *International Journal of Computer Vision*, 126(9):961–972, 2018.

[2] Konstantinos Bousmalis, Nathan Silberman, David Dohan, Dumitru Erhan, and Dilip Krishnan. Unsupervised pixel-level domain adaptation with generative adversarial networks. *CVPR*, pages 95–104, 2017.

[3] Xiaozhi Chen, Kaustav Kundu, Ziyu Zhang, Huimin Ma, Sanja Fidler, and Raquel Urtasun. Monocular 3d object detection for autonomous driving. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2016.

[4] Xiaozhi Chen, Kaustav Kundu, Yukun Zhu, Andrew Berneshawi, Huimin Ma, Sanja Fidler, and Raquel Urtasun. 3d object proposals for accurate object class detection. In *Proceedings of the 28th International Conference on Neural Information Processing Systems - Volume 1*, NIPS, pages 424–432, Cambridge, MA, USA, 2015. MIT Press.

[5] Xiaozhi Chen, Huimin Ma, Ji Wan, Bo Li, and Tian Xia. Multi-view 3d object detection network for autonomous driving. In *IEEE CVPR*, 2017.

[6] Marius Cordts, Mohamed Omran, Sebastian Ramos, Timo Rehfeld, Markus Enzweiler, Rodrigo Benenson, Uwe Franke, Stefan Roth, and Bernt Schiele. The cityscapes dataset for semantic urban scene understanding. In *Proc. of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.

[7] Zhuo Deng and Longin Jan Latecki. Amodal detection of 3d objects: Inferring 3d bounding boxes from 2d ones in rgb-depth images. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017.

[8] Alexey Dosovitskiy, German Ros, Felipe Codevilla, Antonio Lopez, and Vladlen Koltun. CARLA: An open urban driving simulator. In *Proceedings of the 1st Annual Conference on Robot Learning*, pages 1–16, 2017.

[9] David Eigen, Christian Puhrsch, and Rob Fergus. Depth map prediction from a single image using a multi-scale deep network. In *Advances in neural information processing systems*, pages 2366–2374, 2014.

[10] A Gaidon, Q Wang, Y Cabon, and E Vig. Virtual worlds as proxy for multi-object tracking analysis. In *CVPR*, 2016.

[11] Ravi Garg, Vijay Kumar BG, Gustavo Carneiro, and Ian Reid. Unsupervised cnn for single view depth estimation: Geometry to the rescue. In *European Conference on Computer Vision*, pages 740–756. Springer, 2016.

[12] Andreas Geiger, Philip Lenz, and Raquel Urtasun. Are we ready for autonomous driving? the kitti vision benchmark suite. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2012.

[13] Clément Godard, Oisin Mac Aodha, and Gabriel J Brostow. Unsupervised monocular depth estimation with left-right consistency. In *CVPR*, volume 2, page 7, 2017.

[14] Alexander Grabner, Peter M. Roth, and Vincent Lepetit. 3D Pose Estimation and 3D Model Retrieval for Objects in the Wild. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018.

[15] Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross Girshick. Mask R-CNN. In *Proceedings of the International-Conference on Computer Vision (ICCV)*, 2017.

[16] 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*, 2012.

[17] Stefan Hinterstoisser, Vincent Lepetit, Paul Wohlhart, and Kurt Konolige. On pre-trained image features and synthetic images for deep learning. *CoRR*, abs/1710.10710, 2017.

[18] Angjoo Kanazawa, Shubham Tulsiani, Alexei A. Efros, and Jitendra Malik. Learning category-specific mesh reconstruction from image collections. In *ECCV*, 2018.

[19] Hiroharu Kato, Yoshitaka Ushiku, and Tatsuya Harada. Neural 3d mesh renderer. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3907–3916, 2018.

[20] Wadim Kehl, Fabian Manhardt, Federico Tombari, Slobodan Ilic, and Nassir Navab. Ssd-6d: Making rgb-based 3d detection and 6d pose estimation great again. In *The IEEE International Conference on Computer Vision (ICCV)*, Oct 2017.

[21] Alex Kendall, Yarin Gal, and Roberto Cipolla. Multi-task learning using uncertainty to weigh losses for scene geometry and semantics. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018.

[22] Jason Ku, Melissa Mozifian, Jungwook Lee, Ali Harakeh, and Steven Waslander. Joint 3d proposal generation and object detection from view aggregation. *IROS*, 2018.

[23] Abhijit Kundu, Yin Li, and James M. Rehg. 3d-rcnn: Instance-level 3d object reconstruction via render-and-compare. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2018.

[24] Yi Li, Gu Wang, Xiangyang Ji, Yu Xiang, and Dieter Fox. Deepim: Deep iterative matching for 6d pose estimation. In *The European Conference on Computer Vision (ECCV)*, September 2018.

[25] Tsung-Yi Lin, Piotr Dollár, Ross B Girshick, Kaiming He, Bharath Hariharan, and Serge J Belongie. Feature pyramid networks for object detection. In *CVPR*, volume 1, page 4, 2017.

[26] Tsung-Yi Lin, Priyal Goyal, Ross Girshick, Kaiming He, and Piotr Dollár. Focal loss for dense object detection. *IEEE transactions on pattern analysis and machine intelligence*, 2018.

[27] Rosanne Liu, Joel Lehman, Piero Molino, Felipe Petroski Such, Eric Frank, Alex Sergeev, and Jason Yosinski. An intriguing failing of convolutional neural networks and the coordconv solution. *CoRR*, abs/1807.03247, 2018.

[28] William E. Lorensen and Harvey E. Cline. Marching cubes: A high resolution 3d surface construction algorithm. In *Proceedings of the 14th Annual Conference on Computer Graphics and Interactive Techniques*, SIGGRAPH '87, pages 163–169, New York, NY, USA, 1987. ACM.

[29] Reza Mahjourian, Martin Wicke, and Anelia Angelova. Unsupervised learning of depth and egomotion from monocular video using 3d geometric constraints. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2018.

[30] Fabian Manhardt, Wadim Kehl, Nassir Navab, and Federico Tombari. Deep model-based 6d pose refinement in rgb. In *The European Conference on Computer Vision (ECCV)*, September 2018.

[31] Arsalan Mousavian, Dragomir Anguelov, John Flynn, and Jana Kosecka. 3d bounding box estimation using deep learning and geometry. *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 5632–5640, 2017.

[32] Adam Paszke, Sam Gross, Soumith Chintala, Gregory Chanan, Edward Yang, Zachary DeVito, Zeming Lin, Al-

ban Desmaison, Luca Antiga, and Adam Lerer. Automatic differentiation in pytorch. In *NIPS-W*, 2017.

[33] Sudeep Pillai, Rares Ambrus, and Adrien Gaidon. Superdepth: Self-supervised, super-resolved monocular depth estimation, 2018.

[34] Mahdi Rad and Vincent Lepetit. Bb8: A scalable, accurate, robust to partial occlusion method for predicting the 3d poses of challenging objects without using depth. In *The IEEE International Conference on Computer Vision (ICCV)*, Oct 2017.

[35] Mahdi Rad, Markus Oberweger, and Vincent Lepetit. Feature mapping for learning fast and accurate 3d pose inference from synthetic images. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018.

[36] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster R-CNN: Towards real-time object detection with region proposal networks. In *Neural Information Processing Systems (NIPS)*, 2015.

[37] Bugra Tekin, Sudipta N. Sinha, and Pascal Fua. Real-time seamless single shot 6d object pose prediction. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2018.

[38] Endre Weiszfeld. Sur le point pour lequel la somme des distances de n points donnés est minimum. *Tohoku Mathematical Journal, First Series*, 43:355–386, 1937.

[39] Yuxin Wu and Kaiming He. Group normalization. In *CVPR*, 2018.

[40] Yu Xiang, Tanner Schmidt, Venkatraman Narayanan, and Dieter Fox. Posecnn: A convolutional neural network for 6d object pose estimation in cluttered scenes. *Robotics: Science and Systems (RSS)*, 2018.

[41] Bin Xu and Zhenzhong Chen. Multi-level fusion based 3d object detection from monocular images. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2018.

[42] Tinghui Zhou, Matthew Brown, Noah Snavely, and David G Lowe. Unsupervised learning of depth and ego-motion from video. In *CVPR*, volume 2, page 7, 2017.

Part III

Conclusion and Outlook

Summary and Findings 6

This dissertation introduced the problem of 6D pose estimation from monocular data and its accompanying challenges. Solutions grounded on deep learning are further proposed to tackle them fast and reliably. We essentially leverage recent advents in 2D object detection and tailor known backbones to the task of pose estimation. Our new single-stage formulation is one of the first ever deep learning based method, surpassing all related works in run-time and accuracy. Moreover, in contrast to almost all existing methods at the time, our method is capable of processing the **full** 6D pose space as well as multiple objects at more than 10 frames per second. We further show that we can almost fully close the gap towards methods harnessing depth data via learning the well known ICP paradigm in an unambiguous fashion. Leveraging well-established ideas from edge alignment, we enforce a novel visual proxy loss to align the object in 6D. As the approach is fully data-driven, our 6D refinement is not dependant on precise hyper-tuning and is considerably less prone to fall for unpleasant local minima.

Nevertheless, several limitations and challenges can negatively impact the final pose outcomes. Some of the most prominent challenges involve different illumination settings, ambiguities in pose, and processing previously unseen objects. Each challenge can be, however, tackled employing appropriate design choices when modeling the problem. First, we addressed the problem of lighting variation by means image normalization employing Generative Adversarial Networks (GANs). When encouraging the GAN to generate images which are tailored to the associated down-stream task (*e.g.* 2D or 6D pose estimation), the network is forced to implicitly transfer the object from the input domain to the *"light-free"* domain. Our data-driven approach is invariant to excessive light variations, and can also handle non-uniform lighting. Finally, opposed to most related works, neither a particular large dataset is required, nor is prior knowledge of the input data assumed. Second, we deal with ambiguities in pose harnessing multiple hypotheses. Our novel formulation does neither require any input pre-processing nor particular annotations for ambiguities. Moreover, analyzing these multiple hypotheses enables dealing adequately with ambiguous situations as well as understanding the particular kind of ambiguity. In addition, the predicted hypotheses can be also leveraged as a metric of reliability for the 6D pose. Finally, to handle previously unseen objects, we simultaneously estimate metric poses and shapes for each object of a given class. Thereby, we propose to measure the metric misalignment in 3D as objective function. Our evaluation has shown that directly optimizing for the final goal leads to superior results as it avoids explicit weighting of different loss terms. In addition, we demonstrate that synthesizing additional training data from 3D reconstructions of real cars can help to prevent overfitting.

While our works tackle problems of high importance within the field of 6D pose estimation, computing the precise pose using only RGB data is still far from being solved. As for robotics manipulation, only for very simple scenes and limited objects, poses estimated by monocular methods can be sufficiently accurate to enable grasping. A main problem simply resides in the lack of respective data. Thereby, improving the generation of synthetic data and finding new and better ways to self-supervise pose estimation are two of the most important key problems to solve. This is especially true when it comes to class-level full 9D pose estimation as annotating data becomes even more difficult. The next chapter emphasizes a little more these open tasks and also show some explorations towards these problems.

Future Work and Discussion

7

While the interest in category-level 9D pose estimation just recently started to increase, there is still only little work devoted to this topic. This can be contributed to the fact that this task is significantly more complicated than computing the 6D or 7D pose. In particular, 9D pose estimation requires to estimate all 3 degrees-of-freedom for 3D rotation and, most importantly, there is almost no real data available for this task. Moreover, training on synthetic data is rather complicated due to the scale-distance ambiguity. Similar to monocular depth estimation [43, 44], it is indeed possible to exploit geometric priors to resolve the scale-distance ambiguity, however, this requires the possession of synthetic data which follows real physical constraints. A few methods have been proposed to tackle this problem, however, all of them assume depth data as additional input modality [116, 184, 185]. Whereas this allows to address the scale-distance ambiguity, these methods depend on the possession of an expensive depth sensor.

To summarize, since collecting real data for each object of interest is intractable and only handling a few objects significantly limits applications, we believe there a three major directions to investigate. Essentially, the direction of self-supervised learning for 6D and 9D pose estimation should be further explored as this allows training on real data without requiring any annotations [4]. Further, as self-supervision is highly dependant on good initialization, generating high quality and physically correct synthetic images is another important aspect [201]. Finally, current object pose estimators should be extended to also work with unseen objects, even from monocular data [6].

7.1 Improving the Quality of Synthetic Training Data

In terms of synthetic data for 6D pose estimation, a few works for physically correct and high quality renderings using ray-tracing have been recently proposed [201, 202]. Thereby, physically plausible constellation are achieved by dropping CAD models into 3D scenes by means of physics-engines. Leveraging ray-tracing highly realistic images can be generated as appropriate sampling of the rays allows to take probabilistic illumination models into account. These high quality images led to impressive results with respect to the BOP challenge, as the accuracy of most methods more than doubled when using PBR renderings rather than OpenGL renderings [8]. Unfortunately, the utilized data still exhibits several weaknesses. In the core, the whole scene needs to be rendered which is slow and leads to overly clear images, fairly different from real data. As for the BOP challenge, the objects were dropped into cubes with HD textures. While this enabled higher rendering speed, the scenes were rather simple in terms of geometry.

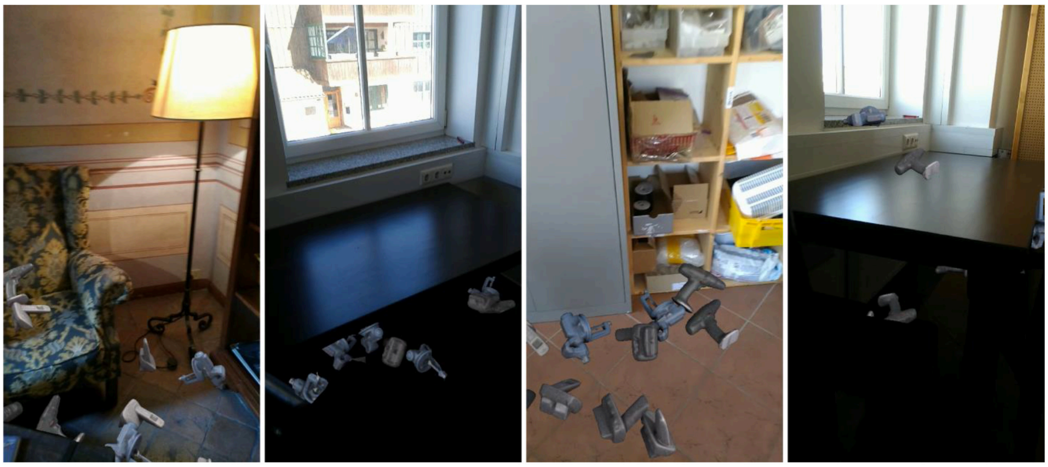


Figure 7.1. **Physically Plausible Renderings in Real Worlds.** To obtain physically plausible 6D poses, we drop 3D CAD models from the objects of interest into reconstructions of real scenes, using a physics engine. We then render the objects on top of images taken during the reconstruction of the scene.

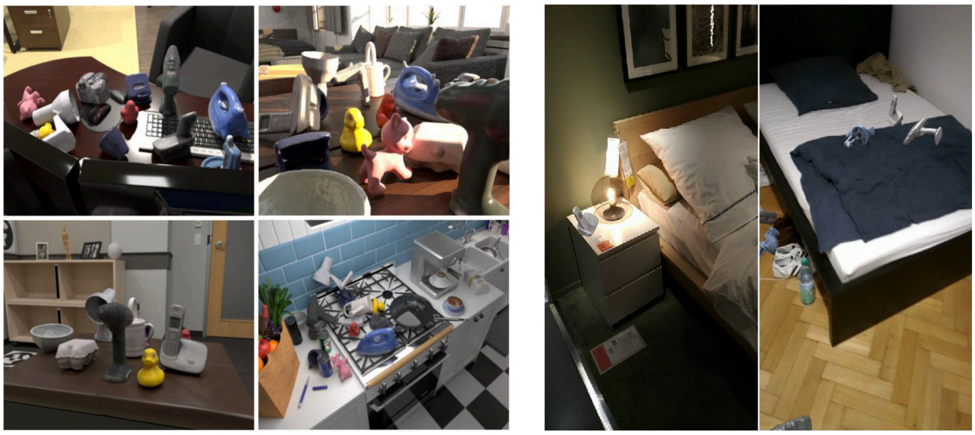


Figure 7.2. **Comparison With [201].** While [201] renders the whole synthetic environment which is slow and unrealistic [left] [©2019 IEEE], we instead leverage a mixed-reality approach to generate data which is as realistic as possible [right]. For example, notice how the phone is placed correctly on the nightstand next to the bed.

To circumvent these issues, we recently explored the idea of fast physically plausible rendering with as little rendering involved as possible. Essentially, we render objects physically correct on top of real images, yet, consider light and reflections from the scene. Hence, since we only ray-trace the objects, the algorithm can produce almost completely real data at high speed. In practice, we drop objects into 3D scans from the large-scale 3RScan dataset [203] using the NVidia PhysX engine (similar to [201]) and render all objects with AppleSeed from the camera poses of the accompanying RGB-D images. The rendered RGB-D image is then blended with the real RGB-D image. We further employ ambient occlusion for realistic contact shadows and self-shadowing. Notice that if the ray hits an object with reflective surface, the ray is further traced against the scene mesh to incorporate realistic reflections. As 3RScan possess 478 separate scenes, pose estimation is less prone to overfitting compared to [201] having only 7 scenes. Exemplary renderings and a small qualitative comparison with [201] are respectively shown in Figure 7.1 and Figure 7.2.

7.2 Class-level Monocular 9D Object Pose Estimation and the Effect of Self-supervised Learning

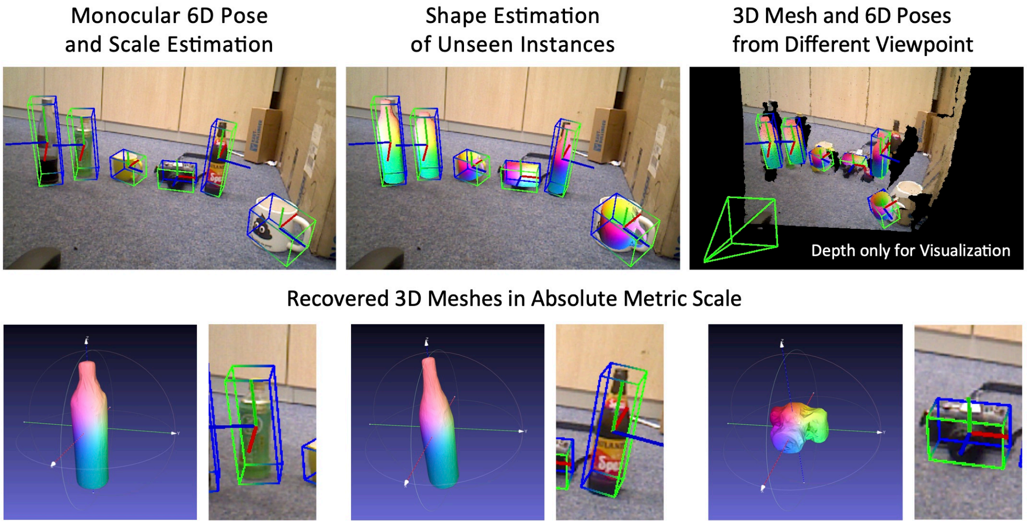

Monocular 6D Pose and Scale Estimation
Shape Estimation of Unseen Instances
3D Mesh and 6D Poses from Different Viewpoint
Depth only for Visualization
Recovered 3D Meshes in Absolute Metric Scale

Figure 7.3. **Estimating Metric Pose and Shape From Monocular Imagery.** Top left: We feed our detector with a monocular image to estimate the 9D pose and geometric properties of unseen objects from a particular class. In particular, we show each object's inferred shape on the bottom and additionally rendered them into the associated scene on the top center. Top Right: We also demonstrate our estimated results from a different viewpoint in an effort to constitute that our method is able to infer all parameters in correct metric scale.

In regard of monocular 9D pose estimation, combining ideas from the aforementioned work from Section 5.3.1 [6] with our work on self-supervised 6D pose estimation, which removes the need for real pose labels [4], we introduce the task of class-level monocular 6D pose paired with metric shape estimation. Leveraging synthetic data, we propose a novel fully differentiable end-to-end pipeline for 6D object pose and shape estimation, which directly aligns the predicted mesh with the scene in 3D. Due to aforementioned limitations, *i.e.* the scale-distance ambiguity and the lack of labeled real data, the initial output poses are not yet very reliable as depth estimation is noisy. Therefore, we adjust our self-supervision from Self6D [4] to bring it to the domain of class-level 9D pose estimation, in an effort to allow training on real data and thus strengthen the predictions by decreasing the domain gap.

Similar to Section 5.3.1 [6], we first learn a low-dimensional shape space. Instead of using TSDFs, we rely on AtlasNet [62] as differentiable rendering for self-supervision requires the possession of 3D meshes. AtlasNet is an AutoEncoder based on PointNet [101], which takes a pointcloud $V \in \mathbb{R}^{3\times N}$ as input and produces a global shape descriptor e. In the following the shape can be inferred feeding the Decoder D_{atlas} with e and 2D locations sampled from a 2D uv-map. Since D_{atlas} is a continuous mapping from 2D to 3D, the triangles $E_{triangles}$ can be simply derived from the sampling. As before, we employ RetinaNet with Focal loss [75] and append different heads for the regression of the allocentric 3D rotation as 4D quaternion q_a, the 3D translation $t = K^{-1}z\,(x, y, 1)^T$ as well as the shape encoding e, the scale (w, h, l),

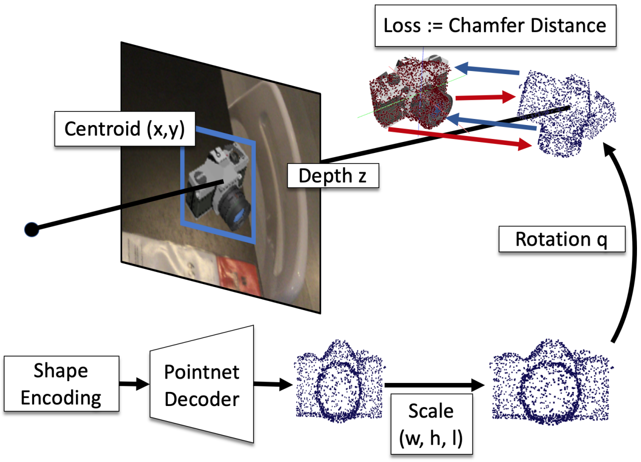


Figure 7.4. **3D Pointcloud Alignment.** We first infer the detected object's shape using AtlasNet. We then scale it to metric size and transform it to camera space by means of the estimated rotation and translation. Finally, we leverage the chamfer distance to seek for an optimal alignment in 3D.

and the visible object mask M^P. We then leverage the individual predictions to compute the associated point cloud and transform it to the 3D camera space

$$\hat{V}_{3D} := q \cdot \left[\begin{pmatrix} w \\ h \\ l \end{pmatrix} \cdot \mathcal{D}_{atlas}(e) \right] \cdot q^{-1} + K^{-1} \begin{pmatrix} x \cdot z \\ y \cdot z \\ z \end{pmatrix}, \tag{7.1}$$

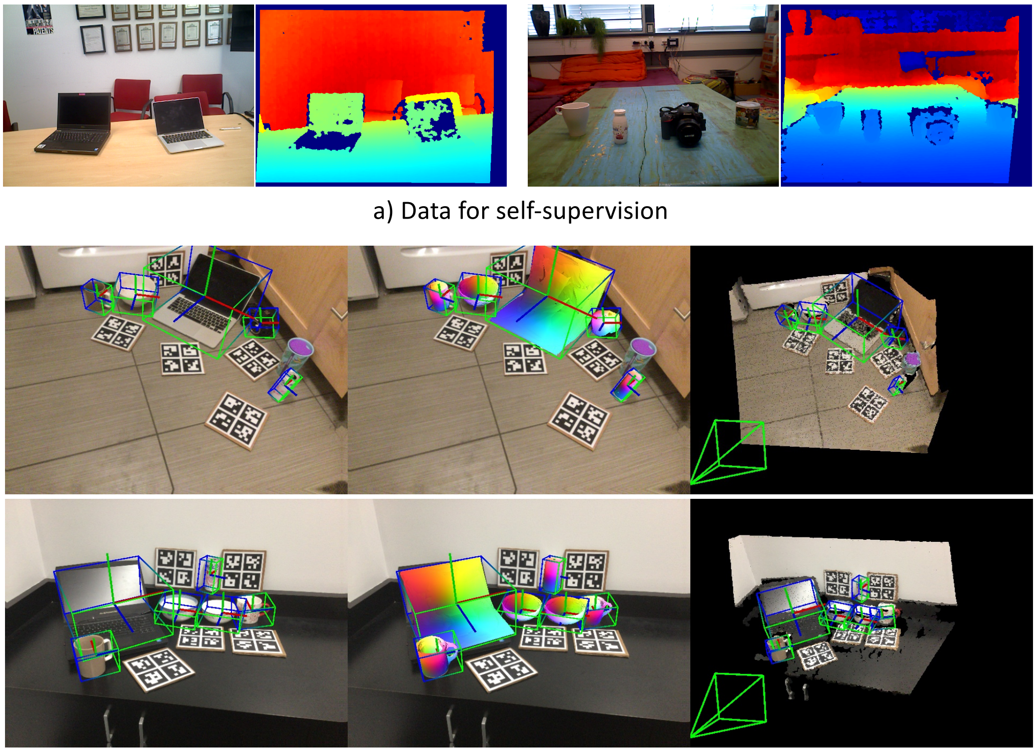
with K being again the camera intrinsic matrix and q denoting again the egocentric rotation after conversion. We then fully supervise the model with the groundtruth pointcloud $\bar{V}_{3D}$ using the chamfer distance according to

$$\mathcal{L}_{3D} := \frac{1}{|\hat{V}_{3D}|} \sum_{\hat{v} \in \hat{V}_{3D}} \min_{\bar{v} \in \bar{V}_{3D}} ||\hat{v} - \bar{v}||_2 + \frac{1}{|\bar{V}_{3D}|} \sum_{\bar{v} \in \bar{V}_{3D}} \min_{\hat{v} \in \hat{V}_{3D}} ||\hat{v} - \bar{v}||_2. \tag{7.2}$$

An illustration of our proposed pointcloud loss can be found in Figure 7.4.

While the network is capable of predicting the 6D pose and 3D mesh of each object, the quality is still not very satisfactory. Therefore, we recorded over 30k unlabeled RGB-D images to enable self-supervision on this task. We made the recorded data publicly available at https://forms.gle/E89Asu3YDkL1WJEj6. Two exemplary samples from the dataset can be found in Figure 7.5 a). To conduct self-supervision we use our modified differentiable renderer $\mathcal{R}$ from [4] to render the visible object mask M^R and depth map D^R with $(M^R, D^R) = \mathcal{R}(q, t, \mathcal{M})$ and $\mathcal{M} = (D_{atlas}(e), E_{triangles})$. As our meshes are not colorized, in contrast to Self6D, we only use the mask loss $\mathcal{L}_{mask}$ for visual supervision

$$\mathcal{L}_{mask} := -\frac{1}{|N_+|} \sum_{j \in N_+} M_j^P \log M_j^R - \frac{1}{|N_-|} \sum_{j \in N_-} \log(1 - M_j^R), \tag{7.3}$$



a) Data for self-supervision

b) Results on the real NOCS dataset

Figure 7.5. **Recorded data for self-supervision and qualitative results on NOCS.** a) Two recorded RGB-D samples, utilized during self-supervision. b) Qualitative results on the real NOCS test split. From left to right: Predicted 3D bounding boxes rendered on top of the input image, estimated 3D shapes rendered on top of the input image using the predicted 6D pose, predicted 3D bounding boxes from a different view to demonstrate that the method can cope with the scale-distance ambiguity.

with N_+ and N_- denoting foreground and background pixels, respectively. Remember that M^P refers to the predicted masks from our network after training on synthetic data. As predicting the visible 2D object mask is a fairly easy task that well translates to real data, we employ the predicted masks as a weak supervision signal. Our experiments in [4] have demonstrated that only the mask loss has considerable influence on the pose, whereas the other terms for visual alignment only have minor impact.

As 3D supervision we then again employ the chamfer distance between the backprojected visible points V^R and V^S from the rendered and raw depth map with

$$\pi^{-1}(D, M, K) = \{\, K^{-1} \begin{pmatrix} x_j \\ y_j \\ 1 \end{pmatrix} \cdot D_j \mid \forall j \in M > 0 \,\} \tag{7.4}$$

$$V^S := \pi^{-1}(D^S, M^P, K) \qquad V^R := \pi^{-1}(D^R, M^R, K). \tag{7.5}$$

Nonetheless, as our initial prediction are not as accurate as those from [4], we first align the visible centroids $c^R = \frac{1}{|V^R|}\sum_{v^R \in V^R} v^R$ and $c^S = \frac{1}{|V^S|}\sum_{v^S \in V^S} v^S$ according to $\delta_c = c^S - c^R$. The final loss for geometric alignment can then be compute as follows

$$\mathcal{L}_{geom} := \frac{1}{|V^S|}\sum_{v^S \in v^S} \min_{v^R \in V^R} \|v^S - v^R + \delta_c\|_2 + \frac{1}{|V^R|}\sum_{v^R \in V^R} \min_{v^S \in V^S} \|v^S - v^R + \delta_c\|_2 + ||\delta_c||_2. \quad (7.6)$$

We experimentally prove that our self-supervision $\mathcal{L} = \mathcal{L}_{mask} + \nu\mathcal{L}_{geom}$ significantly further enhances performance on real data with respect to the NOCS dataset [116] (*e.g.* AP of 14.0 without self-supervision *vs.* 17.7 with self-supervision for 3D IoU at a threshold of 0.5), even on par with methods using RGB-D data in the pure synthetic setting. Moreover, when leveraging ICP, we can also achieve similar results on the real test split as state-of-the-art RGB-D methods relying on real labeled data [116, 184]. Qualitative results are provided in Figure 7.5 b). For more information we kindly refer the reader to [3]

Part IV

Appendix

A Abstracts of Publications not Discussed in this Dissertation

BOP: Benchmark For 6D Object Pose Estimation

Tomas Hodan, Frank Michel, Eric Brachmann, Wadim Kehl, Anders Glent Buch, Dirk Kraft, Bertram Drost, Joel Vidal, Stephan Ihrke, Xenophon Zabulis, Caner Sahin, **Fabian Manhardt**, Federico Tombari, Tae-Kyun Kim, Jiri Matas, Carsten Rother

Proceedings of the European Conference on Computer Vision (ECCV), Munich 2018

Abstract. *We propose a benchmark for 6D pose estimation of a rigid object from a single RGB-D input image. The training data consists of a texture-mapped 3D object model or images of the object in known 6D poses. The benchmark comprises of: i) eight datasets in a unified format that cover different practical scenarios, including two new datasets focusing on varying lighting conditions, ii) an evaluation methodology with a pose-error function that deals with pose ambiguities, iii) a comprehensive evaluation of 15 diverse recent methods that captures the status quo of the field, and iv) an online evaluation system that is open for continuous submission of new results. The evaluation shows that methods based on point-pair features currently perform best, outperforming template matching methods, learning-based methods and methods based on 3D local features. The project website is available at bop.felk.cvut.cz.*

Self6D: Self-Supervised Monocular 6D Object Pose Estimation

Gu Wang*, **Fabian Manhardt***, Jianzhun Shao, Xiangyang Ji, Nassir Navab, Federico Tombari

Proceedings of the European Conference on Computer Vision (ECCV), Glasgow 2020

Abstract. *Estimating the 6D object pose is a fundamental problem in computer vision. Convolutional Neural Networks (CNNs) have recently proven to be capable of predicting reliable 6D pose estimates even from monocular images. Nonetheless, CNNs are identified as being extremely data-driven, yet, acquiring adequate annotations is oftentimes very time-consuming and labor intensive. To overcome this shortcoming, we propose the idea of monocular 6D pose estimation by means of self-supervised learning, which eradicates the need for real data with annotations. After training our proposed network fully supervised with synthetic RGB data, we leverage recent advances in neural rendering to further self-supervise the model on unannotated real RGB-D data, seeking for a visually and geometrically optimal alignment. Extensive evaluations demonstrate that our proposed self-supervision is able to significantly enhance the model's original performance, outperforming all other methods relying on synthetic data or employing elaborate techniques from the domain adaptation realm.*

GDR-Net: Geometry-Guided Direct Regression Network For Monocular 6D Object Pose Estimation

Gu Wang, **Fabian Manhardt**, Federico Tombari, Xiangyang Ji

IEEE Conference on Computer Vision and Pattern Recognition (CVPR). Nashville 2021

Abstract. *6D pose estimation from a single RGB image is a fundamental task in computer vision. The current top-performing deep learning-based methods rely on an indirect strategy, i.e. , first establishing 2D-3D correspondences between the coordinates in the image plane and object coordinate system, and then applying a variant of the PnP/RANSAC algorithm. However, this two-stage pipeline is not end-to-end trainable, thus is hard to be employed for many tasks requiring differentiable poses. On the other hand, methods based on direct regression are currently inferior to geometry-based methods. In this work, we perform an in-depth investigation on both direct and indirect methods, and propose a simple yet effective Geometry-guided Direct Regression Network (GDR-Net) to learn the 6D pose in an end-to-end manner from dense correspondence-based intermediate geometric representations. Extensive experiments show that our approach remarkably outperforms state-of-the-art methods on LM, LM-O and YCB-V datasets. Code is available at https://git.io/GDR-Net.*

Bibliography

[1] 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)*. 2021 (see pp. 1, 28, 36, 38).

[2] F. Manhardt, L. Minciullo, K. Yoshikawa, S. Meier, F. Tombari, and N. Kobori. “DB-GAN: Boosting Object Recognition Under Strong Lighting Conditions”. In: *2021 IEEE Winter Conference on Applications of Computer Vision (WACV)*. 2021 (see pp. 1, 19, 79).

[3] F. Manhardt, G. Wang, B. Busam, M. Nickel, S. Meier, L. Minciullo, X. Ji, and N. Navab. “CPS++: Improving Class-level 6D Pose and Shape EstimationFrom Monocular Images With Self-Supervised Learning”. In: *arXiv preprint arXiv:2003.05848v3* (2020) (see pp. 1, 27, 28, 134).

[4] G. Wang, F. Manhardt, J. Shao, X. Ji, N. Navab, and F. Tombari. “Self6D: Self-Supervised Monocular 6D Object Pose Estimation”. In: *ECCV*. Aug. 2020 (see pp. 1, 21, 28, 129, 131–134).

[5] 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 International Conference on Computer Vision*. 2019, pp. 6841–6850 (see pp. 1, 26–28, 33, 35, 36).

[6] F. Manhardt, W. Kehl, and A. Gaidon. “ROI-10D: Monocular lifting of 2d detection to 6d pose and metric shape”. In: *CVPR*. 2019, pp. 2069–2078 (see pp. 1, 5, 26–28, 32, 39–41, 129, 131).

[7] F. Manhardt, W. Kehl, N. Navab, and F. Tombari. “Deep model-based 6d pose refinement in rgb”. In: *ECCV*. 2018 (see pp. 2, 28, 33).

[8] 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 (see pp. 2, 30, 31, 35, 44, 79, 94, 129).

[9] F. Manhardt, W. Kehl, F. Tombari, S. Ilic, and N. Navab. “SSD-6D: Making RGB-Based 3D Detection and 6D Pose Estimation Great Again”. In: *The IEEE International Conference on Computer Vision (ICCV)*. Oct. 2017 (see pp. 2, 27, 30, 33, 35–38, 43, 57, 93, 94, 109).

[10] 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, pp. 3828–3836 (see pp. 5, 6, 29, 30, 36, 37, 44, 57).

[11] I. P. Howard, B. J. Rogers, et al. *Binocular vision and stereopsis*. Oxford University Press, USA, 1995 (see p. 5).

[12] A. J. McKnight, D. Shinar, and B Hilburn. “The visual and driving performance of monocular and binocular heavy-duty truck drivers”. In: *Accident Analysis & Prevention* 23.4 (1991), pp. 225–237 (see p. 5).

[13] J. Marotta, T. Perrot, D Nicolle, P Servos, and M. Goodale. “Adapting to monocular vision: grasping with one eye”. In: *Experimental Brain Research* 104.1 (1995), pp. 107–114 (see p. 5).

[14] D. J. Tan, F. Tombari, and N. Navab. “Real-time accurate 3D head tracking and pose estimation with consumer rgb-d cameras”. In: *International Journal of Computer Vision* 126.2-4 (2018), pp. 158–183 (see pp. 5, 6).

[15] X. Deng, Y. Xiang, A. Mousavian, C. Eppner, T. Bretl, and D. Fox. “Self-supervised 6D Object Pose Estimation for Robot Manipulation”. In: *ICRA*. 2020 (see pp. 5, 6).

[16] N. Correll, K. E. Bekris, D. Berenson, O. Brock, A. Causo, K. Hauser, K. Okada, A. Rodriguez, J. M. Romano, and P. R. Wurman. “Analysis and observations from the first amazon picking challenge”. In: *IEEE Transactions on Automation Science and Engineering* 15.1 (2016), pp. 172–188 (see p. 6).

[17] D. Morrison, P. Corke, and J. Leitner. “Closing the loop for robotic grasping: A real-time, generative grasp synthesis approach”. In: *arXiv preprint arXiv:1804.05172* (2018) (see p. 6).

[18] A. Mousavian, C. Eppner, and D. Fox. “6-dof graspnet: Variational grasp generation for object manipulation”. In: *ICCV*. 2019 (see p. 6).

[19] C. Wang, D. Xu, Y. Zhu, R. Martín-Martín, C. Lu, L. Fei-Fei, and S. Savarese. “DenseFusion: 6d object pose estimation by iterative dense fusion”. In: *CVPR*. 2019, pp. 3343–3352 (see pp. 6, 41).

[20] J. Wirtz, P. G. Patterson, W. H. Kunz, T. Gruber, V. N. Lu, S. Paluch, and A. Martins. “Brave new world: service robots in the frontline”. In: *Journal of Service Management* (2018) (see p. 6).

[21] M. Quigley, E. Berger, A. Y. Ng, et al. “Stair: Hardware and software architecture”. In: *AAAI 2007 Robotics Workshop, Vancouver, BC*. 2007, pp. 31–37 (see p. 6).

[22] M. Wise, M. Ferguson, D. King, E. Diehr, and D. Dymesich. “Fetch and freight: Standard platforms for service robot applications”. In: *Workshop on autonomous mobile service robots*. 2016 (see p. 6).

[23] T. Yamamoto, K. Terada, A. Ochiai, F. Saito, Y. Asahara, and K. Murase. “Development of Human Support Robot as the research platform of a domestic mobile manipulator”. In: *ROBOMECH journal* 6.1 (2019), p. 4 (see p. 6).

[24] B. Busam, M. Esposito, S. Che’Rose, N. Navab, and B. Frisch. “A stereo vision approach for cooperative robotic movement therapy”. In: *ICCVW*. 2015, pp. 127–135 (see p. 6).

[25] B. Busam, P. Ruhkamp, S. Virga, B. Lentes, J. Rackerseder, N. Navab, and C. Hennersperger. “Markerless inside-out tracking for interventional applications”. In: *arXiv preprint arXiv:1804.01708* (2018) (see p. 6).

[26] R. Elfring, M. de la Fuente, and K. Radermacher. “Assessment of optical localizer accuracy for computer aided surgery systems”. In: *Computer Aided Surgery* 15.1-3 (2010), pp. 1–12 (see p. 6).

[27] D. Ungureanu, F. Bogo, S. Galliani, P. Sama, X. Duan, C. Meekhof, J. Stühmer, T. J. Cashman, B. Tekin, J. L. Schönberger, et al. “HoloLens 2 Research Mode as a Tool for Computer Vision Research”. In: *arXiv preprint arXiv:2008.11239* (2020) (see p. 6).

[28] G. R. Bradski, S. A. Miller, and R. Abovitz. *Methods and systems for creating virtual and augmented reality*. US Patent 10,203,762. 2019 (see p. 6).

[29] C. R. Qi, W. Liu, C. Wu, H. Su, and L. J. Guibas. “Frustum pointnets for 3d object detection from rgb-d data”. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2018, pp. 918–927 (see pp. 6, 41).

[30] A. Geiger, P. Lenz, and R. Urtasun. “Are we ready for autonomous driving? the kitti vision benchmark suite”. In: *CVPR*. 2012 (see pp. 6, 32–34, 110).

[31] E. Raphael, R. Kiefer, P. Reisman, and G. Hayon. “Development of a camera-based forward collision alert system”. In: *SAE International Journal of Passenger Cars-Mechanical Systems* 4.2011-01-0579 (2011), pp. 467–478 (see p. 6).

[32] H. Sossa, E. Zamora, et al. “Vision-Based Blind Spot Warning System by Deep Neural Networks”. In: *Mexican Conference on Pattern Recognition*. Springer. 2020, pp. 185–194 (see p. 6).

[33] L. Keselman, J. Iselin Woodfill, A. Grunnet-Jepsen, and A. Bhowmik. “Intel realsense stereoscopic depth cameras”. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*. 2017, pp. 1–10 (see p. 6).

[34] Z. Zhang. “Microsoft kinect sensor and its effect”. In: *IEEE multimedia* 19.2 (2012), pp. 4–10 (see p. 6).

[35] E. Marder-Eppstein. “Project tango”. In: *ACM SIGGRAPH 2016 Real-Time Live!* 2016, pp. 25–25 (see p. 6).

[36] R. Kesten, M. Usman, J. Houston, T. Pandya, K. Nadhamuni, A. Ferreira, M. Yuan, B. Low, A. Jain, P. Ondruska, S. Omari, S. Shah, A. Kulkarni, A. Kazakova, C. Tao, L. Platinsky, W. Jiang, and V. Shet. *Lyft Level 5 Perception Dataset 2020*. https://level5.lyft.com/dataset/. 2019 (see pp. 6, 32).

[37] A. Kadambi, A. Bhandari, and R. Raskar. “3D Depth Cameras in Vision: Benefits and Limitations of the Hardware”. In: *Computer Vision and Machine Learning with RGB-D Sensors*. Ed. by L. Shao, J. Han, P. Kohli, and Z. Zhang. Cham: Springer International Publishing, 2014, pp. 3–26 (see p. 6).

[38] S. K. Nayar and M. Gupta. “Diffuse structured light”. In: *ICCP*. IEEE. 2012, pp. 1–11 (see p. 6).

[39] M. Gupta, A. Agrawal, A. Veeraraghavan, and S. G. Narasimhan. “Structured light 3D scanning in the presence of global illumination”. In: *CVPR*. IEEE. 2011, pp. 713–720 (see p. 6).

[40] P. Pinggera, D. Pfeiffer, U. Franke, and R. Mester. “Know your limits: Accuracy of long range stereoscopic object measurements in practice”. In: *ECCV*. Springer. 2014, pp. 96–111 (see p. 7).

[41] J. Uhrig, N. Schneider, L. Schneider, U. Franke, T. Brox, and A. Geiger. “Sparsity invariant cnns”. In: *3DV*. IEEE. 2017, pp. 11–20 (see p. 7).

[42] A. Lopez-Rodriguez, B. Busam, and K. Mikolajczyk. “Project to adapt: Domain adaptation for depth completion from noisy and sparse sensor data”. In: *ACCV*. 2020 (see p. 7).

[43] D. Eigen, C. Puhrsch, and R. Fergus. “Depth map prediction from a single image using a multi-scale deep network”. In: *Advances in neural information processing systems*. 2014, pp. 2366–2374 (see pp. 7, 26, 129).

[44] I. Laina, C. Rupprecht, V. Belagiannis, F. Tombari, and N. Navab. “Deeper depth prediction with fully convolutional residual networks”. In: *2016 Fourth international conference on 3D vision (3DV)*. IEEE. 2016, pp. 239–248 (see pp. 7, 26, 27, 129).

[45] R. Hartley and A Zisserman. *Multiple View Geometry in Computer Vision*. Second. Cambridge University Press, ISBN: 0521540518, 2004 (see p. 9).

[46] I. Goodfellow, Y. Bengio, A. Courville, and Y. Bengio. *Deep learning*. Vol. 1. 2. MIT press Cambridge, 2016 (see p. 10).

[47] F. Rosenblatt. “The Perceptron: {A} Probabilistic Model for Information Storage and Organization in the Brain”. In: *Psychological Review* 65 (1958), pp. 386–408 (see pp. 10, 11).

[48] W. S. McCulloch and W. Pitts. “A logical calculus of the ideas immanent in nervous activity”. In: *The bulletin of mathematical biophysics* 5.4 (1943), pp. 115–133 (see p. 10).

[49] M. Minsky and S. Papert. *Perceptrons: An Introduction to Computational Geometry*. MIT Press, 1969 (see p. 10).

[50] G. E. Hinton et al. “Learning distributed representations of concepts”. In: *Proceedings of the eighth annual conference of the cognitive science society*. Vol. 1. Amherst, MA. 1986, p. 12 (see p. 11).

[51] D. E. Rumelhart, G. E. Hinton, and R. J. Williams. “Learning representations by back-propagating errors”. In: *nature* 323.6088 (1986), pp. 533–536 (see pp. 11, 12).

[52] G. E. Hinton, S. Osindero, and Y.-W. Teh. “A fast learning algorithm for deep belief nets”. In: *Neural computation* 18.7 (2006), pp. 1527–1554 (see p. 11).

[53] 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 (see pp. 11, 15, 19, 57).

[54] K. Simonyan and A. Zisserman. “Very deep convolutional networks for large-scale image recognition”. In: *arXiv preprint arXiv:1409.1556* (2014) (see pp. 11, 36, 37, 40).

[55] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich. “Going deeper with convolutions”. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2015, pp. 1–9 (see pp. 11, 36, 40).

[56] K. He, X. Zhang, S. Ren, and J. Sun. “Deep residual learning for image recognition”. In: *CVPR*. 2016, pp. 770–778 (see pp. 11, 15, 38).

[57] C. Szegedy, S. Ioffe, V. Vanhoucke, and A. A. Alemi. “Inception-v4, inception-resnet and the impact of residual connections on learning”. In: *Thirty-First AAAI Conference on Artificial Intelligence*. 2017 (see pp. 11, 36, 57).

[58] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi. “You only look once: Unified, real-time object detection”. In: *CVPR*. 2016 (see pp. 11, 16, 17, 36, 38).

[59] W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. Reed, C.-Y. Fu, and A. C. Berg. “SSD: Single Shot MultiBox Detector”. In: *ECCV*. 2016 (see pp. 11, 16, 17, 35–37, 44, 93, 94).

[60] X. Huang and S. Belongie. “Arbitrary style transfer in real-time with adaptive instance normalization”. In: *Proceedings of the IEEE International Conference on Computer Vision*. 2017, pp. 1501–1510 (see p. 11).

[61] J. J. Park, P. Florence, J. Straub, R. Newcombe, and S. Lovegrove. “Deepsdf: Learning continuous signed distance functions for shape representation”. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2019, pp. 165–174 (see pp. 11, 20).

[62] T. Groueix, M. Fisher, V. G. Kim, B. C. Russell, and M. Aubry. “A papier-mâché approach to learning 3d surface generation”. In: *CVPR*. 2018, pp. 216–224 (see pp. 11, 131).

[63] S. Ioffe and C. Szegedy. “Batch normalization: Accelerating deep network training by reducing internal covariate shift”. In: *arXiv preprint arXiv:1502.03167* (2015) (see pp. 13, 15).

[64] Y. Wu and K. He. “Group normalization”. In: *ECCV*. 2018, pp. 3–19 (see pp. 13, 15).

[65] N. Qian. “On the momentum term in gradient descent learning algorithms”. In: *Neural networks* 12.1 (1999), pp. 145–151 (see p. 13).

[66] Y. E. Nesterov. “A method for solving the convex programming problem with convergence rate O (1/kˆ 2)”. In: *Dokl. akad. nauk Sssr*. Vol. 269. 1983, pp. 543–547 (see p. 13).

[67] J. Duchi, E. Hazan, and Y. Singer. “Adaptive subgradient methods for online learning and stochastic optimization.” In: *Journal of machine learning research* 12.7 (2011) (see p. 14).

[68] M. D. Zeiler. “Adadelta: an adaptive learning rate method”. In: *arXiv preprint arXiv:1212.5701* (2012) (see p. 14).

[69] D. P. Kingma and J. Ba. “Adam: A method for stochastic optimization”. In: *arXiv preprint arXiv:1412.6980* (2014) (see p. 14).

[70] S. Ruder. “An overview of gradient descent optimization algorithms”. In: *arXiv preprint arXiv:1609.04747* (2016) (see p. 14).

[71] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner. “Gradient-based learning applied to document recognition”. In: *Proceedings of the IEEE* 86.11 (1998), pp. 2278–2324 (see p. 14).

[72] K. Fukushima. “Neocognitron: A Self-Organizing Neural Network Model for a Mechanism of Pattern Recognition Unaffected by Shift in Position”. In: *Biological Cybernetics* 36.4 (Apr. 1980), pp. 193–202 (see p. 14).

[73] Z. Tian, C. Shen, H. Chen, and T. He. “FCOS: Fully Convolutional One-Stage Object Detection”. In: *ICCV*. 2019, pp. 9627–9636 (see pp. 16, 17, 36).

[74] 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 (see pp. 16, 17, 36, 38, 40, 44).

[75] T.-Y. Lin, P. Goyal, R. Girshick, K. He, and P. Dollár. “Focal loss for dense object detection”. In: *ICCV*. 2017 (see pp. 16, 17, 40, 109, 131).

[76] K. He, G. Gkioxari, P. Dollár, and R. Girshick. “Mask r-cnn”. In: *ICCV*. 2017 (see pp. 16, 29, 36, 40, 41, 44, 109).

[77] H. Law and J. Deng. “Cornernet: Detecting objects as paired keypoints”. In: *Proceedings of the European Conference on Computer Vision (ECCV)*. 2018, pp. 734–750 (see pp. 16, 17).

[78] K. Duan, S. Bai, L. Xie, H. Qi, Q. Huang, and Q. Tian. “Centernet: Keypoint triplets for object detection”. In: *Proceedings of the IEEE International Conference on Computer Vision*. 2019, pp. 6569–6578 (see pp. 16, 17).

[79] X. Zhou, D. Wang, and P. Krähenbühl. “Objects as Points”. In: *arXiv preprint arXiv:1904.07850*. 2019 (see p. 16).

[80] R. Girshick, J. Donahue, T. Darrell, U. C. Berkeley, and J. Malik. “Rich feature hierarchies for accurate object detection and semantic segmentation”. In: *CVPR*. 2014 (see p. 16).

[81] R. Girshick. “Fast R-CNN”. In: *Proceedings of the IEEE international conference on computer vision*. 2015, pp. 1440–1448 (see pp. 16, 29).

[82] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio. “Generative adversarial nets”. In: *Advances in neural information processing systems*. 2014, pp. 2672–2680 (see pp. 17, 18).

[83] D. Pathak, P. Krahenbuhl, J. Donahue, T. Darrell, and A. A. Efros. “Context Encoders: Feature Learning by Inpainting”. In: *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2016 (see pp. 17, 19).

[84] S. Iizuka, E. Simo-Serra, and H. Ishikawa. “Globally and locally consistent image completion”. In: *ACM Transactions on Graphics (TOG)* 36.4 (2017), p. 107 (see p. 17).

[85] J.-Y. Zhu, P. Krähenbühl, E. Shechtman, and A. A. Efros. “Generative Visual Manipulation on the Natural Image Manifold”. In: *ECCV*. 2016 (see p. 17).

[86] C. Li and M. Wand. “Precomputed Real-Time Texture Synthesis with Markovian Generative Adversarial Networks”. In: *ECCV*. Ed. by B. Leibe, J. Matas, N. Sebe, and M. Welling. 2016 (see p. 17).

[87] P. Isola, J.-Y. Zhu, T. Zhou, and A. A. Efros. “Image-to-image translation with conditional adversarial networks”. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2017, pp. 1125–1134 (see pp. 17–19).

[88] J.-Y. Zhu, T. Park, P. Isola, and A. A. Efros. “Unpaired Image-to-Image Translation using Cycle-Consistent Adversarial Networks”. In: (2017). arXiv: 1703.10593 (see p. 17).

[89] C. Ledig, L. Theis, F. Huszar, J. Caballero, A. P. Aitken, A. Tejani, J. Totz, Z. Wang, and W. Shi. “Photo-Realistic Single Image Super-Resolution Using a Generative Adversarial Network”. In: *CoRR* abs/1609.04802 (2016). arXiv: 1609.04802 (see pp. 17, 19).

[90] J. Wu, C. Zhang, T. Xue, W. T. Freeman, and J. B. Tenenbaum. “Learning a Probabilistic Latent Space of Object Shapes via 3D Generative-Adversarial Modeling”. In: *Advances In Neural Information Processing Systems*. 2016, pp. 82–90 (see p. 18).

[91] M. Mirza and S. Osindero. “Conditional generative adversarial nets”. In: *arXiv preprint arXiv:1411.1784* (2014) (see p. 18).

[92] J. Z. Bingning Wang Kang Liu. “Conditional Generative Adversarial Networks for Commonsense Machine Comprehension”. In: *Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence, IJCAI-17*. 2017, pp. 4123–4129 (see p. 18).

[93] X. Wang and A. Gupta. “Generative Image Modeling using Style and Structure Adversarial Networks”. In: *ECCV* (2016) (see p. 19).

[94] S. E. Reed, Z. Akata, X. Yan, L. Logeswaran, B. Schiele, and H. Lee. “Generative Adversarial Text to Image Synthesis”. In: *CoRR* abs/1605.05396 (2016). arXiv: 1605.05396 (see p. 19).

[95] E. L. Denton, S. Chintala, a. szlam, and R. Fergus. “Deep Generative Image Models using a Laplacian Pyramid of Adversarial Networks”. In: *NIPS*. 2015, pp. 1486–1494 (see p. 19).

[96] 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: *ECCV*. 2014, pp. 740–755 (see p. 19).

[97] D. Beker, H. Kato, M. A. Morariu, T. Ando, T. Matsuoka, W. Kehl, and A. Gaidon. “Monocular Differentiable Rendering for Self-Supervised 3D Object Detection”. In: *European Conference on Computer Vision*. 2020 (see pp. 20, 32, 40).

[98] A. Kanazawa, S. Tulsiani, A. A. Efros, and J. Malik. “Learning Category-Specific Mesh Reconstruction from Image Collections”. In: *ECCV*. 2018 (see p. 20).

[99] S. Marschner and P. Shirley. *Fundamentals of computer graphics*. CRC Press, 2015 (see p. 20).

[100] Y. Zhou and O. Tuzel. “Voxelnet: End-to-end learning for point cloud based 3d object detection”. In: *CVPR*. 2018, pp. 4490–4499 (see p. 20).

[101] C. R. Qi, H. Su, K. Mo, and L. J. Guibas. “Pointnet: Deep learning on point sets for 3d classification and segmentation”. In: *CVPR*. 2017 (see pp. 20, 36, 41, 131).

[102] T. H. Nguyen-Phuoc, C. Li, S. Balaban, and Y. Yang. “Rendernet: A deep convolutional network for differentiable rendering from 3d shapes”. In: *Advances in Neural Information Processing Systems*. 2018, pp. 7891–7901 (see p. 20).

[103] H. Kato, D. Beker, M. Morariu, T. Ando, T. Matsuoka, W. Kehl, and A. Gaidon. “Differentiable Rendering: A Survey”. In: *arXiv preprint arXiv:2006.12057* (2020) (see p. 20).

[104] M. M. Loper and M. J. Black. “OpenDR: An Approximate Differentiable Renderer”. In: *ECCV*. Vol. 8695. 2014, pp. 154–169 (see pp. 20, 21).

[105] H. Kato, Y. Ushiku, and T. Harada. “Neural 3d mesh renderer”. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2018, pp. 3907–3916 (see pp. 20, 21, 40).

[106] S. Liu, T. Li, W. Chen, and H. Li. “Soft Rasterizer: A Differentiable Renderer for Image-based 3D Reasoning”. In: *ICCV* (2019), pp. 7708–7717 (see pp. 20, 21).

[107] W. Chen, H. Ling, J. Gao, E. Smith, J. Lehtinen, A. Jacobson, and S. Fidler. “Learning to predict 3d objects with an interpolation-based differentiable renderer”. In: *NeurIPS*. 2019, pp. 9605–9616 (see pp. 20, 21).

[108] X. Yan, J. Yang, E. Yumer, Y. Guo, and H. Lee. “Perspective transformer nets: Learning single-view 3d object reconstruction without 3d supervision”. In: *Advances in neural information processing systems*. 2016, pp. 1696–1704 (see p. 21).

[109] S. Tulsiani, T. Zhou, A. A. Efros, and J. Malik. “Multi-view supervision for single-view reconstruction via differentiable ray consistency”. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2017, pp. 2626–2634 (see p. 21).

[110] O. Wiles, G. Gkioxari, R. Szeliski, and J. Johnson. “Synsin: End-to-end view synthesis from a single image”. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2020, pp. 7467–7477 (see p. 21).

[111] E. Insafutdinov and A. Dosovitskiy. “Unsupervised learning of shape and pose with differentiable point clouds”. In: *Advances in neural information processing systems*. 2018, pp. 2802–2812 (see p. 21).

[112] S. Zakharov, W. Kehl, A. Bhargava, and A. Gaidon. “Autolabeling 3D Objects With Differentiable Rendering of SDF Shape Priors”. In: *CVPR*. June 2020 (see p. 21).

[113] B. Mildenhall, P. P. Srinivasan, M. Tancik, J. T. Barron, R. Ramamoorthi, and R. Ng. “Nerf: Representing scenes as neural radiance fields for view synthesis”. In: *European Conference on Computer Vision*. 2020 (see p. 21).

[114] A. Tewari, O. Fried, J. Thies, V. Sitzmann, S. Lombardi, K. Sunkavalli, R. Martin-Brualla, T. Simon, J. Saragih, M. Nießner, et al. “State of the Art on Neural Rendering”. In: *arXiv preprint arXiv:2004.03805* (2020) (see p. 21).

[115] B. Xu and Z. Chen. “Multi-Level Fusion Based 3D Object Detection From Monocular Images”. In: *CVPR*. 2018 (see pp. 26, 40).

[116] 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: *CVPR*. 2019 (see pp. 26, 41, 129, 134).

[117] A. Simonelli, S. Rota Bulo, L. Porzi, M. Lopez-Antequera, and P. Kontschieder. “Disentangling Monocular 3D Object Detection”. In: *ICCV*. 2019 (see pp. 26, 39, 40, 110).

[118] X. Chen, K. Kundu, Z. Zhang, H. Ma, S. Fidler, and R. Urtasun. “Monocular 3d object detection for autonomous driving”. In: *CVPR*. 2016 (see pp. 26, 32, 39, 110).

[119] X. Ma, S. Liu, Z. Xia, H. Zhang, X. Zeng, and W. Ouyang. “Rethinking pseudo-lidar representation”. In: *European Conference on Computer Vision*. Springer. 2020, pp. 311–327 (see pp. 26, 32, 39, 41).

[120] Y. Xiang, T. Schmidt, V. Narayanan, and D. Fox. “PoseCNN: A Convolutional Neural Network for 6D Object Pose Estimation in Cluttered Scenes”. In: *RSS* (2018) (see pp. 27, 28, 30, 31, 36, 44, 57, 93, 109).

[121] C. Godard, O. Mac Aodha, and G. J. Brostow. “Unsupervised monocular depth estimation with left-right consistency”. In: *CVPR*. 2017, pp. 270–279 (see pp. 27, 40).

[122] 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 (see p. 28).

[123] Y. Labbé, J. Carpentier, M. Aubry, and J. Sivic. “CosyPose: Consistent multi-view multi-object 6D pose estimation”. In: *European Conference on Computer Vision*. Springer. 2020, pp. 574–591 (see pp. 28, 36, 44, 79).

[124] B. Tekin, S. N. Sinha, and P. Fua. "Real-Time Seamless Single Shot 6D Object Pose Prediction". In: *CVPR*. 2018, pp. 292–301 (see pp. 29, 36, 38, 44, 57).

[125] S. Zakharov, I. Shugurov, and S. Ilic. "Dpod: Dense 6d pose object detector in rgb images". In: *ICCV*. 2019 (see pp. 29, 36, 38).

[126] V. Lepetit, F. Moreno-Noguer, and P. Fua. "EPnP: An Accurate O(N) Solution to the PnP Problem". In: *Int. J. Comput. Vision* 81.2 (Feb. 2009), pp. 155–166 (see pp. 29, 37).

[127] 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 (see pp. 29, 37).

[128] Z. Li, G. Wang, and X. Ji. "CDPN: Coordinates-Based Disentangled Pose Network for Real-Time RGB-Based 6-DoF Object Pose Estimation". In: *ICCV*. 2019, pp. 7678–7687 (see pp. 29, 30, 36, 38, 39, 44).

[129] A. Kundu, Y. Li, and J. M. Rehg. "3d-rcnn: Instance-level 3d object reconstruction via render-and-compare". In: *CVPR*. 2018 (see pp. 29, 39, 109, 110).

[130] M. Sundermeyer, Z.-C. Marton, M. Durner, M. Brucker, and R. Triebel. "Implicit 3D Orientation Learning for 6D Object Detection from RGB Images". In: *ECCV*. 2018 (see pp. 30, 36, 37).

[131] 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*. Springer. 2012, pp. 548–562 (see pp. 30–33, 35, 43, 44, 58, 80, 93).

[132] E. Brachmann, A. Krull, F. Michel, S. Gumhold, J. Shotton, and C. Rother. "Learning 6D Object Pose Estimation Using 3D Object Coordinates". In: *ECCV*. 2014, pp. 536–551 (see pp. 30, 31, 35–37).

[133] T. Hodan, 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: *WACV*. 2017, pp. 880–888 (see pp. 31, 80).

[134] B. Drost, M. Ulrich, P. Bergmann, P. Hartinger, and C. Steger. "Introducing mvtec itodd-a dataset for 3d object recognition in industry". In: *Proceedings of the IEEE International Conference on Computer Vision Workshops*. 2017, pp. 2200–2208 (see p. 31).

[135] R. Kaskman, S. Zakharov, I. Shugurov, and S. Ilic. "HomebrewedDB: RGB-D Dataset for 6D Pose Estimation of 3D Objects". In: *ICCVW*. 2019, pp. 0–0 (see pp. 31, 44).

[136] 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: *IEEE Robotics and Automation Letters* 1.2 (2016), pp. 1179–1185 (see p. 31).

[137] A. Doumanoglou, R. Kouskouridas, S. Malassiotis, and T.-K. Kim. "6D Object Detection and Next-Best-View Prediction in the Crowd". In: *CVPR*. 2016 (see p. 31).

[138] A. Tejani, D. Tang, R. Kouskouridas, and T.-k. Kim. "Latent-class hough forests for 3D object detection and pose estimation". In: *ECCV*. 2014 (see pp. 31, 44, 58).

[139] C. Song, J. Song, and Q. Huang. "Hybridpose: 6d object pose estimation under hybrid representations". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2020, pp. 431–440 (see pp. 30, 36, 38).

[140] H. Caesar, V. Bankiti, A. H. Lang, S. Vora, V. E. Liong, Q. Xu, A. Krishnan, Y. Pan, G. Baldan, and O. Beijbom. "nuscenes: A multimodal dataset for autonomous driving". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2020, pp. 11621–11631 (see p. 32).

[141] M.-F. Chang, J. Lambert, P. Sangkloy, J. Singh, S. Bak, A. Hartnett, D. Wang, P. Carr, S. Lucey, D. Ramanan, et al. "Argoverse: 3d tracking and forecasting with rich maps". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2019, pp. 8748–8757 (see p. 32).

[142] P. Sun, H. Kretzschmar, X. Dotiwalla, A. Chouard, V. Patnaik, P. Tsui, J. Guo, Y. Zhou, Y. Chai, B. Caine, et al. "Scalability in perception for autonomous driving: Waymo open dataset". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2020, pp. 2446–2454 (see p. 32).

[143] T. Hodaň, J. Matas, and Š. Obdržálek. "On Evaluation of 6D Object Pose Estimation". In: *ECCVW* (2016), pp. 606–619 (see p. 33).

[144] M. Everingham, L. Van Gool, C. K. Williams, J. Winn, and A. Zisserman. "The pascal visual object classes (voc) challenge". In: *IJCV* 88.2 (2010), pp. 303–338 (see p. 34).

[145] D. G. Lowe. "Object recognition from local scale-invariant features". In: *ICCV*. Vol. 2. 1999, pp. 1150–1157 (see p. 35).

[146] A. C. Romea, M. M. Torres, and S. Srinivasa. "The MOPED framework: Object recognition and pose estimation for manipulation". In: *International Journal of Robotics Research* 30.10 (Sept. 2011), pp. 1284–1306 (see p. 35).

[147] 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: *TPAMI* 34.5 (2012), pp. 876–888 (see p. 35).

[148] J. Vidal, C.-Y. Lin, X. Lladó, and R. Martí. "A method for 6D pose estimation of free-form rigid objects using point pair features on range data". In: *Sensors* 18.8 (2018), p. 2678 (see p. 35).

[149] S. Hinterstoisser, V. Lepetit, N. Rajkumar, and K. Konolige. "Going further with point pair features". In: *ECCV*. 2016, pp. 834–848 (see p. 35).

[150] A. Krull, E. Brachmann, F. Michel, M. Ying Yang, S. Gumhold, and C. Rother. "Learning analysis-by-synthesis for 6D pose estimation in RGB-D images". In: *ICCV*. 2015, pp. 954–962 (see p. 35).

[151] T. Do, M. Cai, T. Pham, and I. D. Reid. "Deep-6DPose: Recovering 6D Object Pose from a Single RGB Image". In: *CoRR* abs/1802.10367 (2018). arXiv: 1802.10367 (see p. 36).

[152] P. Wohlhart and V. Lepetit. "Learning Descriptors for Object Recognition and 3D Pose Estimation". In: *CVPR*. 2015 (see pp. 36, 37).

[153] W. Kehl, F. Tombari, N. Navab, S. Ilic, and V. Lepetit. "Hashmod: A Hashing Method for Scalable 3D Object Detection". In: *BMVC*. 2015 (see p. 36).

[154] S. Zakharov, W. Kehl, B. Planche, A. Hutter, and S. Ilic. "3d object instance recognition and pose estimation using triplet loss with dynamic margin". In: *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE. 2017, pp. 552–559 (see pp. 36, 37).

[155] 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: *CVPR*. 2020, pp. 13916–13925 (see pp. 36, 37).

[156] E. Brachmann, F. Michel, A. Krull, M. Ying Yang, S. Gumhold, and C. Rother. "Uncertainty-driven 6D pose estimation of objects and scenes from a single RGB image". In: *CVPR*. 2016, pp. 3364–3372 (see pp. 36, 37).

[157] M. Oberweger, M. Rad, and V. Lepetit. "Making deep heatmaps robust to partial occlusions for 3d object pose estimation". In: *Proceedings of the European Conference on Computer Vision (ECCV)*. 2018, pp. 119–134 (see pp. 36, 38).

[158] S. Peng, Y. Liu, Q. Huang, X. Zhou, and H. Bao. "PVNet: Pixel-wise Voting Network for 6DoF Pose Estimation". In: *CVPR*. 2019 (see pp. 36, 38).

[159] K. Park, T. Patten, and M. Vincze. "Pix2Pose: Pixel-Wise Coordinate Regression of Objects for 6D Pose Estimation". In: *ICCV*. 2019 (see pp. 36, 38, 93).

[160] O. Ronneberger, P. Fischer, and T. Brox. "U-net: Convolutional networks for biomedical image segmentation". In: *International Conference on Medical image computing and computer-assisted intervention*. Springer. 2015, pp. 234–241 (see pp. 36, 38).

[161] J. Redmon and A. Farhadi. "Yolov3: An incremental improvement". In: *arXiv preprint arXiv:1804.02767* (2018) (see pp. 36, 38).

[162] 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 (see pp. 36, 38).

[163] T. Hodan, D. Barath, and J. Matas. "EPOS: Estimating 6D Pose of Objects with Symmetries". In: *CVPR*. 2020, pp. 11703–11712 (see pp. 36, 38, 39, 79).

[164] L.-C. Chen, G. Papandreou, F. Schroff, and H. Adam. "Rethinking atrous convolution for semantic image segmentation". In: *arXiv preprint arXiv:1706.05587* (2017) (see p. 36).

[165] Y. Li, G. Wang, X. Ji, Y. Xiang, and D. Fox. "DeepIM: Deep Iterative Matching for 6D Pose Estimation". In: *IJCV* (2019), pp. 1–22 (see p. 36).

[166] 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: *ECCV*. 2016 (see p. 37).

[167] Y. Hu, J. Hugonot, P. Fua, and M. Salzmann. "Segmentation-driven 6D Object Pose Estimation". In: *CVPR*. 2019, pp. 3385–3394 (see p. 38).

[168] D. Barath and J. Matas. "Graph-cut RANSAC". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2018, pp. 6733–6741 (see p. 39).

[169] D. Barath and J. Matas. "Progressive-X: Efficient, anytime, multi-model fitting algorithm". In: *Proceedings of the IEEE International Conference on Computer Vision*. 2019, pp. 3780–3788 (see p. 39).

[170] Y. Wang, W.-L. Chao, D. Garg, B. Hariharan, M. Campbell, and K. Q. Weinberger. "Pseudo-lidar from visual depth estimation: Bridging the gap in 3d object detection for autonomous driving". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2019, pp. 8445–8453 (see pp. 39, 41).

[171] X. Ma, Z. Wang, H. Li, W. Ouyang, and P. Zhang. "Accurate monocular 3D object detection via Color-Embedded 3D reconstruction for autonomous driving". In: *ICCV*. 2019 (see pp. 39, 41).

[172] B. Li, W. Ouyang, L. Sheng, X. Zeng, and X. Wang. "Gs3d: An efficient 3d object detection framework for autonomous driving". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2019, pp. 1019–1028 (see p. 39).

[173] A. Simonelli, S. R. Bulò, L. Porzi, E. Ricci, and P. Kontschieder. "Single-stage monocular 3d object detection with virtual cameras". In: *arXiv preprint arXiv:1912.08035* (2019) (see p. 40).

[174] J. Ku, A. D. Pon, and S. L. Waslander. "Monocular 3D Object Detection Leveraging Accurate Proposals and Shape Reconstruction". In: *CVPR*. 2019 (see p. 40).

[175] G. Brazil and X. Liu. "M3d-rpn: Monocular 3d region proposal network for object detection". In: *Proceedings of the IEEE International Conference on Computer Vision*. 2019, pp. 9287–9296 (see p. 40).

[176] Y. Chen, L. Tai, K. Sun, and M. Li. "MonoPair: Monocular 3D Object Detection Using Pairwise Spatial Relationships". In: *CVPR*. 2020, pp. 12093–12102 (see p. 40).

[177] S. Pillai, R. Ambruş, and A. Gaidon. “Superdepth: Self-supervised, super-resolved monocular depth estimation”. In: *ICRA*. 2019, pp. 9250–9256 (see p. 40).

[178] Z. Qin, J. Wang, and Y. Lu. “Monogrnet: A geometric reasoning network for monocular 3d object localization”. In: *Proceedings of the AAAI Conference on Artificial Intelligence*. Vol. 33. 2019, pp. 8851–8858 (see p. 40).

[179] V. Guizilini, R. Ambrus, S. Pillai, A. Raventos, and A. Gaidon. “3D Packing for Self-Supervised Monocular Depth Estimation”. In: *CVPR*. June 2020 (see p. 40).

[180] M. Ding, Y. Huo, H. Yi, Z. Wang, J. Shi, Z. Lu, and P. Luo. “Learning depth-guided convolutions for monocular 3d object detection”. In: *CVPRW*. 2020, pp. 1000–1001 (see p. 40).

[181] H. Fu, M. Gong, C. Wang, K. Batmanghelich, and D. Tao. “Deep ordinal regression network for monocular depth estimation”. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2018, pp. 2002–2011 (see p. 41).

[182] J. Ku, M. Mozifian, J. Lee, A. Harakeh, and S. L. Waslander. “Joint 3d proposal generation and object detection from view aggregation”. In: *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE. 2018, pp. 1–8 (see pp. 41, 109).

[183] S. Umeyama. “Least-squares estimation of transformation parameters between two point patterns”. In: *IEEE Transactions on Pattern Analysis & Machine Intelligence* (1991), pp. 376–380 (see p. 41).

[184] D. Chen, J. Li, Z. Wang, and K. Xu. “Learning canonical shape space for category-level 6d object pose and size estimation”. In: *CVPR*. 2020, pp. 11973–11982 (see pp. 41, 129, 134).

[185] M. Tian, M. H. Ang, and G. H. Lee. “Shape Prior Deformation for Categorical 6D Object Pose and Size Estimation”. In: *European Conference on Computer Vision*. Springer. 2020, pp. 530–546 (see pp. 41, 129).

[186] K. Park, A. Mousavian, Y. Xiang, and D. Fox. “LatentFusion: End-to-End Differentiable Reconstruction and Rendering for Unseen Object Pose Estimation”. In: *CVPR*. 2020, pp. 10710–10719 (see p. 42).

[187] P. Besl and N. McKay. “A Method for Registration of 3-D Shapes”. In: *TPAMI* (1992) (see p. 57).

[188] V. A. Prisacariu and I. D. Reid. “PWP3D: Real-Time Segmentation and Tracking of 3D Objects”. In: *IJCV* (2012) (see p. 57).

[189] H. Tjaden, U. Schwanecke, and E. Schoemer. “Real-Time Monocular Segmentation and Pose Tracking of Multiple Objects”. In: *ECCV*. 2016 (see pp. 57, 58).

[190] W. Kehl, F. Tombari, S. Ilic, and N. Navab. “Real-Time 3D Model Tracking in Color and Depth on a Single CPU Core”. In: *CVPR*. 2017 (see p. 57).

[191] M. Garon and J.-F. Lalonde. “Deep 6-DOF tracking”. In: *IEEE Transactions on Visualization and Computer Graphics* 23.11 (2017), pp. 2410–2418 (see p. 58).

[192] C. Rupprecht, I. Laina, R. DiPietro, M. Baust, F. Tombari, N. Navab, and G. D. Hager. “Learning in an Uncertain World: Representing Ambiguity Through Multiple Hypotheses”. In: *ICCV*. 2017 (see p. 79).

[193] M. Rad, P. M. Roth, and V. Lepetit. “ALCN: Adaptive Local Contrast Normalization for Robust Object Detection and 3D Pose Estimation”. In: *BMVC*. 2017 (see p. 93).

[194] Y. Jiang, X. Gong, D. Liu, Y. Cheng, C. Fang, X. Shen, J. Yang, P. Zhou, and Z. Wang. “EnlightenGAN: Deep Light Enhancement without Paired Supervision”. In: *arXiv preprint arXiv:1906.06972* (2019) (see p. 93).

[195] C. Wei, W. Wang, W. Yang, and J. Liu. "Deep retinex decomposition for low-light enhancement". In: *arXiv preprint arXiv:1808.04560* (2018) (see p. 93).

[196] Y.-S. Chen, Y.-C. Wang, M.-H. Kao, and Y.-Y. Chuang. "Deep photo enhancer: Unpaired learning for image enhancement from photographs with gans". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2018, pp. 6306–6314 (see p. 93).

[197] J. Johnson, A. Alahi, and L. Fei-Fei. "Perceptual losses for real-time style transfer and super-resolution". In: *ECCV*. 2016, pp. 694–711 (see p. 93).

[198] X. Chen, H. Ma, J. Wan, B. Li, and T. Xia. "Multi-view 3d object detection network for autonomous driving". In: *CVPR*. 2017 (see p. 109).

[199] A. Kendall, Y. Gal, and R. Cipolla. "Multi-Task Learning Using Uncertainty to Weigh Losses for Scene Geometry and Semantics". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2018 (see p. 110).

[200] Z. Liu, Z. Wu, and R. Tóth. "SMOKE: Single-Stage Monocular 3D Object Detection via Keypoint Estimation". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*. 2020, pp. 996–997 (see p. 110).

[201] 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: *ICIP* (2019) (see pp. 129, 130).

[202] 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) (see p. 129).

[203] J. Wald, A. Avetisyan, N. Navab, F. Tombari, and M. Nießner. "Rio: 3d object instance re-localization in changing indoor environments". In: *Proceedings of the IEEE International Conference on Computer Vision*. 2019, pp. 7658–7667 (see p. 130).

List of Figures

List of Tables