2D and 3D Geometric Attributes Estimation in Images via deep learning

Xuchong Qiu

École Doctorale Paris-Est

Mathématiques & Sciences et Technologies

de l'Information et de la Communication

Thèse de doctorat
de l'Université Paris-Est

Domaine : Traitement du Signal et des Images

Présentée par

Xuchong QIU

pour obtenir le grade de

Docteur de l'Université Paris-Est

2D and 3D Geometric Attributes Estimation in Images via Deep Learning

---

Soutenue publiquement le 11 Février 2021 devant le jury composé de :

Eric Marchand	Professor, University of Rennes 1	Rapporteur
Christian Wolf	Associate Professor, INSA de Lyon	Rapporteur
Diane Larlus	Principal Scientist, Naver Labs Europe	Examinatrice
Vincent Lepetit	Director of Research, École des Ponts ParisTech	Examinateur
Renaud Marlet	Senior Researcher, École des Ponts ParisTech	Directeur de thèse
Chaohui Wang	Associate Professor, University Gustave Eiffel	Co-directeur de thèse

Abstract

The visual perception of 2D and 3D geometric attributes (e.g. translation, rotation, spatial size and etc.) is important in robotic applications. It helps robotic system build knowledge about objects of interest and its surrounding environment, and can serve as the input for down-stream tasks such as object localization, scene understanding and motion planning.

The main goal of this thesis is to automatically detect 2D and 3D geometric attributes of objects and the environment. In particular, we are interested in the low-level task of estimating occlusion relationship in single images and the high-level tasks of object visual tracking and object pose estimation.

The first focus is to track the object of interest with correct locations and sizes in a given video. We first study systematically the tracking framework based on discriminative correlation filter (DCF) and propose to leverage semantics information in two tracking stages: the visual feature encoding stage and the target localization stage. Our experiments demonstrate that the involvement of semantics improves the performance of both localization and size estimation in our DCF-based tracking framework. We also make an analysis for failure cases.

The second focus is using object shape information to improve the performance of object 6D pose estimation and do object pose refinement. We propose to estimate the 2D projections of object 3D surface points with deep models to recover object 6D poses. Our results show that the proposed method benefits from the properties of these 3D-to-2D point correspondences and achieves better performance. As a second part, we study the constraints of existing object pose refinement methods and develop a pose refinement method for objects in the wild. Our experiments demonstrate that our models trained on either real data or generated synthetic data can refine pose estimates for objects in the wild, even though these objects are not seen during training.

The third focus is studying geometric occlusion in single images to better discriminate objects in the scene. We first formalize geometric occlusion definition and propose a method to automatically generate high-quality occlusion annotations. Then we propose a new occlusion relationship formulation (i.e. P2ORM) and the corresponding inference method. Experiments on occlusion reasoning benchmarks demonstrate the superiority of the proposed formulation and method. To recover accurate depth discontinuities, we also propose a depth map refinement method and a single-stage monocular depth estimation method. By using the estimates of occlusion relationship as guidance, these two methods achieve the state-of-the-art performance.

All the methods that we propose leverage on the versatility and power of deep learning. This should facilitate their integration in the visual perception module of modern robotic systems.

Besides the above methodological advances, we also made available software (for occlusion and pose estimation) and datasets (of high-quality occlusion information) as a contribution to the scientific community.

Résumé

La perception visuelle d'attributs géométriques 2D et 3D (ex. la translation, la rotation, la taille, etc.) est très importante dans les applications robotiques. Elle permet à un système robotique d'acquérir des connaissances sur des objects intéressées et son environnement, et peut fournir des entrées pour des tâches telles que la localisation d'objets, la compréhension de scènes et la planification de mouvement.

Le principal objectif de cette thèse est d'estimer des attributs géométriques 2D et 3D des objets et du environment. En particulier, nous nous intéressons à la tâche de bas niveau d'estimation de la relation d'occultation dans des images, et aux tâches de plus haut niveau de suivi visuel d'objets et d'estimation de la pose d'objets.

Le premier axe d'étude est le suivi (tracking) d'un objet d'intérêt dans une vidéo, avec des locations et tailles correctes. Tout d'abord, nous étudions attentivement le cadre du suivi d'objet basé sur des filtres de corrélation discriminants et proposons d'exploiter des informations sémantiques à deux niveaux : l'étape d'encodage des caractéristiques visuelles et l'étape de localisation de la cible. Nos expériences démontrent que l'usage de la sémantique améliore à la fois les performances de la localisation et de l'estimation de taille de l'objet suivi. Nous effectuons également des analyses pour comprendre les cas d'échec.

Le second axe d'étude est l'utilisation d'informations sur la forme des objets pour améliorer la performance de l'estimation de la pose 6D d'objets et de son raffinement. Nous proposons d'estimer avec un modèle profond les projections 2D de points 3D à la surface de l'objet, afin de pouvoir calculer la pose 6D de l'objet. Nos résultats montrent que la méthode que nous proposons bénéficie des propriétés des correspondances de points 3D à 2D et permet d'obtenir une meilleure précision des estimations. Dans un deuxième temps, nous étudions les contraintes des méthodes existantes pour raffiner la pose d'objets et développons une méthode de raffinement des objets dans des contextes arbitraires. Nos expériences montrent que nos modèles, entraînés sur des données réelles ou des données synthétiques générées, peuvent raffiner avec succès les estimations de pose pour les objets dans des contextes quelconques.

Le troisième axe de recherche est l'étude de l'occultation géométrique dans des images, dans le but de mieux pouvoir distinguer les objets dans la scène. Nous formalisons d'abord la définition de l'occultation géométrique et proposons une méthode pour générer automatiquement des annotations d'occultation de haute qualité. Ensuite, nous proposons une nouvelle formulation de la relation d'occultation (P2ORM) et une méthode d'inférence correspondante. Nos expériences sur les jeux de tests pour l'estimation d'occultations montrent la supériorité de notre formulation et de notre méthode. Afin de déterminer des discontinuités de profondeur précises, nous proposons également une méthode de raffinement de cartes de profondeur et une méthode monoculaire d'estimation de la profondeur en une étape. En utilisant l'estimation de relations d'occultation comme guide, ces deux méthodes atteignent les performances de l'état de l'art.

Toutes les méthodes que nous proposons s'appuient sur la polyvalence et la puissance de l'apprentissage profond. Cela devrait faciliter leur intégration dans le module de perception visuelle des systèmes robotiques modernes.

Outre les avancées méthodologiques mentionnées ci-dessus, nous avons également rendu

publiquement disponibles des logiciels (pour l'estimation de l'occlusion et de la pose) et des jeux de données (informations de haute qualité sur les relations d'occultation) afin de contribuer aux outils offerts à la communauté scientifique.

Acknowledgments

Renaud and Chaohui, I would like to thank you for your greatest support during my PhD as my advisors. I am greatly indebted for academic advice and guidance from both of you. Thanks Renaud for guiding me to make the right decisions at the right time and detailed discussions about my research topics. Thanks Chaohui for motivating me to come up with new research ideas and keep constant enthusiasm. Thank you for your invaluable time, attention, dedication, and inspiration.

I am grateful to the I-Site FUTURE initiative of the DiXite project for generously funding my research. I would like to thank Imagine team for providing an excellent research environment.

Many thanks go to Eric Marchand, Christian WOLF, Vincent Lepetit and Diane Larlus for agreeing to take part in my thesis jury.

I would like to thank Mattieu Aubry, Pascal Monasse, Vincent Lepetit and David Picard for your advice and suggestions about my research projects.

Thanks to Yang Xiao and Pierre-Alain Langlois. It's a pleasure to work with you and brainstorm for new ideas. I would also like to thank Xi Shen, Yuming Du and Michael Ramamonjisoa for your kind help and insightful discussions.

All members of Imagine team, thanks for the friendly lab environment and helpful talks with you.

Last but not least, I am deeply grateful to my family for their love and support.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

This thesis addresses visual perception of 2D and 3D geometric attributes (e.g., translation, rotation, spatial size, etc.) with a focus on object visual tracking, object 6D pose estimation and occlusion relationship estimation. This chapter presents the goals, the motivations, the challenges and the contributions of our work.

1.1 Goals

Our main goal is to automatically detect 2D and 3D geometric attributes in images via deep learning. In particular, we are interested in the low-level task of estimating occlusion relationship in single images and the high-level tasks of object visual tracking and object pose estimation.

The first focus of this thesis is to track the object of interest with correct locations and object sizes in a given video. Given the ground-truth bounding box of the object in the first frame of the video, the tracker should be able to predict 2D bounding boxes which successfully track the object in the subsequent frames. In particular, we try to leverage the semantics information estimated by deep semantic segmentation models in the DCF-based tracking framework. Our objective is reducing both localization errors and scale estimation errors with the involvement of semantics. In Figure 1.1,

Figure 1.1: Object visual tracking task examples where the red bounding boxes are the tracking ground truths.

we provide two examples for the object visual tracking task.

Second, with an object image and the object 3D shape, we aim to develop object 6D pose estimation methods and object pose refinement methods which offer the relative transformation between the object and the camera. The estimated transformation can be used for robot grasping tasks once the relative transformation between the camera and the robot is known. Our first interest is exploiting the 3D-to-2D correspondences between the object 3D surface and the observed object pixels to achieve a better pose estimation performance. Our second interest is improving the accuracy of coarse pose estimates for objects in the images which are captured during daily life by different people without any experimental setup. In other words, we try to alleviate the usage constraints of existing refinement methods and make our pose refinement method more applicable in real world. In Figure 1.2, some examples about the object 6D pose estimation task and the object pose refinement task are displayed.

Third, we study the low-level geometric occlusion existing in single images as we think it is a helpful information for scene understanding and discriminating objects laying in the scene. As current geometric occlusion representations lack good formulations and high-quality annotations, we first aim to develop new geometric occlusion formulations which are reasonable and facilitate automatic methods which generate

Figure 1.2: Examples of the object 6D pose estimation task and the pose refinement task. The first row presents some objects whose poses are to be estimated where the green bounding boxes are the projected object 3D bounding boxes using the ground-truth 6D poses; The second row presents the iterative pose refinement of an object, from left to right: the input image, the rendered object using the ground-truth pose, the rendered object using the initial pose estimate, the rendered object using the first pose update, the rendered object using the second pose update.

high-quality annotations. With the new occlusion formulation and the generated annotations, we try to develop a corresponding inference method which recovers geometric occlusions from single images. The new occlusion inference method can also help other scene understanding tasks such as monocular depth map estimation due to the inherent relationship between geometric occlusion and other scene geometric attributes. Thus, we also aim to develop scene understanding methods which profit from our occlusion formulation and achieve a better performance. In Figure 1.3, we present some scene geometric attributes to estimate in color images.

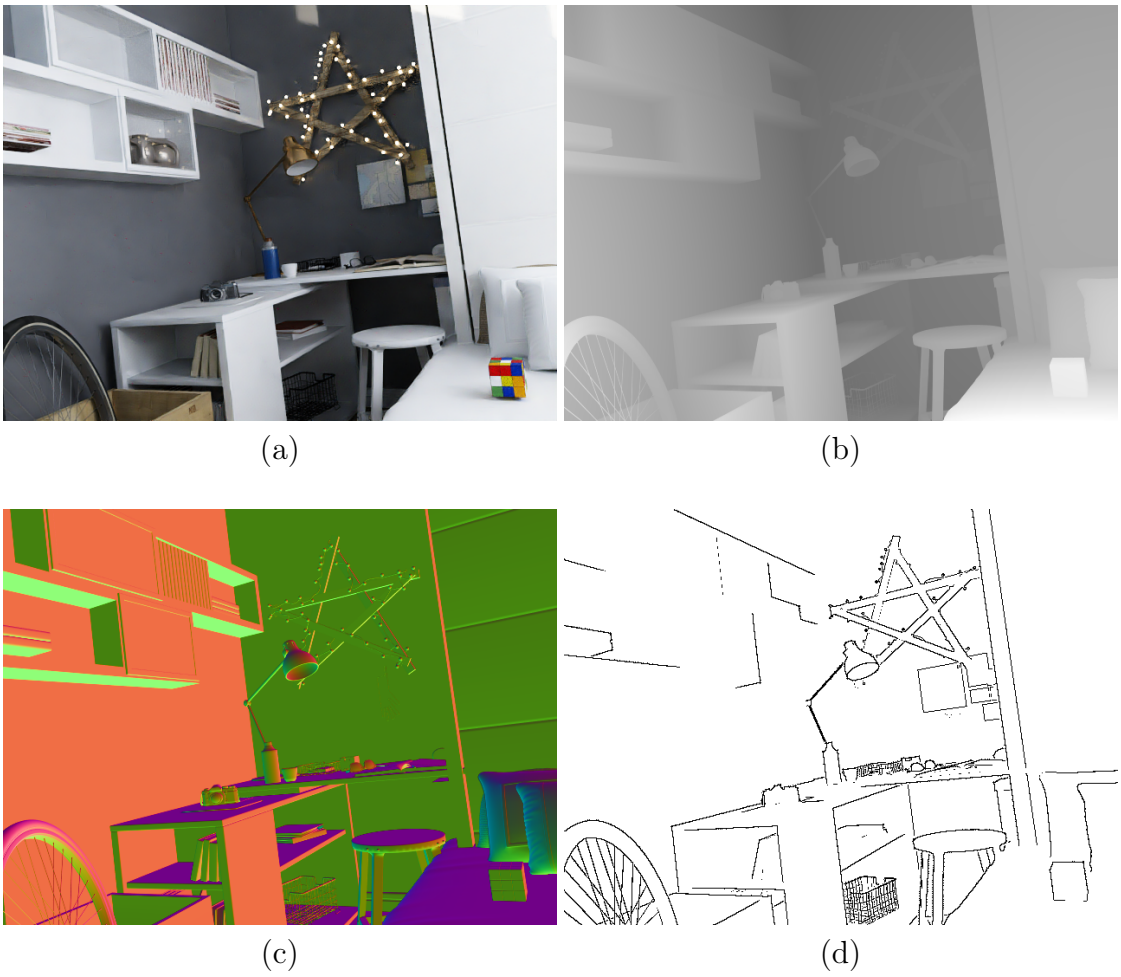


Figure 1.3: Examples of scene geometric attributes estimation tasks: (a) Input color image, (b) Ground-truth depth map, (c) Ground-truth surface normal map, (d) Ground-truth geometric occlusion boundaries.

1.2 Motivations

Computer vision research has become increasingly successful with recent advancements of deep learning techniques. Meanwhile, the computer vision research in robotics domain draws more and more attention because of its vast application prospects in daily life. Among all these related tasks, the perception of geometric attributes of the objects and of the surrounding environments plays an important role, and serve as an important building block for downstream tasks. While the input

of a robot perception system can be varied (e.g., image, depth, IR, etc.) and multimodal, doing estimations with only monocular images as input is very challenging but has the broadest potential applications due to the ubiquitous availability of RGB cameras. Considering the complexity of real world, deep-learning-based methods have a big possibility to offer better solutions for these tasks as they are data-driven and use the enormous data collected from real world. In the following, we list various reasons why the geometric attributes understanding in images is an active research area, together with the potential applications in the context of robotics.

Why geometric attributes understanding? Understanding the contents existing in images is always one core topic in computer vision. For a robotic agent in a real scene, the contents in the scene can be roughly divided into the background and the objects (including human beings) which the robot is interested in or may interact with. Understanding the geometric attributes of the objects (e.g., 2D/3D location, spatial size, orientation) help the robot keep attention on the objects of interest for other applications (e.g., object recognition, defect detection, sentiment analysis) or even interact with these objects physically (e.g., object grasping, object assembly, elderly care). Meanwhile, understanding the geometric attributes of the whole scene (e.g., depth map, surface normal map, occlusion) gives the robot some knowledge about its surrounding environment which facilitates robot navigation in the scene and some operations on the objects. In short, the understanding of geometric attributes is one of the foundation stones which help intelligent systems understand our physical world.

Applications of object visual tracking. Once the object of interest is selected or detected, the object visual tracking system offers the 2D locations and the spatial sizes of the object in an image sequence. Object visual tracking is one of the fundamental problems in computer vision and has numerous applications in real world.

In autonomous driving, the vehicle needs to track other vehicles and pedestrians in real-time to secure the security during driving. Video surveillance systems detect and track criminal suspects in videos to help the police. The entertainment industry use UAVs to track actors/actresses in outdoor environments and create amazing pictures in movies. Except for aforementioned direct applications, the object regions offered by tracking frameworks also facilitate many fine-grained applications which take the estimated regions as input.

Applications of object pose estimation. The object pose estimation system predicts the spatial position and orientation of the object of interest w.r.t. a reference coordinate system. Many applications in robotics and augmented reality are based on the knowledge of object poses. In assembly lines, industrial robots can grasp the needed components successfully when the pose between the component and the robot end-effector is given. Augmented reality games estimate the pose of objects in daily life (e.g., tables, chairs) and cast virtual contents on these objects to create joyful interactions with our real world. In autonomous driving, the poses of vehicles offer useful information for predicting the future behaviours of vehicles.

Applications of scene geometry understanding. Scene geometry understanding relates to estimating geometric attributes of the scene such as depth, occlusion and surface normal. In scene 3D reconstruction, computing depth allows us to project images captured from multiple views into 3D points where registration and matching of all the points are applied later to reconstruct the scene. Geometric occlusion appears as the depth discontinuities along the boundaries between different objects and the recovered occlusions help discriminating foreground/background in numerous applications such as the portrait mode of smartphone cameras. Microsoft HoloLens calculates the surface normal of the scene and put furniture virtually and naturally in real world to help interior designers think about new designs.

In this section, we introduce briefly the potential applications of geometric attributes estimation from visual data. Next, we discuss some of the challenges which raise open research questions in these topics.

1.3 Challenges

The human visual system has no difficulty to continually perceive the objects of interest and the surrounding environments. However, this is still a difficult task for machines when only monocular images are given. The machines have to extract meaningful geometric attributes (e.g., object regions in 2D images, spatial locations in 3D scenes) from a set of raw pixels captured by cameras. In this thesis, we focus on the aforementioned extraction task using deep models and study three important specific tasks: object visual tracking, object pose estimation and scene geometry understanding. The perception of geometric attributes in images is an active research field that addresses in particular the following challenges.

Tracking objects accurately and robustly. Despite the achievement of existing methods, the features of real-world videos still make object visual tracking a challenging task. Specifically, most object visual tracking benchmarks include videos featuring object appearance variations, occlusions, motion blur, illumination changes, etc. (see some examples in Figure 1.4). Despite the aforementioned challenges, an object visual tracking system is required to track the target accurately and robustly. Besides, the visual quality of videos is usually lower than the one of still images so that methods working well on still images can not be easily used on videos.

Estimating object 6D pose from a single image. The 6D pose of a rigid object can be decomposed into a 3D translation and a 3D rotation w.r.t. a reference coordinate system. Recent methods relying on distance information (e.g., depth maps captured by depth cameras) have achieved a good performance. However, methods using

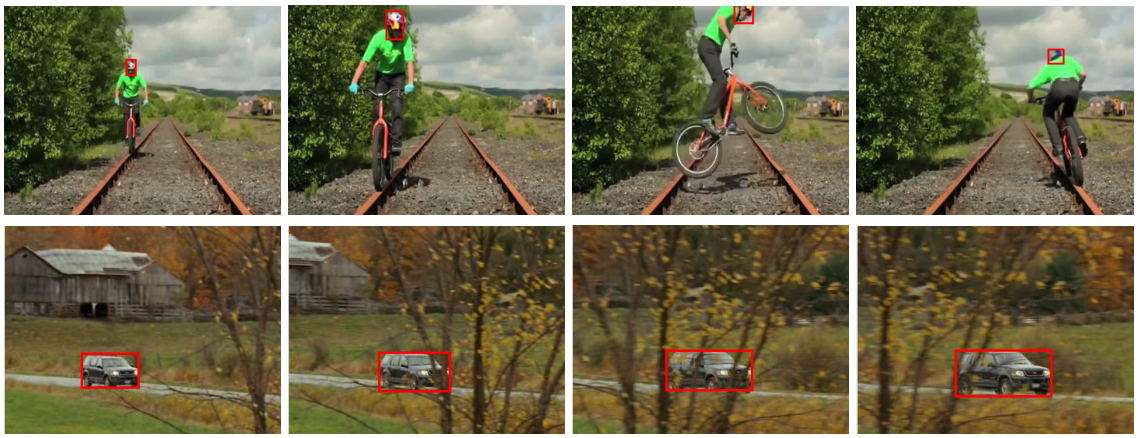


Figure 1.4: Some challenges in object visual tracking task. The first row illustrates object appearance and size variations while the second row illustrates the additional impact of occlusion. The red bounding boxes represent the tracking ground truths.

only color images still often fail due to the lack of 3D information as input. Meanwhile, the weakly-textured or non-textured object appearance as in Fig 1.2 degrades the performance of traditional methods based on detecting local features in images. Besides, most existing methods are tested with images captured in constrained experimental environments while methods working on images captured during daily life (cf. Figure 1.5) are less addressed.

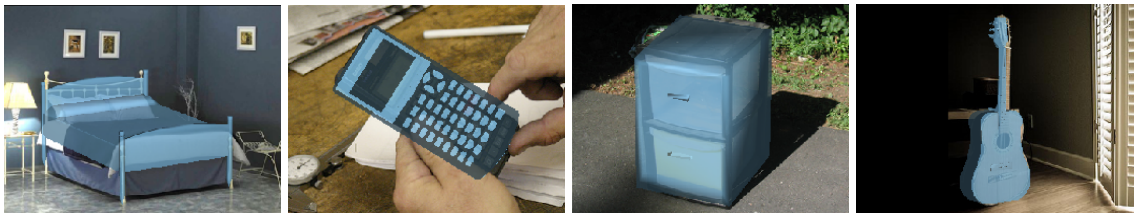


Figure 1.5: Examples illustrating the unconstrained backgrounds in everyday images for object pose estimation task.

Good formulation and data for occlusion reasoning. The low-level information about geometric occlusion in single images is useful for other scene geometry understanding tasks (e.g., depth estimation, surface normal estimation) and also helps discriminating different objects when there are inter-object occlusions between mul-

tiple objects. However, current occlusion representations are not extremely good for deep models to learn and inference. Besides, the existing occlusion annotations are either labeled manually with a great human labor or generated automatically with a lack of completeness (cf. Figure 1.6). The aforementioned challenges disturb the application of occlusion cues in downstream tasks although occlusions are ubiquitous in 2D images.

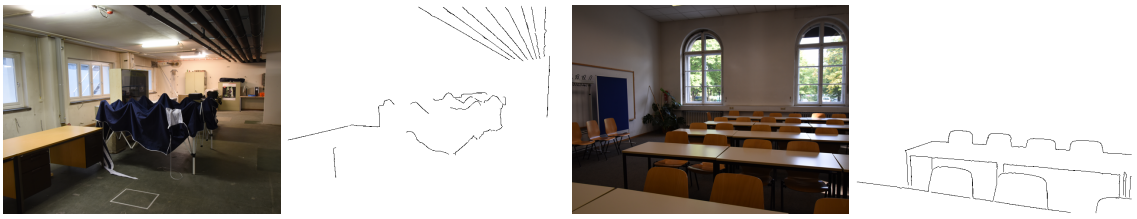


Figure 1.6: Examples illustrating incomplete geometric occlusion boundary annotations.

1.4 Thesis Structure and Contributions

We break our thesis into three main chapters that address each of the previously introduced challenges. In the remaining of this section, we describe the objective of each chapter, the proposed methods and the corresponding contributions.

Single object visual tacking. In chapter 2, we present our work about leveraging semantics information in object visual tracking framework. We first study systematically the popular tracking framework based on discriminative correlation filter. Then we propose to use semantics information estimated by semantic segmentation models in two tracking stages: the visual feature encoding stage and the target localization stage. We conclude that the involvement of semantics in the feature encoding stage improves the performance of both localization and scale estimation in the tracking framework. Our use of semantics in the tracking localization stage does not bring improvements in general, and we analyze the correlation between the tracking per-

formance and the semantic segmentation quality to explain this phenomenon.

Object pose estimation with object shapes. In chapter 3, we focus on using object shape information to improve the performance of object 6D pose estimation and do object pose refinement. Given an object 3D model and its 2D projection in an image, we propose to estimate the 2D coordinates of the projected object 3D surface points with deep models and to predict the object 6D pose using the estimated 3D-to-2D point correspondences. The proposed method benefits from the properties of these correspondences and achieves a better performance than other methods, in particular corner-based methods. In a second part, we study the constraints of existing object pose refinement methods and develop a pose refinement method for objects in the wild. Our experiments demonstrate that our models trained on either real data or generated synthetic data can refine pose estimates for objects in real images which are captured in daily life.

Geometric occlusion and depth map estimation. In chapter 4, we deal with geometric occlusion in single images. We first formalize a definition of geometric occlusion and propose a method to automatically generate high-quality occlusion annotations. We then express a new occlusion relationship formulation (we call it P2ORM) and propose a corresponding inference method. We show the relevance of the proposed occlusion relationship as a guidance signal for a depth map refinement method. According to evaluations on public benchmarks, both our occlusion estimation method and depth map refinement method achieve state-of-the-art performance. Last, we introduce our single-stage monocular depth estimation method which directly recovers better depth discontinuities with the help of P2ORM. Besides, we release three datasets containing the generated occlusion relationship annotations (i.e., InteriorNet-OR, NYUv2-OR and iBims-1-OR) to contribute to the advancement of research in the community.

In the following, we list our publications as well as the software and dataset releases that were performed during the course of this thesis.

1.4.1 Publications

The work done during this PhD led to the following publications:

- Qiu, X., Xiao, Y., Wang, C., Marlet, R.: Pixel-Pair occlusion relationship map (P2ORM): Formulation, Inference & Application. In: European Conference on Computer Vision (ECCV) (2020). [108] (Chapter 4)

- Xiao, Y., Qiu, X., Langlois, P., Aubry, M., Marlet, R.: Pose from shape: Deep pose estimation for arbitrary 3D objects. In: British Machine Vision Conference (BMVC) (2019). [152] (Chapter 3)

1.4.2 Software and dataset contributions

Software.

- P2ORM: The code and pretrained models for occlusion relationship estimation and depth map refinement are released. The code for high-quality occlusion annotation generation is also released as a part of the project presented in [108] (Chapter 4). https://github.com/tim885/P2ORM

- PoseFromShape: The code and pretrained models for object pose estimation are released as a part of the project presented in [152] (Chapter 3). https://github.com/YoungXIAO13/PoseFromShape

- BlockEstimation: The PyTorch [100] re-implementation for block pose estimation and robotic manipulation is released as a part of the project presented in [77]. https://github.com/tim885/blockEstimation

- BlockDimEstimation: The code and pretrained model of block 3D dimension estimation for robotic manipulation are released. https://github.com/tim885/blockDimEstimation

InteriorNet-OR dataset. We have publicly released the InteriorNet-OR dataset (http://imagine.enpc.fr/~qiux/P2ORM/) with the publication of [108] (Chapter 4). The synthetic RGB-D pairs come from the InteriorNet dataset [70] and the pixel-pair occlusion relationship annotations are generated by the method proposed in [108]. It contains 10K realistic images captured in 500 different indoor scenes and can serve as the training dataset for occlusion relationship estimation and depth map refinement on real images.

iBims-OR dataset. We have publicly released the iBims-OR dataset (http://imagine.enpc.fr/~qiux/P2ORM/) with the publication of [108] (Chapter 4). The real high-quality RGB-D pairs come from the iBims-1 dataset [58] and the pixel-pair occlusion relationship annotations are generated by the method proposed in [108]. It contains 100 real images captured in 20 different indoor scenes and can serve as an evaluation dataset for occlusion relationship estimation and monocular depth estimation.

NYUv2-OR dataset. We have publicly released the NYUv2-OR dataset (http://imagine.enpc.fr/~qiux/P2ORM/) with the publication of [108] (Chapter 4). Based on RGB-D pairs from the NYUv2 dataset [93] and occlusion boundaries offered by [111], the pixel-pair occlusion relationship annotations are generated by the method proposed in [108]. It contains 654 real images captured in 464 different indoor scenes and can serve as an evaluation dataset for occlusion relationship estimation and monocular depth estimation.

1.5 Outline

This thesis consists of five chapters including this introduction. In chapter 2, we present our work on leveraging structural semantics information in object visual tracking framework. In chapter 3, we focus on using object shape information to improve object 6D pose estimation performance and do object pose refinement for objects in the wild. In chapter 4, we address geometric occlusion in single images and propose methods both for annotation generation and inference. We also introduce our depth estimation methods using the proposed geometric occlusion formulation. Finally, we conclude our thesis in chapter 5 where we also present possible future work.

Chapter 2

Single Object Tracking with Structural Semantics

2.1 Introduction

In this chapter, we focus on (online) single object tracking, which is a very important topic in computer vision and has many practical applications such as video surveillance, robot navigation and autonomous driving. Only given the ground-truth bounding box of a target in the first frame of a video sequence, the tracker needs to successfully track the target in subsequent frames, by overcoming a series of challenges such as appearance variations, occlusions, motion blur and illumination changes.

The recent advances of deep learning community has led to tremendous progress in object tracking [139, 48, 82, 8, 129, 21, 18, 37, 67, 122, 9], in particular those discriminative methods, which utilize a classifier model to tell whether a target sample is the object of interest or not. Model training is performed by collecting positive and negative examples from the region of interest that is provided at the beginning of the tracking (e.g., examples in the first frame, possibly with other offline examples). The object localization in a frame is generally performed by looking for the candidate location with the highest classification score.

Among discriminative approaches for tracking, those based on Discriminative Correlation Filter (DCF) have proved both the effectiveness and the computational efficiency, thanks to Fast Fourier Transform (FFT). Given video frame(s) as input, the DCF-based approaches regress all the circular-shifted versions of input features (e.g., pixel-wise intensity) to a target Gaussian function with a small spatial bandwidth, while the maxima of response function indicates the target center position. Recently, feature extractors based on Convolutional Neural Networks (CNNs) have achieved a great success in many computer vision tasks (in particular object recognition) and also demonstrated a powerful generalization ability in other domains where they are not originally designed for. Meanwhile, convolutional features provide a type of powerful and robust visual features for various visual perception problems, and thus many DCF-based trackers using CNN features are proposed. However, these trackers often suffer from localization drifts, and we argue that one possible reason is that they lack structural semantics information of the target object in the image space. Inspired by recent advancements of semantic segmentation, we propose a DCF-based tracker which explicitly uses semantics information offered by deep models, so as to improve the precision of localization and also the robustness.

Meanwhile, the Siamese-network-based trackers have also received significant attentions, thanks to their well-balanced tracking accuracy and efficiency. These trackers formulate visual tracking as a cross-correlation problem and are expected to better leverage the merits of deep networks from end-to-end learning. In order to produce a similarity map from cross-correlation of the two branches, they train a Y-shaped neural network that combines the two branches, one for the target template and the other one for the candidate region. However, the state of the art of Siamese-network-based methods is lower than that of DCF-based methods.

To summarize, the main contribution of our work presented in this chapter is the inclusion of structural semantics information in the image space within the DCF-based tracking framework. The resulting tracker is demonstrated to be able to improve

both the accuracy and the robustness, based on the experiments on standard object tracking benchmarks.

The remainder of the chapter is structured as follows: We discuss related work in Section 2.2. We introduce DCF-based tracking framework in Section 2.3. We show how to explicitly use semantics information in Section 2.4. Implementation details and experimental evaluation are presented in Section 2.5 and Section 2.6, respectively. Finally, we make the conclusion in Section 2.7.

2.2 Related Work

DCF-based tracking framework. Starting from the proposition of MOSSE [10] in 2010, Discriminative Correlation Filter (DCF) based approaches have evolved quickly during these years with a continuous improvement of the performance on object tracking. MOSSE trains the filter by minimizing the total squared error between the actual and the desired correlation outputs on a set of grayscale sample patches. With the help of circular correlation and Fast Fourier transform, the computation can be done efficiently in the Fourier domain by point-wise operations. The initial DCF framework is later extended to use multi-channel feature maps (e.g., correlation for object alignment, multi-channel correlation filter, color attributes, HOG, Color Names,). Based on such a multi-channel DCF framework, the performance of trackers has been improved continuously by further extensions, including the exploitation of non-linear kernels cast as kernelized correlation filters (e.g., CSK [42], KCF [43]), the involvement of long-term memory components [49], part-based modeling [75], adaptive scale estimation [71], and the reduction of boundary effects caused by circular convolution [20], .

Recently, with the great achievement of Convolutional Neural Network (CNN) in many computer vision tasks such as object recognition (e.g., Alexnet [60], VGG [121], ResNet [39]) and semantic segmentation(e.g., FCN [78], DeepMask [105], Deeplab [16],

the strong generalization power of image representations produced by CNNs is also exploited by object tracking community. Danelljan et al. [19] involves CNN last convolutional layer feature in the DCF framework and achieves a significant performance improvement. Ma et al. [82] employs both shallow and deep layers features from CNN in a hierarchical ensemble of DCF trackers, so as to benefit from both low-level and high-level visual information. CCOT [21] proposes a continuous-domain formulation of the DCF framework with an integration of multi-resolution deep features and achieves a top performance on standard object tracking benchmarks [146, 59]. Based on CCOT, ECO [18] revisits the DCF core components and successfully achieves a speedup.

Semantic segmentation is a fundamental task in scene parsing and the goal is to assign a semantic class label to each pixel in the image. In the era of deep learning, state-of-the-art methods for semantic segmentation are mostly based on fully convolutional network (FCN), which only consists of convolutional layers. One important advantage of FCN architecture is that the network input images can have different spatial sizes. In order to enlarge the receptive fields of networks without downsampling by pooling layer, methods like [15, 157] propose to add dilated convolutional layers. Based on FCN, people also exploit the use of multi-scale feature ensembling to benefit from both the precise location information offered by low-level layer features and the semantics information offered by high-level layer features [119, 4]. Meanwhile, people try to apply structure prediction methods to refine segmentation predictions. Pioneer work [16] uses a Conditional Random Field (CRF) model to perform such a refinement, and following methods [106, 52] apply end-to-end modeling. Both of the two directions enhance the localization ability of models and therefore achieve better detection on image boundary regions. In our work, the FCN model proposed by [78] is adopted in the our tracking framework so as to exploit and fuse semantics information. Another relevant segmentation task is semi-supervised video object

segmentation where the object ground-truth mask in the video first frame is given. Recent methods mainly propose different ways to quickly adapt the parameters of a segmentation network during test time for the object segmentation in the video resting frames [104, 137, 87, 155, 98].

Tracking with semantics information. Compared to traditional methods for computer vision tasks, the superiority of deep models mainly owes to the ability of capturing implicit semantics information in images. Typical deep-learning-based recognition algorithms convert the output of CNN's last layer into a feature vector and do classification based on it. This is a reasonable choice as deep layers are more sensitive to category-level semantics information and more invariant to visual variations, such as illumination changes, color changes, deformations, . More concretely, the majority of state-of-the-art trackers [19, 82, 21, 18] regard deep features from CNN as additional input features which capture some implicit semantics information of both the tracked targets and the surrounding environment. However, the used deep features are not distributed spatially according to the target shape in the image space, and somehow limit a further improvement of the tracking performance. Whereas, deep segmentation models (e.g., FCN [78], DeepMask [105], Deeplab [16]) provide a type of tools to address this limitation. Hence, in our work, with a proper adoption of such segmentation models within the DCF-based tracking framework, we propose a new tracker exploiting a type of semantics features (named *structural semantics features*). This improves both the accuracy and the robustness of object tracking, as demonstrated by the experiments on standard object tracking benchmarks (e.g., OTB-2015 [146]).

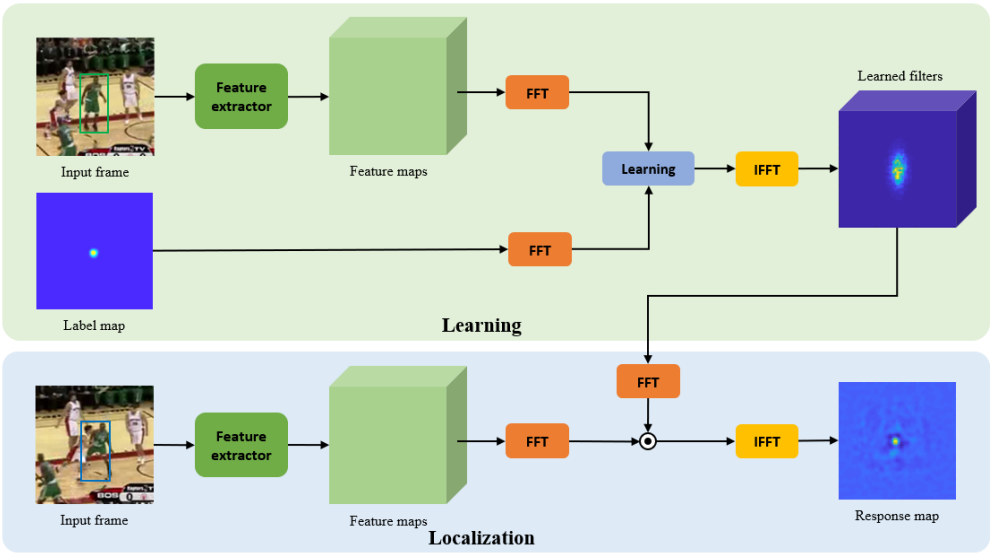


Figure 2.1: Overview of the DCF-based tracking framework. In the learning stage, the correlation filters are learned with the extracted feature maps and a ground-truth response map as input. In the localization stage, the estimated response map is obtained by the calculation between the new extracted feature maps and the correlation filters learned in the learning stage. Here, 'FFT' and 'IFFT' represent Fast Fourier Transformation and Inverse Fast Fourier Transformation respectively. The green bounding box is used as the ground-truth bounding box for learning while the blue bounding box is the estimated bounding box based on the estimated response map.

2.3 DCF-based Tracker

We adopt the DCF-based tracking framework as our tracking framework because of its accuracy and robustness. A typical DCF-based tracker learns a set of correlation filters $\{f_l\}_{l=1}^{L}$ in the first frame of the video sequence and update them during the tracking process. Given the feature maps $\{x_l\}_{l=1}^{L}$ extracted from the image patch in a new frame, a set of correlation response maps $\{S_l\}_{l=1}^{L}$ are obtained by the learned filters $\{f_l\}_{l=1}^{L}$ and the tracker estimates the translation and the scale change of the target object based on the obtained response maps $\{S_l\}_{l=1}^{L}$. The overview of the DCF-based tracking framework is illustrated in Figure 2.1, we introduce the details in Section 2.3.1 and Section 2.3.2.

2.3.1 Learning correlation filter

Given a video frame and the associated target bounding box (either manually provided or estimated), a DCF-based tracker will firstly crop an image patch I where the target bounding box is in the center and then resize the patch to a spatial size $M \times N$. Now we consider the case where each correlation filter f is learned from an extracted multi-channel feature map x of size $M \times N \times D$ (where D is the number of channels). We consider all the circular shifts of x along the two spatial dimensions as training samples to augment data. For each shifted sample $x(m, n)$, where $(m, n) \in \{0, 1, ..., M-1\} \times \{0, 1, ..., N-1\}$, it has a 2-D correlation response map label $y(m, n)$ following a Gaussian distribution: $y(m, n) = \mathrm{e}^{-\frac{(m-M/2)^2+(n-N/2)^2}{2\sigma^2}}$, where σ is the kernel width depending on the response map size (e.g., $0.1\sqrt{MN}$). Given feature map x, the correlation filter f of size $M \times N \times D$ can be learned by minimizing the following objective function:

$$E(f) = \sum_{m=1,n=1}^{M,N} \|f(m, n) \cdot x(m, n) - y(m, n)\|_{L^2}^2 + \lambda \sum_{d=1}^{D} \left\|f^d(m, n)\right\|_{L^2}^2 . \tag{2.1}$$

Here λ is a regularization parameter ($\lambda \geq 0$) and the inner product is induced by a linear kernel in the Hilbert space: $f(m, n) \cdot x(m, n) = \sum_{d=1}^{D} f^d(m, n) x^d(m, n)$. As stated in previous DCF-based methods, the minimization problem in Equation (2.1) can be solved in each individual feature map channel d using Fast Fourier Transformation (FFT). We use the capital letters to denote the corresponding Fourier transformed signals of variables introduced before. By finding stationary point as stated in [10], a closed-form solution is found for each channel of the correlation filter f:

$$F^d = \frac{Y \odot \bar{X}^d}{\sum_{i=1}^{D}(X^i \odot \bar{X}^i) + \lambda}, \tag{2.2}$$

where X^d, Y, F^d denote the Fourier-transformed signals of x^d, y, f^d, respectively, $\bar{X}^d$ complex conjugation of X^d, and the operator $\odot$ the Hadamard (element-wise) prod-

uct.

During the tracking, the appearance of target can often vary following the change of its rotation, scale, pose, the light conditions, (and also non-rigid deformation for non-rigid objects). Therefore, filters need to quickly adapt to target visual variance. A running average across video frames is therefore used to generate correlation filter F_j^d in the current frame j. Let us consider two consecutive frames $j, j-1$, and the running average of filter at current frame F_j^d is defined as:

$$A_j^d = \eta Y \odot \bar{X}_j^d + (1-\eta) A_{j-1}^d \tag{2.3a}$$

$$B_j^d = \eta \sum_{i=1}^{D} X_j^i \odot \bar{X}_j^i + (1-\eta) B_{j-1}^d \tag{2.3b}$$

$$F_j^d = \frac{A_j^d}{B_j^d + \lambda}\,. \tag{2.3c}$$

where η is a learning rate between consecutive frames and the temporal context is therefore involved in filter learning. After the initialization of correlation filters $\{f_l\}_{l=1}^{L}$ in the first frame, they are updated from the second frame $j = 2$ to the last frame $j = J$.

2.3.2 Target localization and scale estimation

Given a video frame and the target bounding box of the previous frame, the tracker crops an image patch which is centered at the bounding box and whose size is larger than the bounding box. The image patch is then resized to $M \times N$, and the target bounding box of the current frame needs to be estimated based on this image patch I. With the extracted feature map X_l of I and the learned correlation filters F_l of the previous frame, the corresponding correlation response map S_l of size $M \times N$ can be calculated as follows:

$$S_l = \mathcal{F}^{-1}\Big(\sum_{i=1}^{D} F_l^d \odot \bar{X}_l^d\Big)\,. \tag{2.4}$$

The operator $\mathcal{F}^{-1}$ denotes the inverse FFT transform and F_l^d denotes the d channel of filter F_l. The tracked target center location can then be estimated by searching for the position of the maximum value of the correlation response map S_l:

$$(\hat{m}, \hat{n}) = \arg\max_{(m,n)} S_l(m, n)\,. \tag{2.5}$$

Given a set of response maps $\{S_l\}_{l=1}^{L}$ relevant to feature maps $\{X_l\}_{l=1}^{L}$, we can compute the summation of all response maps and than determine the location via maximization, i.e.:

$$(\hat{m}, \hat{n}) = \arg\max_{(m,n)} \sum_{l=1}^{L} S_l(m, n)\,. \tag{2.6}$$

If DCF-based trackers use deep models with spatial pooling layer as feature extractor, the feature maps are originally CNN convolutional layers whose spatial size decreases with the increase of the network depth, and then upsampled to the image patch size, i.e., $M \times N$. Thus a response map based on shallow layers have a more fine-grained localization ability w.r.t deep layers, and a coarse-to-fine localization strategy can be adopted. Given two response maps S_{l-1}, S_l and the maximum value location $(\hat{m}_l, \hat{n}_l)$ on coarser response map S_l, the maximum value location on S_{l-1} is calculated as follows:

$$(\hat{m}_{l-1}, \hat{n}_{l-1}) = \arg\max_{(m,n)} S_{l-1}(m, n) + \gamma S_l(m, n) \;\; s.t. \;\; \|m - \hat{m}_l\| + \|n - \hat{n}_l\| \leq r\,. \tag{2.7}$$

The final response map for localization is a linear combination of S_{l-1}, S_l with a coefficient γ. Meanwhile, the constraint indicates that only the $r \times r$ neighboring regions of $(\hat{m}_l, \hat{n}_l)$ are considered in the maximization process. By using Equation (2.7), the final target location can be obtained by searching from the coarsest response map S_L to the finest response map S_1.

The tracker also needs to detect the size of target during tracking, a multi-scale detection strategy is used here. Given the target bounding box size $H_{j-1} \times W_{j-1}$

in the previous frame $j-1$, the size of the bounding box in the current frame j is estimated as: $\alpha_p H_{j-1} \times \alpha_p W_{j-1}$, where a scaling factor α_p should be determined. As described in [20, 43], the tracker will firstly crop image patches using multiple scaling factors (e.g., $\{\alpha_p\}_{p=1}^{P} \in \{1.02^{-2}, 1.02^{-1}, 1, 1.02^{1}, 1.02^{2}\}$) w.r.t. patch size in the previous frame and then resize them to the size of $M \times N$ for target detection. Given the response maps calculated from patches with different scales $\{S_p\}_{p=1}^{P}$ and the estimated target center location $\{(\hat{m}_p, \hat{n}_p)\}_{p=1}^{P}$, the estimated scaling factor index $\hat{p}$ is calculated as follows:

$$\hat{p} = \arg\max_{p} \{S_p(\hat{m}_p, \hat{n}_p)\}_{p=1}^{P} . \tag{2.8}$$

2.3.3 Baseline: CCOT

In this work, we adopt a recently proposed DCF-based tracker, CCOT [21], as our baseline. The main contribution of CCOT is the learning of convolution filters in a continuous domain instead of discrete domain as introduced Section 2.3.1. Here, we briefly describe the CCOT formulation. For clarity, we only present the one-dimensional domain formulation, please refer to [21] for the generalization in higher dimensions. The CCOT discriminatively learns a convolution filter f based on a collection of M feature maps $\{x_j\}_{j=1}^{M} \subset \mathcal{X}$ extracted from M video frames. Each feature map x_j contains D channels and each feature channel $x_j^d \in \mathbb{R}^{N_d}$ has an independent scalar N_d to indicate the dimension of x_j, whereas the feature map channels of previous DCF-based trackers usually share the same resolution N. Each channel of the feature map x^d is transferred from the discrete spatial domain $\{n\}_0^{N_d-1}$ to the continuous spatial domain $t \in [0, T)$ by introducing an interpolation model as follows:

$$J_d\{x^d\}(t) = \sum_{n=0}^{N_d-1} x^d[n] b_d \left(t - \frac{T}{N_d} n \right) . \tag{2.9}$$

Here, b_d is an interpolation kernel with period $T > 0$. The result $J_d\{x^d\}$ is thus an interpolated feature channel, viewed as a continuous T-periodic function. We use $J\{x\} = \{J\{x^d\}\}_{d=1}^{D}$ to denote the entire interpolated feature map.

In the CCOT formulation, a continuous T-periodic multi-channel convolution filter $f = (f^1 \dots f^D)$ is trained to predict the detection scores $S\{x\}(t)$ of the tracked target location as follows:

$$S\{x\} = f * J\{x\} = \sum_{d=1}^{D} f^d * J_d\{x^d\} \,. \tag{2.10}$$

The scores are defined in the continuous spatial domain $t \in [0, T)$. In Equation (2.10), the convolution between single-channel T-periodic functions $f(t), g(t)$ is defined as $f * g(t) = \frac{1}{T}\int_0^T f(t-\tau)g(\tau)\,\mathrm{d}\tau$. The multi-channel convolution $f * J\{x\}$ is obtained by summing the result across all the channels, as defined in Equation (2.10). The filters are learned by minimizing the following objective function:

$$E(f) = \sum_{j=1}^{M} \alpha_j \left\| S\{x_j\} - y_j \right\|_{L^2}^2 + \sum_{d=1}^{D} \left\| w f^d \right\|_{L^2}^2 \,. \tag{2.11}$$

The expected target detection scores $y_j(t)$ of sample x_j is set to a periodically repeated Gaussian function. The objective function consists of the weighted detection error, given by the L^2-norm $\|g\|_{L^2}^2 = \frac{1}{T}\int_0^T |g(t)|^2\,\mathrm{d}t$ and $\alpha_j \geq 0$, which is the weight of sample x_j. The regularization term integrates a spatial penalty $w(t)$ to mitigate the drawbacks of the periodic assumption, while enabling an extended spatial support [20].

As introduced in Section 2.3.1, a more tractable optimization problem is obtained by switching the objective function into the Fourier domain. Parseval's formula implies the equivalent loss as follows:

$$E(f) = \sum_{j=1}^{M} \alpha_j \left\| \widehat{S\{x_j\}} - \hat{y}_j \right\|_{\ell^2}^2 + \sum_{d=1}^{D} \left\| \hat{w} * \hat{f}^d \right\|_{\ell^2}^2 \,. \tag{2.12}$$

Here, we use $\hat{g}$ to denote the Fourier series coefficients $\hat{g}[k] = \frac{1}{T}\int_0^T g(t)e^{-i\frac{2\pi}{T}kt}\,\mathrm{d}t$ of a T-periodic function $g(t)$, and the ℓ^2-norm is defined by $\|\hat{g}\|_{\ell^2}^2 = \sum_{-\infty}^{\infty}|\hat{g}[k]|^2$. The Fourier coefficients of the detection score of one training sample x in Equation (2.10) is given by the formula $\widehat{S\{x\}} = \sum_{d=1}^{D}\hat{f}^d X^d \hat{b}_d$, where X^d is the Discrete Fourier Transform (DFT) of x^d. Assuming that each filter f^d has a finite number of non-zero Fourier coefficients $\{\hat{f}^d[k]\}_{k=-K_d}^{K_d}$, where $K_d = \lfloor \frac{N_d}{2} \rfloor$, Equation (2.12) then becomes a quadratic problem, which can be optimized by solving the following equation,

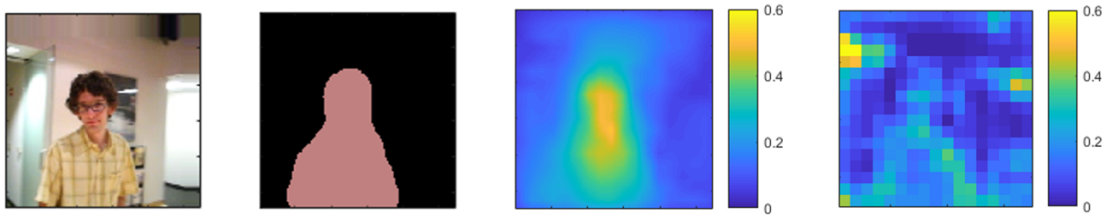
$$\left(A^{\mathrm{H}}\Gamma A + W^{\mathrm{H}}W\right)\hat{f} = A^{\mathrm{H}}\Gamma\hat{y}\,. \tag{2.13}$$

Here, the superscript ^{H} is used to denotes the conjugate-transpose of a matrix. $\hat{f}$ and $\hat{y}$ are vectorizations of the Fourier coefficients: $\hat{f} = [(\hat{f}^1)^T(\hat{f}^2)^T\cdots(\hat{f}^D)^T]^T, \hat{y} = [\hat{y}_1^T\hat{y}_2^T\cdots\hat{y}_M^T]^T$. The matrix A is of a sparse structure, with diagonal blocks containing elements of the form $X_j^d[k]\hat{b}_d[k]$. Further, Γ is a diagonal matrix of the weights $\alpha_1, \alpha_2, \ldots, \alpha_M$ and W is a convolution matrix with the kernel $\hat{w}[k]$. The CCOT [21] employs the Conjugate Gradient (CG) method [95] to iteratively approach the solution of (2.13). By doing so, a convolution filter f in the continuous domain can then be learned from a set of training feature maps $\{x_j\}_{j=1}^M$.

2.4 Tracking with Structural Semantics

2.4.1 Semantics representation for tracking

In this section, we present our approach of using semantic segmentation network outputs as additional features for object tracking. Such outputs offer pixel-wise semantics information about the target and the background, and may help target localization and scale estimation. The segmentation network that we use only consists of convolutional layers. Given the semantic segmentation network $F_{seg}(.)$ with its learned weights $W_{F_{seg}}$ and an image patch I of spatial size $M \times N$, the convolutional features



(a) (b) (c) (d)

Figure 2.2: Illustration of convolutional features from image segmentation models and the ones from image classification models: (a) Input image, (b) Semantic segmentation predicted by the segmentation model [78], (c) The average over channels of last layer features from the image segmentation model [78], (d) The average over channels of Conv-5 features from the image classification model [121].

of network layers are extracted as follows:

$$\{x_o\}_{o=1}^{O} = F_{seg}(I; W_{F_{seg}}) \,. \tag{2.14}$$

To differentiate from feature maps used in previously introduced DCF trackers, we use $\{x_o\}_{o=1}^{O}$ to denote the feature maps extracted by F_{seg}, where o presents the index of relevant convolutional layer. Similar to the features extracted by deep object recognition models [60, 121, 39], shallow layers of F_{seg} contains more fine-grained local features, while deep layers contain more semantic-aware information. In particular, the last layer feature x_O is of size $M \times N \times K$, where K is the number of semantic classes for semantic segmentation. Compared to deep features of image classification models, the main difference of x_O is that it contains semantics information which is more structural w.r.t. image pixel locations because F_{seg} is trained with pixel-level supervision. As illustrated in Figure 2.2, the convolutional features from image segmentation models are more sensitive to object spatial locations than the ones from image classification models. The semantic class index of one pixel at position (m, n) is estimated as follows:

$$\hat{k} = \arg\max_{k} x_O(m, n, k) \,. \tag{2.15}$$

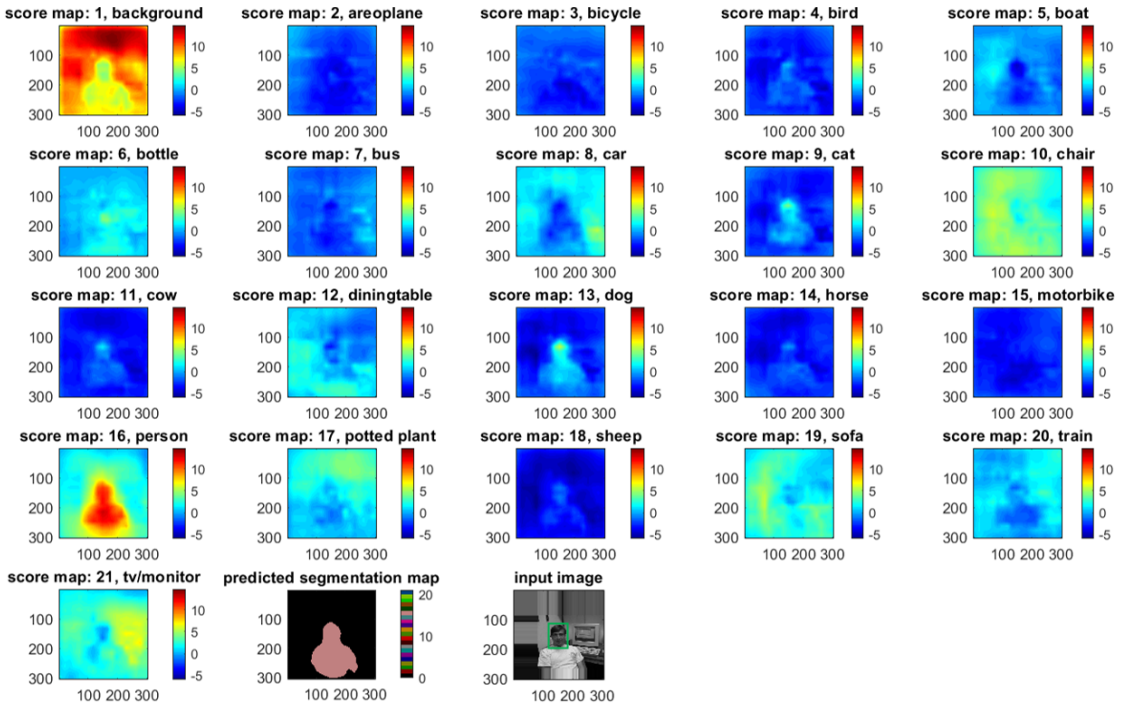


Figure 2.3: The channel-wise score maps of the F_{seg} last layer feature x_O. The green bounding box in the input image is the tracking ground truth.

The channels of x_O are shown in Figure 2.3 where F_{seg} is trained on PASCAL VOC 2011 dataset [28] with $K = 21$ in Figure 2.3. Each channel of x_O corresponds to a specific semantic class, and the value at one position indicates the possibility of being the corresponding class for the relevant pixel in the image space. For simplicity, we use x to represent x_O in the following paragraphs.

For a well-trained F_{seg}, $x(m, n, k)$ would be high, if and only if k is the ground-truth semantic class of a pixel (m, n). For the tracked target of semantic class k, the value of $x^k = x(\cdot, \cdot, k)$ would be high for those pixels within the target region, which demonstrates that x is an appropriate high-level feature (named as *structural semantics feature*) for tracking.

If the target semantic class k is given, we can directly use x^k as feature to train relevant correlation filter f and calculate relevant target response map S in DCF-based tracking framework. To validate, we evaluate one most common and challenging

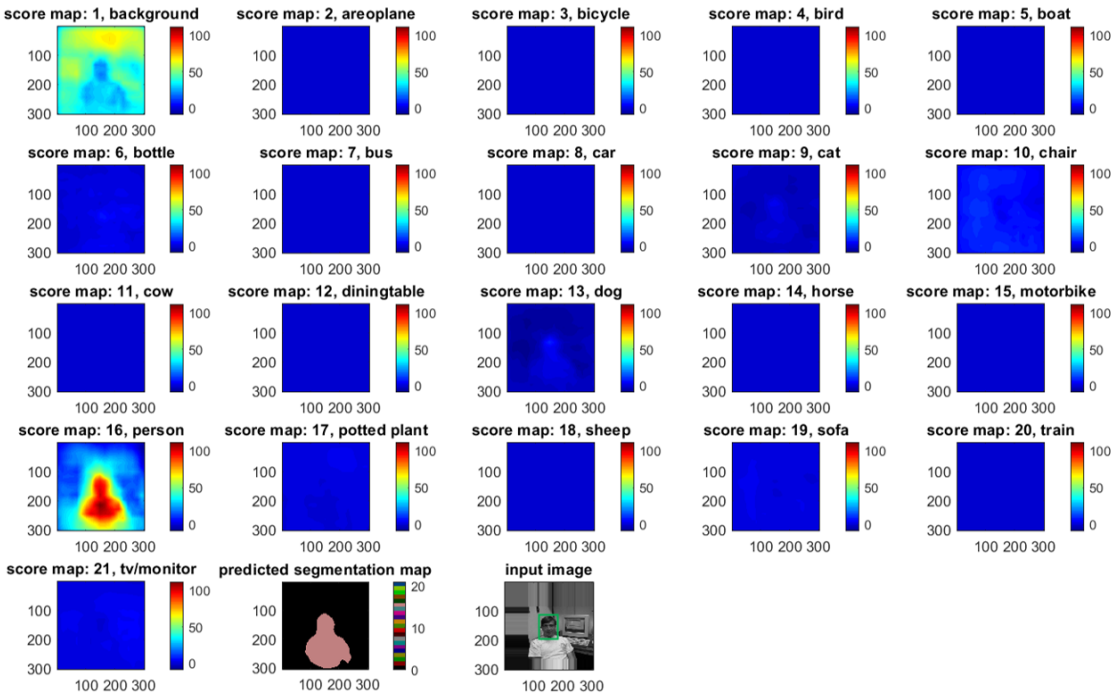


Figure 2.4: Scaled segmentation score maps as the input feature maps of the object tracking framework. The green bounding box in input image is the tracking ground truth.

target class, i.e., the human, on the tracking benchmark as stated in Section 2.6.3.

In a more general case, the target semantic class is not given and a single channel x^k cannot be chosen. An alternative way is to detect the target class in the first frame based on the segmentation feature maps x and choose relevant class channel. However, a more general way is to use all the channels of x as features in the DCF-based tracking framework. One issue of this method is that the feature channels which are not relevant to target class may cause much nuisance and degrade the effectiveness of the indication of the target region.

In order to enhance the importance of the channel of the target semantic class and reduce negative impact of the other channels, we propose a feature re-scaling mechanism for x. Concretely, for each class channel x^k, we compute its mean score $\bar{x}^k$ within the target bounding box $\mathcal{B}$ of size $M_{\mathcal{B}} \times N_{\mathcal{B}}$, and then a re-scaling factor

α_k by averaging all such mean scores:

$$\bar{x}^k = \frac{1}{M_{\mathcal{B}}, N_{\mathcal{B}}} \sum_{(m,n)\in\mathcal{B}} x^k(m,n) \tag{2.16a}$$

$$\alpha_k = \frac{\bar{x}^k}{\frac{1}{K}\sum_{k=1}^{K} \bar{x}^k} \tag{2.16b}$$

$$x_s^k = \alpha_k x^k \,, \tag{2.16c}$$

where x_s^k denotes the re-scaled feature map channel. Owing to this mechanism, for a good semantic segmentation network F_{seg}, the score values of the target class would be scaled up and the score values of the other classes would be scaled down, which enhances the effectiveness of the semantic-aware features. These scaled-down score maps make the tracker pay less attention on those non-target classes, therefore reduces drifts to distractors. The channels of x_s are illustrated in Figure 2.4.

After re-scaling, we fed x_s into the DCF-based tracking framework as features. As introduced in Section 2.3, given a set of feature maps $\{x_l\}_{l=1}^{L}$, a set of response maps $\{S_l\}_{l=1}^{L}$ are output by the learned feature-specific correlation filters $\{f_l\}_{l=1}^{L}$ during the localization phase. The estimated target location is calculated as in Equation (2.6) and the estimated scale factor for bounding box size is calculated as in Equation (2.8). In our experiments, the feature map set $\{x_l\}_{l=1}^{L}$ consists of the input image, the features of VGG network [121] Conv-5 layer and the re-scaled features of the FCN [78] last layer x_s. The whole method for the estimation of the localization response map is illustrated in Figure 2.5. VGG and FCN are pre-trained on the image recognition and semantic segmentation tasks, respectively, while their weights are fixed during the tracking for the whole sequence. Meanwhile, as introduced in Section 2.3, the weights of correlation filters $\{f_l\}_{l=1}^{L}$ are initialized in the first frame and continuously updated in the following frames.

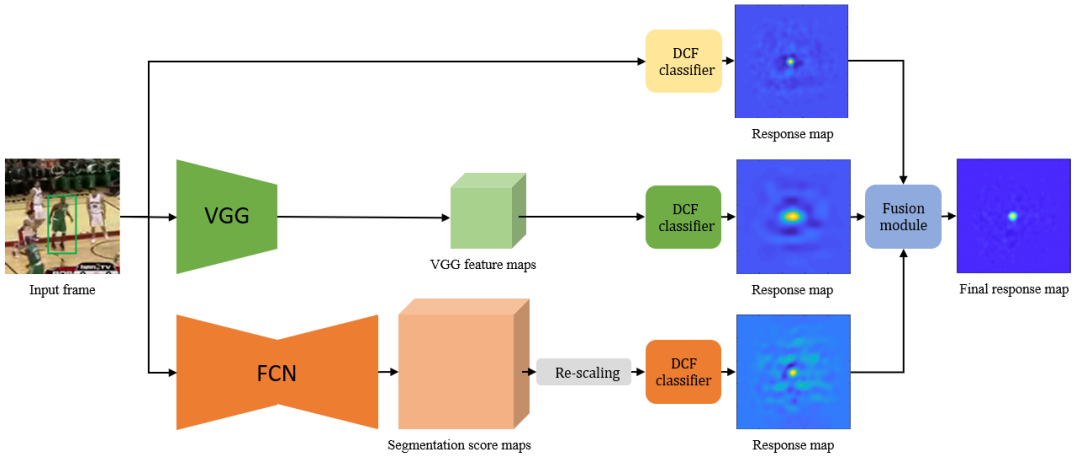


Figure 2.5: Overview of our method for target localization response map generation. The VGG network is trained for image classification task and extracts deep convolutional features. The FCN network is trained for image semantic segmentation task and predicts the segmentation score maps.

2.4.2 Module for generating target class probability distribution

In this section, we introduce how to generate the probability distribution of target class in images during tracking. We first use the output of the segmentation model last layer x (of size $M \times N \times K$) to estimate the target semantic class t in the first frame with the ground-truth bounding box of the target $\mathcal{B}$ (of size $M_{\mathcal{B}} \times N_{\mathcal{B}}$). We count the number of pixels belonging to different classes within $\mathcal{B}$ in the segmentation map $\hat{k}$ as the criterion,

$$\hat{k}(m,n) = \arg\max_{k} x(m,n,k) \tag{2.17a}$$

$$\hat{N}_k = \frac{1}{M_{\mathcal{B}} N_{\mathcal{B}}} \sum_{(m,n)\in\mathcal{B}} \mathbb{1}(\hat{k}(m,n) = k) \tag{2.17b}$$

$$\hat{t} = \arg\max_{k} \{\hat{N}_k\}_{k=1}^{K}, \tag{2.17c}$$

where the number of pixels belonging to $\hat{t}$ is the most among all classes $\{k\}_{k=1}^{K}$, thus $\hat{t}$ is the most possible semantic class of the target. If $\hat{N}_{\hat{t}} > 0.5$, we regard $\hat{t}$ as the target semantic class, otherwise we think the segmentation model output is not reliable enough in the given sequence and disable the methods proposed in this section and Section 2.4.3 for this sequence.

If the target semantic class is $\hat{t}$, we generate a pixel-wise probability distribution of the target class considering the fact that segmentation outputs are not always reliable. Based on x, the initial probability map of the target $\tilde{P}$ of size $M \times N$ is calculated as follows,

$$\hat{p}(m,n,i) = \frac{\mathrm{e}^{x(m,n,i)}}{\sum_{j=1}^{K} \mathrm{e}^{x(m,n,j)}} \tag{2.18a}$$

$$\tilde{P}(m,n) = \hat{p}(m,n,\hat{t})\,, \tag{2.18b}$$

where $\hat{p}$ (of size $M \times N \times K$) is regarded as the pixel-wise semantic class probability map. If the segmentation model performs well, the pixels within the target region will have high probabilities in $\tilde{P}$. Unfortunately, due to the domain gap between semantic segmentation datasets and object tracking datasets, the segmentation model may perform bad in some video sequences and result in unreliable target class probability maps. We therefore consider estimating the uncertainty of segmentation predictions by generating a pixel-wise confidence map $\hat{K}$ whose values are between 0 and 1. For one pixel location in the segmentation prediction $x(m,n,\cdot)$, the value of each channel and the relationship between the values of different channels both reflect in a degree the confidence of the segmentation model. Thus we further assume that $\hat{K}$ can be decomposed into $\hat{K}_a$ based on the absolute value of each channel and $\hat{K}_r$ based on the relationship between different channels. Without loss of generality, we use the channels $\hat{k}_1$ and $\hat{k}_2$ which have the highest value and the second highest value in

$x(m, n, \cdot)$ respectively to generate the confidence value $\hat{K}(m, n)$:

$$\hat{K}(m,n) = \hat{K}_a(m,n)\hat{K}_r(m,n) \tag{2.19a}$$

$$\hat{K}_a(m,n) = \frac{1}{1+\mathrm{e}^{-\theta_a(x(m,n,\hat{k}_1)-c_a)}} \tag{2.19b}$$

$$\hat{K}_r(m,n) = \frac{1}{1+\mathrm{e}^{-\theta_r(x(m,n,\hat{k}_1)-x(m,n,\hat{k}_2)-c_r)}}\,, \tag{2.19c}$$

where (θ_a, c_a) and (θ_r, c_r) are the parameters of two sigmoid functions respectively. In our experiments, we empirically set $(\theta_a, c_a) = (3, 1)$ and $(\theta_r, c_r) = (5, 0.25)$. Given the initial probability map $\tilde{P}$ and the segmentation confidence map $\hat{K}$, we generate the updated target class probability map $\hat{P}$. Here, a conservative policy is proposed to reduce the impact of the poor quality of some segmentation predictions during tracking. More concretely, we initially assume that all pixels within the image patch possibly belong to the target class; we believe the initial probability map $\tilde{P}(m, n)$ in function with the confidence score $K(m, n)$ as follows:

$$\hat{P}(m,n) = (1-\hat{K}(m,n)) + \hat{K}(m,n)\tilde{P}(m,n)\,. \tag{2.20}$$

According to Equation (2.20), when the segmentation model does not have enough confidence about its pixel-wise predictions, the relevant pixels can still have possibilities to be the pixels belonging to the target class.

Meanwhile, the estimated target class probability map $\hat{P}$ may have poor quality in some frames with extreme conditions such as background clutters or strong illumination variations. As there is only a little variation between consecutive frames in a video, temporal contexts between a small batch of video frames can also be used to improve the robustness of $\hat{P}$ while sacrificing a little bit accuracy:

$$\hat{P} = \sum_{i=1}^{N_f} w_i \hat{P}_{j-i+1}\,, \tag{2.21}$$

where P_{j-i+1} is the target class probability map (cf. Equation (2.20)) calculated in the frame $j-i+1$ and $\hat{P}$ is the final target class probability map in the current frame j considering temporal contexts. w_i and N_f are respectively the weight for the frame $j-i+1$ and the number of involved frames. In our experiments, we empirically set $N_f = 2$ and $\{w_i\}_{i=1}^{N_f} = \{0.75, 0.25\}$.

2.4.3 Module for weighting localization response map

The estimated target class probability map $\hat{P}$ introduced in Section 2.4.2 indicates the regions possibly belonging to the target class during target localization phase. In order to reduce tracklet drifts to the regions not belonging to the target class, we therefore use $\hat{P}$ to weight the localization response maps $\{S_l\}_{l=1}^{L}$ introduced in Equation (2.6). Considering the robustness of $\hat{P}$, we first generate the weighting map $\hat{W}_s$ of size $M \times N$ and then weight $\{S_l\}_{l=1}^{L}$ with $\hat{W}_s$:

$$\hat{W}_s(m,n) = \begin{cases} \hat{P}(m,n) & \text{if } \hat{P}(m,n) > \gamma_{low} \\ \gamma_{low} & \text{if } \hat{P}(m,n) \leq \gamma_{low} \end{cases} \tag{2.22}$$

$$S_w = \hat{W}_s \odot \sum_{l=1}^{L} S_l \,, \tag{2.23}$$

where γ_{low} is the lower bound of $\hat{W}_s$ to avoid too strong weighting (in our experiments, $\gamma_{low} = 0.5$) and S_w of size $M \times N$ is the final localization response map weighted by $\hat{W}_s$. S_w is then used for target location and scale estimation as described in Section 2.3.2.

2.5 Implementation Details

For the VGG network introduced in Section 2.4.1, we use the publicly available imagenet-vgg-m-2048 model [14] which is pretrained on ImageNet [22] for image classification. The fully connected layers of the VGG network are removed and the network

works as a feature maps extractor. For the FCN model introduced in Section 2.4.1 for semantic segmentation task, we use publicly available pascal-fcn16s-dag model [78] and train it on PASCAL VOC 2011 dataset [28] with segmentation ground truths. Our tracker is implemented in Matlab with open-source MatConvNet library [136]. We train and test our models on a single GeForce GTX TITAN X (12GB) GPU.

2.6 Experimental Evaluation

2.6.1 Datasets

OTB-2015. We evaluate our object tracking system on Online Object Tracking Benchmark (OTB-2015) [146], which is a standard object tracking benchmark commonly used in the tracking community. It contains 100 challenging sequences with different difficulties such as appearance variations, occlusions, motion blur, illumination changes, etc. Moreover, the authors of [146] select 50 most challenging sequences in OTB-2015 (named TB-50), on which we evaluate the proposed method and compare with 9 state-of-the-art trackers: CCOT [21], CF2 [82], DeepSRDCF [19], MEEM [160], Staple [7], SRDCF [20], LCT [83], SAMF [71] and DCFNet [142]. Among TB-50 sequences, the targets of 33 sequences are human beings, we make these 33 sequences make up a new subset (called TB-50-Human). Meanwhile, as the semantic classes of tracking targets in TB-50 are partially shared with the ones of the semantic segmentation dataset in our experiments (i.e., PASCAL VOC 2011 [28]), we also create a new subset (called TB-50-Included) which consists of 44 sequences satisfying the condition.

PASCAL VOC 2011. We train our semantic segmentation model on the training split of PASCAL VOC 2011 semantic segmentation subset [28]. The subset contains 2913 labeled images and 21 object semantic classes.

2.6.2 Evaluation metrics

One Pass Evaluation (OPE). We choose the OPE metric where the tracking errors are measured from the first frame to the last frame of each sequence. The reason of using the OPE metric is that it reflects the practical situation when trackers are used in online object tracking applications.

Precision plots measure the estimated target center location errors. Here, the precision score of a sequence is defined as the percentage of sequence frames whose estimated locations are within the given 2D distance threshold of the ground-truth locations. The precision plot of a sequence shows the precision scores using distance thresholds varying from 0 pixels to 50 pixels and the precision plot of each tracker is an average over the precision plots of all sequences. For the comparison between trackers, we use the precision score using the distance threshold of 20 pixels as the representative precision score of each tracker.

Success plots measure the overlaps between the estimated target bounding boxes and the ground-truth bounding boxes. The overlap between two bounding boxes is defined as Intersection over Union (IoU) and the success score of a sequence is the percentage of sequence frames whose IoUs are larger than the given IoU threshold. The success plot of a sequence shows the success scores using IoU thresholds varying from 0 to 1 and the success plot of each tracker is an average over the success plots of all sequences. For the comparison between trackers, the Area Under Curve (AUC) of each success plot is used to rank trackers. Compared to precision plots, success plots also measure the target scale estimation errors.

2.6.3 Tracking with structural semantics representation

In this section, we asses the performance of the proposed method using structural semantics features as described in Section 2.4.1. For the case where the target semantic

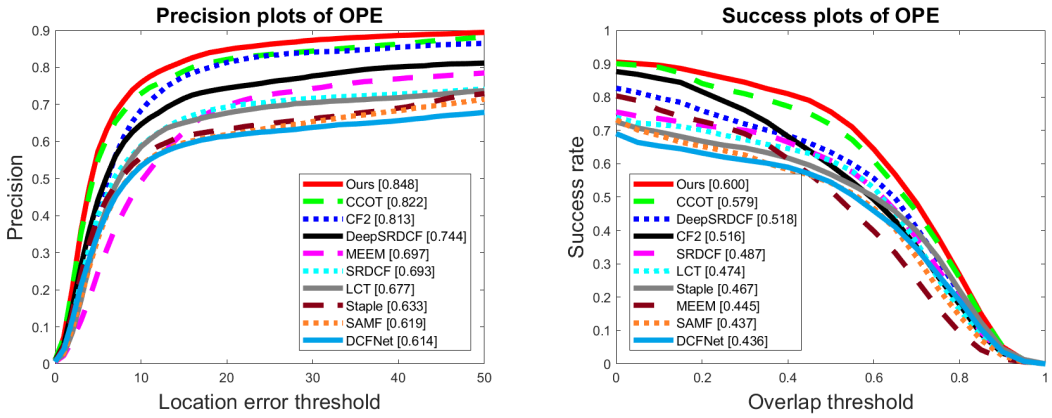


Figure 2.6: Precision plots and success plots of our method and other methods on TB-50-Human sequences whose targets are persons.

class is known, we perform the evaluation on the 33 sequences of TB-50-Human whose targets are persons, since humans are one most common and challenging tracking target. The precision plots and success plots are shown in Figure 2.6. Compared to our baseline CCOT tracker [21] and other state-of-the-art trackers, the results demonstrate that the inclusion of single class structural semantics features indeed improves the accuracy of target localization and of target size estimation.

When the target semantic class is not given, we evaluate our method on the 44 sequences of TB-50-Included whose targets semantic classes are included in PASCAL VOC 2011. As shown in Figure 2.7, the inclusion of all classes structural semantics features with our feature re-scaling mechanism improves our baseline performance although targets classes are not known. We further offer an attribute-based evaluation as in Figure 2.8, the proposed method gets a significant performance boost on sequences with background clutter, illumination variation, in-plane rotation, low resolution, out-of-plane rotation and out-of-view. The attribute-based evaluation also demonstrates that the inclusion of structural semantics information in image space alleviates the performance degradation of state-of-the-art trackers caused by the different nuisances. We present some qualitative results of our method in Figure 2.9.

We also evaluate the proposed method on the 50 sequences of TB-50, despite

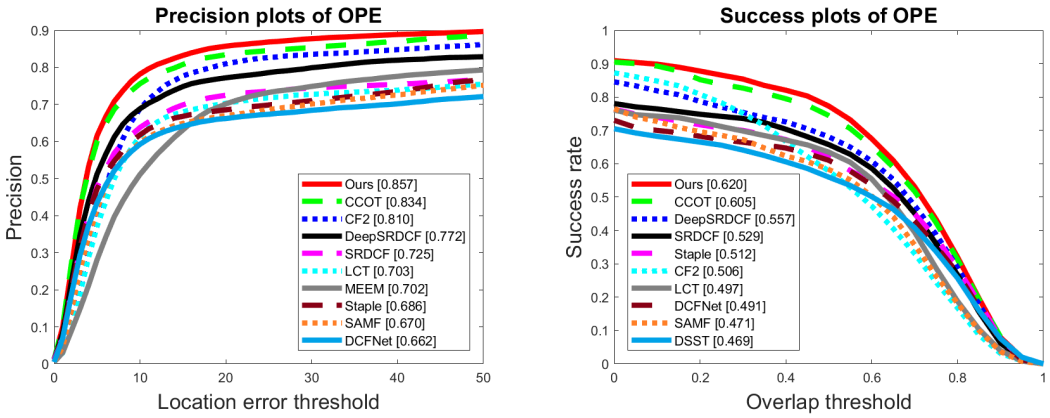


Figure 2.7: Precision plots and success plots of our method and other methods on TB-50-Included sequences whose targets semantic classes are included in PASCAL VOC 2011.

the fact that the target semantic classes of six sequences are not seen during the training of our semantic segmentation model. For these unseen targets, our semantic segmentation model cannot offer explicit semantics features for tracking. As shown in Fig 2.10, although the improvement is reduced w.r.t. the one on TB-50-Included, a performance boost can still be observed.

To better understand the contributions of the inclusion of structural semantics information and of the feature re-scaling mechanism, we performs an additional experiment to compare the performance with v.s. without feature re-scaling mechanism (i.e., ours (w/ feat-scaling) v.s. ours (w/o feat-scaling) in Figure 2.11). As demonstrated by the results in Figure 2.11, both of these two components can boost the performance of the baseline tracker [21].

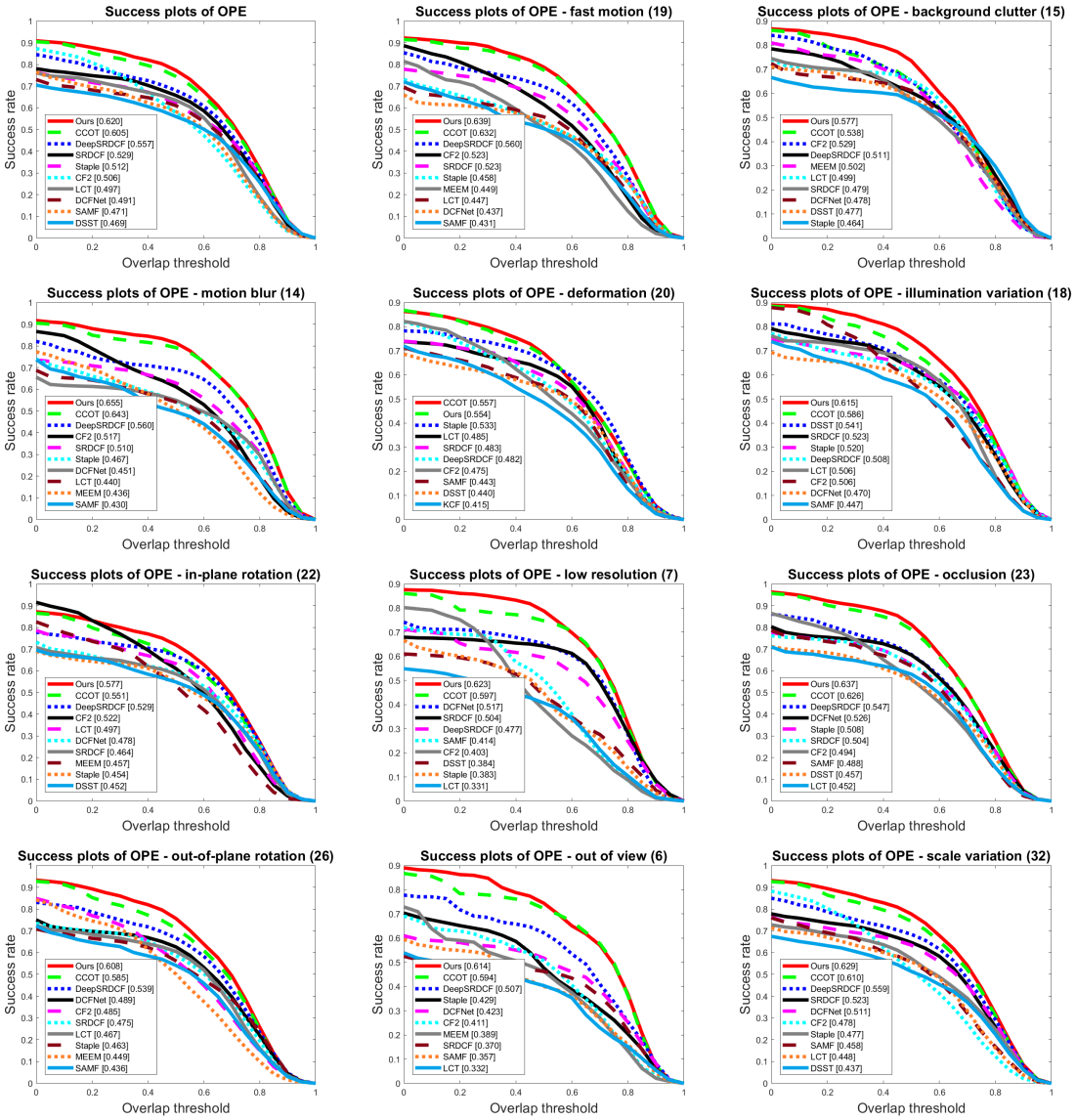


Figure 2.8: Success plots of our method and other methods on TB-50-Included sequences whose target semantic classes are included in PASCAL VOC 2011. The total success plot (top-left) is displayed along with the plots for all 11 attributes. The title of each plot indicates the name of the attribute and the number of sequences associated with it.

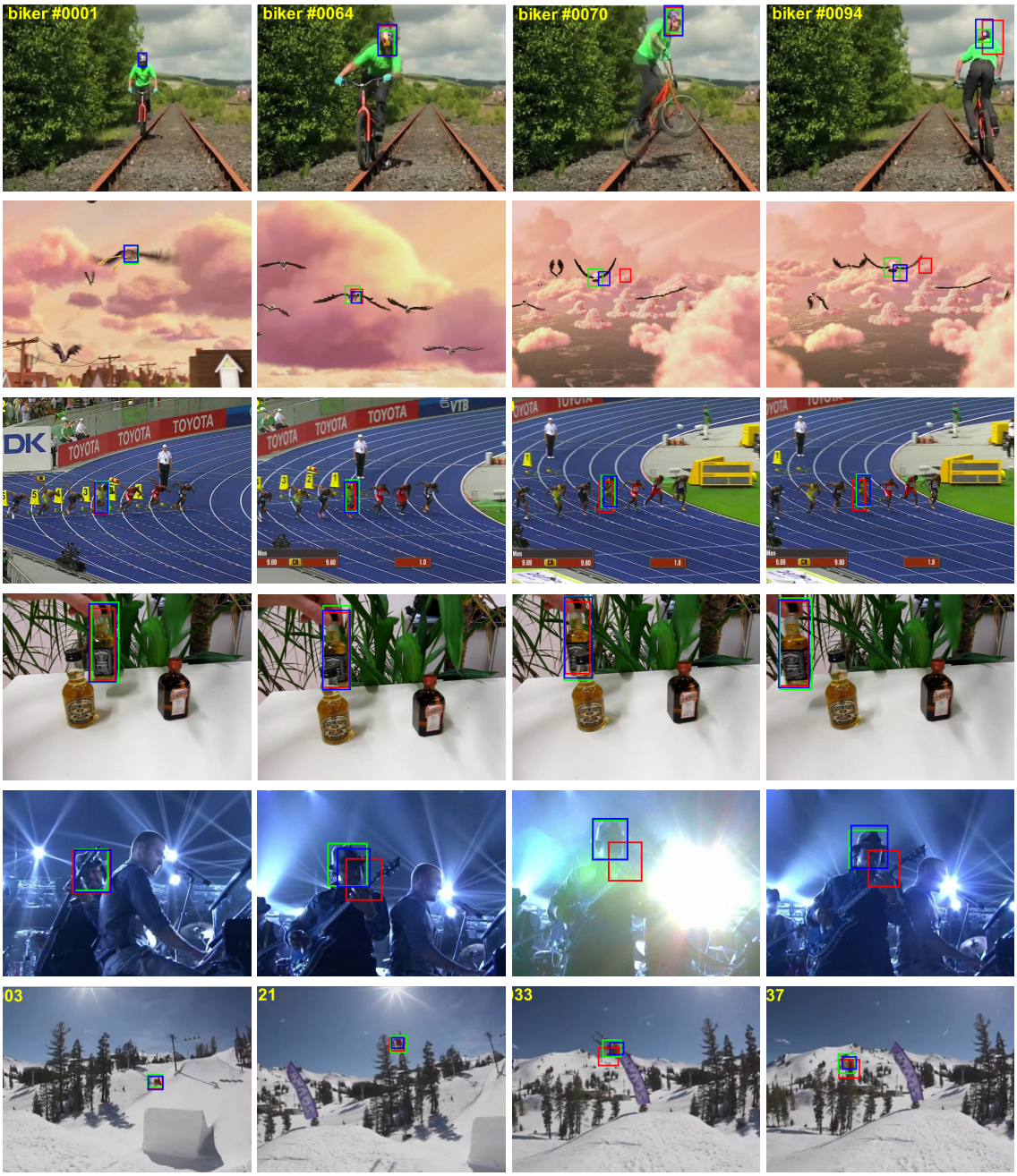


Figure 2.9: Qualitative results of our method and our baseline method CCOT [21] on TB-50 sequences. **green**: the ground-truth bounding box; **red**: the bounding box predicted by CCOT [21]; **blue**: the bounding box predicted by our method.

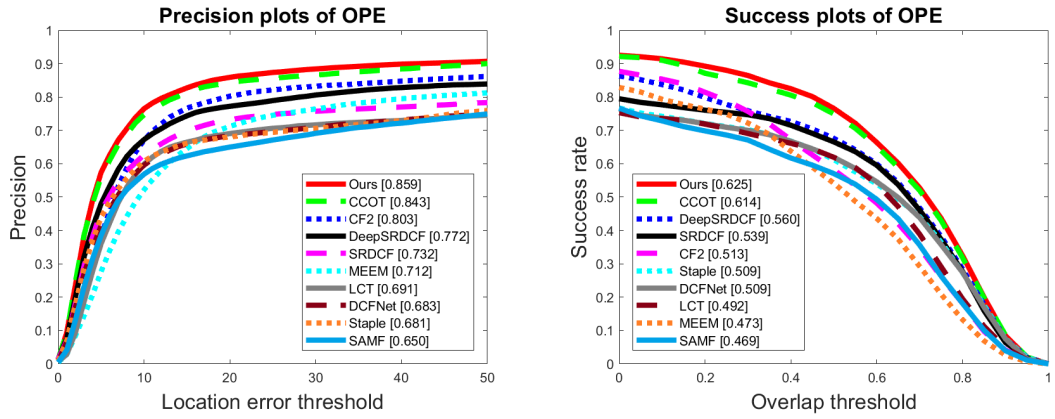


Figure 2.10: Precision plots and success plots of our method and other methods on TB-50 sequences.

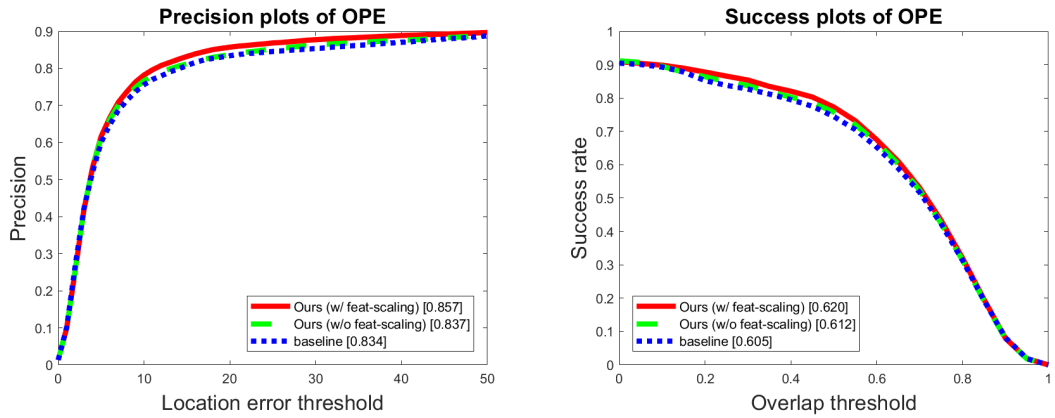


Figure 2.11: Precision plots and success plots of our method and baseline method CCOT [21] on TB-50-Included sequences whose target semantic classes are included in PASCAL VOC 2011. *ours (w/ feat-scaling)* and *ours (w/o feat-scaling)* refer to the fact that feature re-scaling mechanism is used or not

2.6.4 Tracking with weighted localization response map

In this section, we evaluate our method which weights the target localization response map with the generated target class probability distribution. As the proposed method introduced in Section 2.4.3 is conditioned on the assumption that the target semantic classes are seen during the training of our semantic segmentation model, we therefore evaluate our method on the 44 sequences of TB-50-Included whose target semantic classes are included in PASCAL VOC 2011 [28]. As shown in Figure 2.12, the proposed method is equivalent in performance w.r.t the baseline method CCOT [21]. By comparing the performance on each sequence w.r.t. our baseline, our method improves the performances on 10 sequences and degrades the performance on 9 sequences, while the performances on the other 25 sequences are almost unchanged. Based on a sequence-wise analysis, we think that the improvement/degradation phenomenon correlates to the semantic segmentation quality during tracking. Here, we visualize the output of our semantic segmentation model for the sequences with performance boost and the ones with performance degradation. As shown in Figure 2.13, when the segmentation model performs well, some maximums on the initial localization response map S can be filtered out by the proposed weighting map W_s in Equation (2.23), and the robustness of tracker is improved. On the contrary, if the input image is too difficult for the semantic segmentation model (e.g., extreme illumination condition), the quality of the generated weighting map W_s is bad, which can degrade the performance of the final target localization map S_I and consequently degrade the tracking performance. These results show that the proposed method is highly conditioned on the quality of segmentation and the domain gap between segmentation datasets and tracking datasets is in fact very big. Methods which alleviate the domain gap are worth future research.

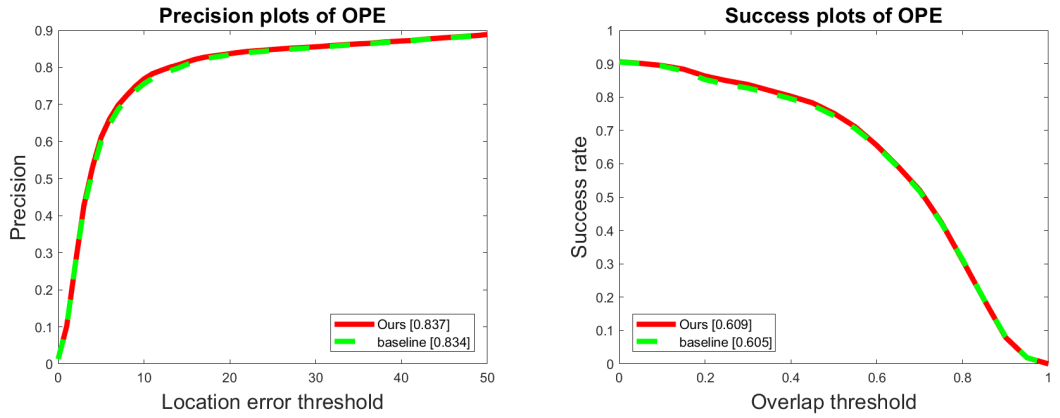
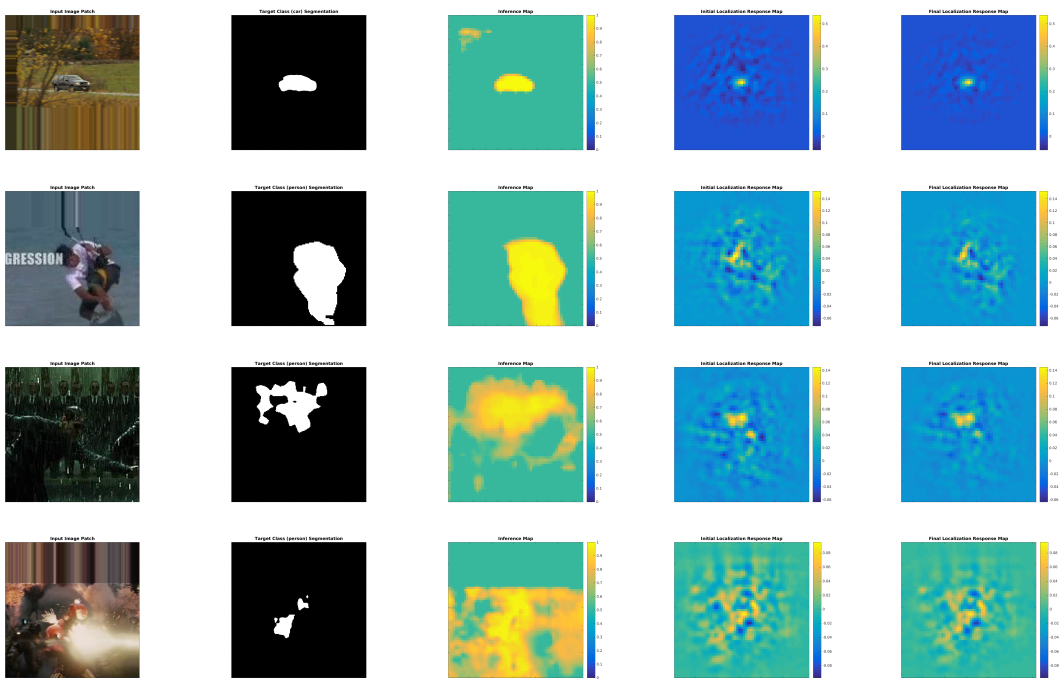


Figure 2.12: Precision plots and success plots of our method and the baseline method CCOT [21] on TB-50-Included sequences whose target semantic classes are included in PASCAL VOC 2011 [28]



(a) (b) (c) (d) (e)

Figure 2.13: Visualizations of the proposed method components. The first two rows show two examples where the segmentation model performs well, while the last two rows show two counter-examples: (a) RGB input images, (b) Binary segmentation about target semantic class, (c) Weighting map W_s introduced in Equation (2.23), (d) Initial target localization response map S, (e) Final target localization response map S_I.

2.7 Conclusion

In this chapter, we first systematically summarize the DCF-based tracking framework and then propose to include structural semantics information within it to enhance the tracking performance. This information can either serve as an appearance-invariant representation for tracking or an indication during the target detection phase. The experimental results demonstrate that the involvement of structural semantics representation can help to improve both the localization precision and the robustness of the DCF-based tracking framework. Meanwhile, using structural semantics in the target detection phase is highly conditioned on the quality of estimated semantics and is worth future research.

Chapter 3

Object Pose Estimation with Object Shapes

3.1 Introduction

In this chapter, we study the task of estimating the 6D pose of an rigid object in a singe view with the object 3D shape. Concretely, given an object image and the object 3D shape (e.g., object CAD model), we want to estimate the 3D translation and the 3D rotation of the object frame w.r.t. a reference frame (e.g., camera frame). The object image is the 2D projection of the object 3D shape (with a texture map) in a scene-specific lighting, while the object 3D model describes the object shape, the linked object frame, and optionally the object surface appearance. Recently, 6D pose estimation of object instances has become more and more popular because of its applications in robotics, virtual reality and augmented reality. Here we propose an approach which takes advantage of 3D-to-2D point correspondences between the object 3D shape and its 2D projections in images to improve the performance of our pose estimator. Given initial object pose estimates, people also develop pose refinement methods profiting from the object shape information to improve the accuracy of pose estimates. To reduce the constraints of applying object pose refinement for daily

life images, we propose a category-agnostic object pose refinement method which can refine pose estimates for objects in the wild.

Traditionally, people use either feature-matching methods or template-matching methods to estimate the pose of an object given its 3D shape. Feature-matching methods first extract 2D local features (e.g., SIFT [80], SURF [6], ORB [120]) from the object image and match them to the given object 3D shape. Then a PnP algorithm is applied to recover the 6D pose based on the estimated 3D-to-2D point correspondences. One main default of these methods is that local feature detectors usually do not work well when the object appearance is poorly-textured. On the other hand, template-matching methods [79, 66, 45, 46] match directly the observed object image to one of the stored object templates (e.g., the rendered objects using the object 3D models) which are relevant to specific object poses. Although this strategy can solve the poorly-textured appearance problem, the performance of template-matching methods is often affected greatly by occlusions and truncation in the observed object images. In the era of deep learning, more and more methods train deep models to estimate the 2D projections associated to the object 3D shape in images (e.g., the 2D projections of the object 3D bounding box corners), and then recover the object pose with some PnP algorithm [109, 130, 96, 35]. This trend can be regarded as a renaissance of feature-matching methods as deep models can estimate more stable 2D features (e.g., 2D keypoints) for the poorly-textured objects owing to their large receptive fields and implicit understanding of semantics. Although these methods achieve the state-of-the-art performance, the accuracy of estimating the bounding box corners is not always ideal and the low accuracy often results in bad pose estimates. We argue that this phenomenon is possibly due to the fact that some bounding box corner projections are very far away from the object in the image and it is therefore very hard for deep models to learn the mapping from the observed image to the corresponding 2D projections. To this end, we propose a object pose estimator which directly predicts the 2D projections of the object 3D surface points

instead of the bounding box corners to improve the accuracy of the 2D keypoints regression and recover more accurate pose estimates.

Given an initial pose estimate and an object 3D shape, the accuracy of the pose estimate can be further improved by a pose refinement method which compares the rendered object image using the initial pose against the observed object image, and then estimates the pose update. Based on ICP, [54] compares the estimated image contours using the pose estimate against the observed ones to refine the pose estimate. Deep refinement methods use deep models to predict 2D projection corrections [109] or a pose update [85, 72]. While the aforementioned methods work well on synthetic datasets or real datasets [46, 150] whose images have very constrained backgrounds (e.g., a table), object pose refinement for objects in the wild is usually less addressed. The main challenges include varying image sizes, unconstrained backgrounds, image-wise camera intrinsics and diverse object shapes. In other words, the object images are captured during daily life by different people without any experimental setup. Inspired by the refinement methods using deep models [109, 86, 72], we propose a pose refinement method conditioned on the object instance shapes to overcome the aforementioned challenges. The proposed refinement method is category-agnostic and conditions on the instance-wise object 3D shape to predict the relative pose transformation. Our experiments demonstrate that the proposed method can refine the pose estimates for objects in the wild. Moreover, the generalization ability of our refinement method is illustrated by evaluating on object categories which are not seen during the training of our models.

To summarize, the contributions of our work presented in this chapter, are as follows:

- We propose a monocular object pose estimator which predicts the 2D projections of the object 3D surface points to reduce 2D estimation errors and recover more accurate pose estimates.

- We propose an object pose refinement method which can refine initial pose estimates from images in the wild while existing methods work on images with constrained backgrounds.
- We evaluate the proposed pose estimator on LINEMOD [46] and the proposed pose refinement method on ObjectNet3D [148]. Our experiments show that the proposed pose estimator achieves improvements over state-of-the-art methods and the proposed refinement method can refine initial pose estimates for objects in the wild, even for the objects which are not seen during the training of our models.

The remainder of the chapter is structured as follows: We discuss related work in Section 3.2. We introduce the proposed monocular object pose estimator in Section 3.3. We discuss how to refine the coarse object pose estimates for objects in the wild in Section 3.4. Experimental evaluations are presented in Section 3.5. We make our conclusion in Section 3.6.

3.2 Related Work

In this section, we discuss pose estimation of a rigid object from a single RGB image, first in the case where the shape of the object is known, then when the shape is unknown.

Pose estimation explicitly using object shape. Given an object image and the object 3D shape, traditional methods for monocular object pose estimation can be roughly divided into feature-matching methods and template-matching methods. The object shape representation has many variants such as a 3D mesh, voxels, a point cloud or synthetic rendered images based on the object shape. Feature-matching methods try to extract local features from the image, match them to the given object 3D model and then use a PnP algorithm to recover the 6D pose based on the estimated

3D-to-2D point correspondences. Increasingly robust local feature descriptors [80, 6, 133, 135, 101] and more effective PnP algorithms [64, 162, 69, 29] have been used in this type of pipeline. Although performing well on textured objects, these methods usually struggle with poorly-textured objects. To deal with this type of objects, template-matching methods try to match the observed object to a stored pose-specific template [66, 79, 45, 46]. However, they usually perform badly in the case of partial occlusion or image truncation. Apart from the aforementioned two types of pipeline, it is also possible to try to directly predict the 3D coordinates of the object model vertices for each pixel and infer the object pose [12].

More recently, deep models have been trained for the object pose estimation from an image of a known or estimated object 3D model. Most methods estimate the 2D projections in the test image of the object 3D bounding box [109, 130, 96, 35] or category-agnostic semantic keypoints [101, 33] to find 2D-to-3D point correspondences and then apply a PnP algorithm just like feature-matching methods.

Pose refinement with object shape. Once a coarse pose has been estimated, the given object shape also enables a pose refinement method to refine the coarse pose estimate. Based on ICP, [54] compares the estimated image contours using the pose estimate against the observed ones to refine the pose estimate. Deep refinement methods use deep models to predict 2D projection corrections [109] or a pose update [85, 72] by matching the rendered object image using the current pose estimate against the input object image.

Pose estimation not explicitly using object shape. In recent years, with the release of large-scale datasets [32, 46, 149, 148, 127], data-driven learning methods (on real and/or synthetic data) have been introduced which do not rely on an explicit knowledge of the object 3D models. Learning-based approaches only using object images without object shape information become possible and have proved its effec-

tiveness. Conventionally, the model is trained on supervised data to directly map the observed object appearance to the object pose w.r.t. a reference frame. These methods can be roughly separated into methods that estimate the pose of any object of a training category and methods that focus on a single object or scene. For category-wise pose estimation, people assume that all objects within the same category have the same canonical view which describes how the object appearance is linked with the object frame. As the category-specific canonical view is not given during test, they have to be learned implicitly by models during training and these models are thus category-specific. The prediction can be cast as a regression problem [99, 102, 89], a classification problem [135, 125, 27] or a combination of both [92, 36, 65, 84]. Besides, Zhou et al. manually label category-agnostic object 3D keypoints and regress their 2D projections in images to recover the object pose [163]. It is also possible to estimate the pose of a camera w.r.t. a single object 3D model but without actually using the 3D model information. In fact, many recent works have applied this strategy to recover the full 6-DoF pose for object [132, 92, 54, 150, 65] and do camera re-localization in the scene [56, 55].

3.3 Pose Estimation with Rich 3D-to-2D Point Correspondences

Given the image of a rigid object captured by a camera which is linked with its coordinate system, the 6D pose of the object frame w.r.t the camera frame is defined with a rotation matrix $\mathbf{R}$ and a translation vector $\mathbf{t}$ that transform the object from its local coordinate system to the camera coordinate system. The rotation $\mathbf{R}$ and the translation $\mathbf{t}$ of the object both have 3 degrees-of-freedom (DoF), therefore the rigid object pose estimation problem is referred to as object 6D pose estimation task. If we regard the object frame as the reference frame, this task can also be interpreted as

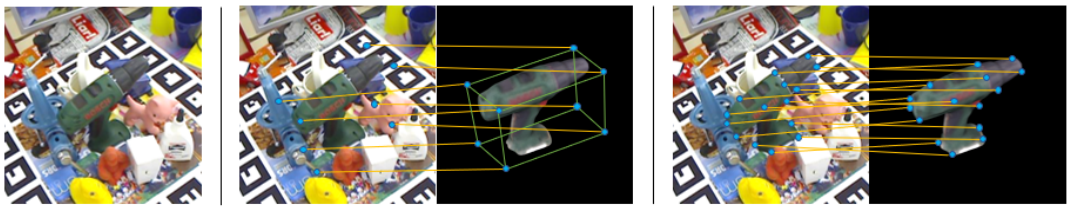
a camera extrinsic calibration. Meanwhile, object shape representations (e.g., mesh, point cloud, voxel, surface textures, etc.) describe both the appearance of object and the linked object frame. Given the object 3D model, the object appearance in an 2D image is in fact the 2D projection of its shape and surface textures with the relevant object 6D pose and a scene-specific lighting. By using the 3D-to-2D correspondences between the 2D projection and the relevant object 3D shape information, the object 6D pose can be recovered. Thus developing methods which predict accurate 3D-to-2D correspondences is one main-stream direction for object 6D pose estimation task.

If we represent the object shape sparsely with n 3D control points, we can formulate the 6D pose estimation problem in terms of predicting the image 2D coordinates of object 3D control points. The 2D coordinate vectors $\{\mathbf{p}_i\}_{i=1}^n$ in the 2D image coordinate system and their corresponding 3D coordinate vectors $\{\mathbf{P}_i\}_{i=1}^n$ in the object coordinate system are related by $\mathbf{R}$, $\mathbf{t}$ with the perspective projection model of pinhole camera:

$$s_i \bar{\mathbf{p}}_i = \mathbf{K}[\mathbf{R}|\mathbf{t}]\bar{\mathbf{P}}_i \tag{3.1}$$

where $\bar{\mathbf{P}}_i = [x\, y\, z\, 1]^T$ and $\bar{\mathbf{p}}_i = [u\, v\, 1]^T$ are represented in homogeneous coordinates. $\mathbf{K}$ is the camera intrinsic matrix and s_i is the z-axis coordinate of the point $\mathbf{P}_i$ in the camera frame. Given these 3D-to-2D point correspondences, a Perspective-n-Point algorithm can be adopted to estimate $\mathbf{R}$ and $\mathbf{t}$. The minimal amount of needed correspondences is 3 and commonly used solutions to the problem are called P3P algorithms. For a more general case where $n \geq 3$, many solutions are proposed and they are referred to PnP algorithms.

Previous work parameterizes the 3D model of each object with 9 control points in the object coordinate system where 8 points are 8 corners of the tight 3D bounding box fitted to the 3D model and the 9th point is the centroid of the object 3D bounding box [109, 130]. As we can see, once the 3D-to-2D point correspondences are fixed and a standard PnP solver such as [64] is chosen, the quality of the estimated pose $\hat{\mathbf{R}}, \hat{\mathbf{t}}$



(a) (b) (c)

Figure 3.1: Illustration of the correspondences between the 3D keypoints and their 2D projections in images captured by cameras: (a) The observed image, (b) The 3D-to-2D point correspondences of the 3D bounding box corners, (c) The 3D-to-2D correspondences of the object surface keypoints.

relies directly on the accuracy of the estimated 2D coordinates $\{\hat{\mathbf{p}}_i\}_{i=1}^{n}$. Recently proposed approaches mainly focus on a better regression of these 2D coordinates with deep models [109, 130].

3.3.1 Modeling

As introduced previously, people try to make deep models learn the mapping from the object appearance to the object 3D bounding box corners and the centroid in the 2D image. However, 2D coordinates of corners are not easy to estimate with high accuracy in practice. As illustrated in Figure 3.1(b), one main reason is that the 3D bounding box corners in fact represent more the object 3D size than its 3D shape, and their projected 2D points may be far away from the object appearance region. Estimating 2D coordinates of points lying in the region of background without strong object appearance indication is a tough task for deep models, and it usually results in a performance degradation. Another possible reason is the large displacement of each 2D corner point in images captured from different camera views. As the point estimator only see limited views during training, the large displacement of 2D corner point between supervised views and unseen views makes the point localization difficult for the point estimator.

As a first step to alleviate this problem, we consider adding 6 object 3D bounding

box face centroids as additional control points to recover object pose. In this setting, there are therefore 15 3D-to-2D point correspondences which are fed into a PnP algorithm. Compared to bounding box corners, the 2D projections of face centroids are usually closer to the object appearance in the image, they are thus expected to be estimated more easily by deep models. Our experiments in Section 3.5.3 validate our intuition.

In a more general setting, as illustrated in Figure 3.1(c), we propose to directly estimate 2D projections of n points sampled on the object surface and recover the object pose with these n 3D-to-2D point correspondences. Given a point cloud (or mesh vertices) $\{\mathcal{P}_i\}_{i=1}^m$ representing the object shape, we uniformly sample n points from this point set: $\{\mathcal{P}_i\}_{i=1}^n \subset \{\mathcal{P}_i\}_{i=1}^m$, and use the 3D-to-2D point correspondences to recover the object pose. The 2D coordinates to estimate $\{\mathbf{p}_i\}_{i=1}^n$ therefore lie in the region of object appearance which offers more direct indication to deep models. As the number of correspondences can be much bigger than 9 (in our experiments, $n = 100$), we call it a pose estimation with rich 3D-to-2D point correspondences. The number of surface points is chosen mainly based on two factors: the sampled point cloud should represent approximately the object 3D shape; the spatial distance between two sampled points should be large enough so that deep models can easily discriminate each point in images. In our experiments, the object point cloud always contains 3000 points ($m = 3000$), so uniformly sampling 100 points ($n = 100$) from this point set is a reasonable choice.

More specifically, given an RGB image I of size $H \times W \times 3$, we first resize it to size $M \times M \times 3$ and then feed it into our 2D coordinates estimation network f_Θ which has a fully-convolutional architecture. The output of f_Θ is a 3D tensor of size $S \times S \times D$ where $S \times S$ grid cells represent $S \times S$ 2D grid regions of the input image as in YOLO [115]. For each grid cell $j \in S \times S$, the output vector $\mathbf{v}_j$ of size D contains the predicted 2D coordinates of the 3D control points $\{\hat{\mathbf{p}}_i\}_{i=1}^n$, one cell confidence score and C object class scores. In our case, we sample n object surface points as the

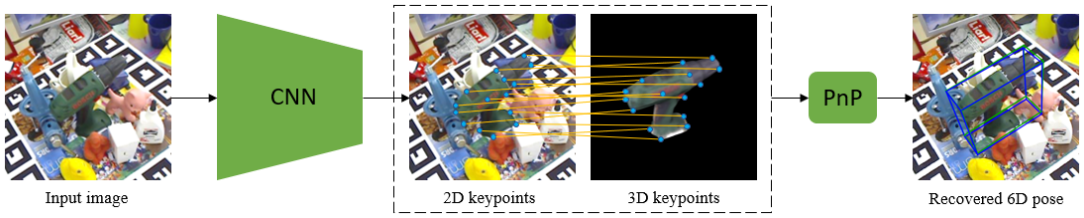


Figure 3.2: Overview of our object pose estimation method. Our model takes an RGB image as input and predicts the 2D projection coordinates of the object surface 3D keypoints in the image. The object 6D pose is estimated by feeding the predicted 3D-to-2D point correspondences into a PnP algorithm [64]. **green**: ground-truth 3D bounding box; **blue**: 3D bounding box estimated by our method.

control points, so $D = n \times 2 + 1 + C$. In each cell, f_Θ actually predicts the offsets w.r.t. the top-left corner of the associated grid region for the 2D coordinates of the control points. The cell confidence score between 0 and 1 presents whether objects are present in the current cell or not. As in image classification tasks, the output C class scores pass trough a softmax function to get C class probabilities. We train our network f_Θ to predict the target tensor of size $S \times S \times D$. During test, we first select the cells containing the object, then get the corresponding 2D coordinates in a voting manner. Concretely, we drop the cells with confidence scores lower than 0.1 and pick the cell which has the maximum confidence score and the cells in the 3×3 neighborhood of it. The final 2D coordinate estimates are weighted averages over all picked cells where the weights are the confidence scores of the associated cells. By this way, multiple cells can contribute to the 2D coordinate localization for cases where objects are rather big in the image. After the 2D coordinates estimation, a PnP algorithm [64] is adopted to recover object 6D poses. Our method is illustrated in Figure 3.2.

3.3.2 Training procedure

Given one color image I, our 2D coordinates estimation network f_Θ outputs 2D coordinates of control points, a confidence score and class probabilities for each grid

cell as introduced in Section 3.3.1. During the training stage, we minimize the 2D Euclidean distance $\mathcal{L}_{pt}$ over the parameters Θ of our network f_Θ to estimate 2D coordinates. To generate the ground-truth cell confidence score, we adopt the confidence function F_{conf} proposed by [130] which decreases with the increment of the average 2D distance between the predicted 2D coordinates $\{\hat{\mathbf{p}}_i\}_{i=1}^n$ and the ground-truth ones $\{\mathbf{p}_i\}_{i=1}^n$. Thus the confidence ground truth is calculated on the fly during training. The object classification ground truth is a one-hot vector as in image classification tasks. To conclude, for each cell, the loss for backpropagation is a linear combination of the 2D coordinates regression loss $\mathcal{L}_{pt}$, the cell confidence loss $\mathcal{L}_{conf}$ and the object classification loss $\mathcal{L}_{cls}$:

$$\mathcal{L} = \mathcal{L}_{pt} + \lambda_{conf}\mathcal{L}_{conf} + \mathcal{L}_{cls} \tag{3.2}$$

$$\mathcal{L}_{pt} = \frac{1}{n}\sum_{i=1}^{n} \|\mathbf{p}_i - \hat{\mathbf{p}}_i\|_{L^2}^2 \tag{3.3}$$

$$\mathcal{L}_{conf} = \left\| F_{conf}(\{\mathbf{p}_i\}_{i=1}^n, \{\hat{\mathbf{p}}_i\}_{i=1}^n) - \hat{C} \right\|^2 , \tag{3.4}$$

where $\mathcal{L}_{cls}$ is a standard cross-entropy loss and $\hat{C}$ is the predicted confidence score. As proposed in [114], λ_{conf} is set to 0.1 for cells which do not overlap objects and is set to 5.0 for cells overlapping objects according to the ground truth. The final loss for training is an average over all cell losses.

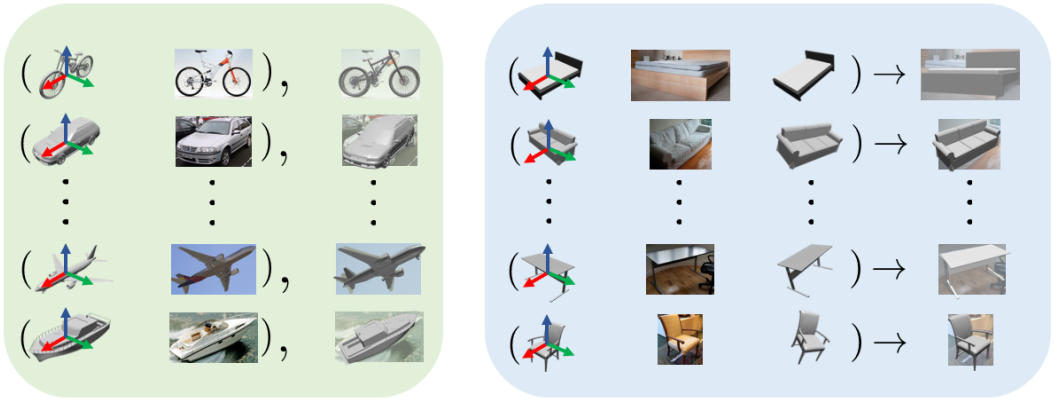
3.3.3 Implementation details

During training, the network input image size M is randomly chose from the set $\{320, 352, ..., 736, 768\}$ to make our network robust to objects with different spatial sizes. Similar to [130], the network downsamples the images by a factor of 32 so that the output spatial size $S = M/32$. When testing, we fix the input image size as 672×672. For all our experiments, we set the batch size to 16 and train our network using the SGD optimizer [128] with a learning rate of 10^{-3} during the first half of

train epochs, then we divide the learning rate by 10 for the rest of training epochs. For the experiments on LINEMOD dataset [46], the number of training epochs is set to 700. We train and test our models on a single TITAN X GPU with 12 GB memory, and the speed during test is around 15 fps.

3.4 Pose Refinement for Objects in the Wild

As introduced in Section 3.3, the object shape (the object 3d model in practice) describes both the object appearance and the linked object frame. Except for offering 3D-to-2D point correspondences for object pose estimation, object shapes can be further used to do pose refinement given initial pose estimates. In this section, we propose a category-agnostic pose refinement method which compares the difference between the observed object appearance and the rendered object using an initial pose estimate, to refine the pose estimate. Similar to [72], we make our models conditioned on the instance-wise object appearance and the object frame offered by the object shape during training. This enables a category-agnostic pose refinement during test as our models do not learn any category-specific information. While existing deep-learning-based methods [86, 72] evaluate pose refinement either on real datasets [46, 150] with constrained backgrounds (e.g., tables) or on synthetic datasets (e.g., ModelNet [147]) with a black background, our pose refinement method can work for objects with varying shapes in real images which are captured during daily life without any experimental setup. We therefore call our method a pose refinement method for objects in the wild. To demonstrate that our method has a good generalization ability on objects which are not seen during training, we divide objects into the objects for training and the ones for test and only evaluate our models on test objects, as illustrated in Figure 3.3.



(a) (b)

Figure 3.3: Illustration of our object pose refinement method. (a) Training data: object 3D shapes, input images and pose annotations for everyday man-made objects; (b) At testing time, pose refinement of object categories which are not seen during training, given object 3D shapes, input images and initial pose estimates.

3.4.1 Modeling

Given an object image I, an initial pose estimate (i.e., a 3D translation vector $\bar{\mathbf{t}} = (\bar{x}, \bar{y}, \bar{z})$ and a 3D rotation matrix $\bar{\mathbf{R}}$) and the relevant object 3D shape, our method directly outputs a relative $SE(3)$ transformation that can be applied to the initial pose estimate to improve the estimate accuracy. Concretely, our model encodes simultaneously the observed image and the object image rendered with the initial pose estimate, and outputs a 3D translation update $\hat{\mathbf{v}} = (\hat{v}_x, \hat{v}_y, \hat{v}_z)$ and a 4D quaternion vector $\hat{\mathbf{q}}$ representing the rotation update matrix $\hat{\mathbf{R}}_\Delta$. Here, we use the object image rendered with the initial estimate to represent the reference view, and the goal of our model is therefore predicting the relative transformation from the reference view to the observed view. Similar to DeepIM [72], the 3D rotation update $\hat{\mathbf{R}}_\Delta$ is defined as the rotation around the estimated object center instead of the camera center so that the rotation update does not have an impact on the translation update. Meanwhile, in order to let the model do not learn the object size information and the metric translation information during training, the 3D translation update $\hat{\mathbf{v}}$ is actually defined as a 2D translation in image space and a scaling factor which matches the

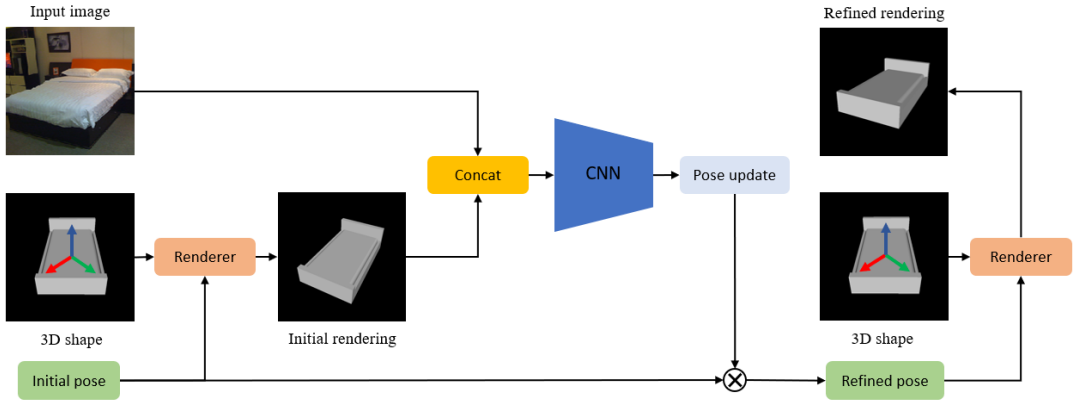


Figure 3.4: Overview of our object pose refinement method. Given an input image, an object shape and an initial pose estimate, our method compares the input image and the rendered object using the initial pose, and predicts a pose update to refine the initial pose estimate.

rendered image of the object against the observed object. Given a camera intrinsic $\mathbf{K}$ containing the focal lengths f_x and f_y, an initial pose estimate $(\bar{\mathbf{t}}, \bar{\mathbf{R}})$ and a pose update $(\hat{\mathbf{v}}, \hat{\mathbf{R}}_\Delta)$, the refined pose $(\hat{\mathbf{t}}, \hat{\mathbf{R}})$ is calculated as:

$$\hat{\mathbf{R}} = \hat{\mathbf{R}}_\Delta \bar{\mathbf{R}} \tag{3.5a}$$

$$\hat{x} = \hat{z}\left(\frac{\hat{v}_x}{f_x} + \frac{\bar{x}}{\bar{z}}\right) \tag{3.5b}$$

$$\hat{y} = \hat{z}\left(\frac{\hat{v}_y}{f_y} + \frac{\bar{y}}{\bar{z}}\right) \tag{3.5c}$$

$$\hat{z} = \frac{\bar{z}}{e^{\hat{v}_z}}, \tag{3.5d}$$

where the refined translation estimate $\hat{\mathbf{t}} = (\hat{x}, \hat{y}, \hat{z})$. The overview of our method is illustrated in Figure 3.4. Meanwhile, as the refined pose estimates can serve as the new initial pose estimates, an iterative refinement strategy can be adopted. In our experiments, we set the number of refinement iterations to 2. For the network architecture, we adopt the encoder of FlowNetSimple [24] and add three fully-connected layers (with dimension 256, 256 and 7) on the top of it to predict the corresponding pose update.

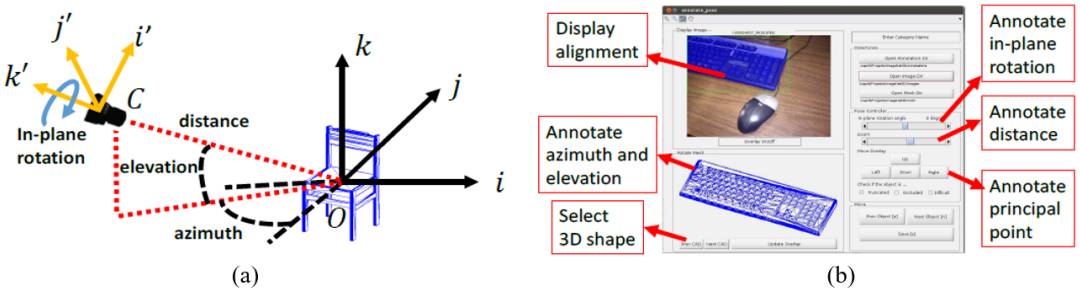


Figure 3.5: Illustration of the annotation system of ObjectNet3D [148]: (a) The virtual camera system, (b) The annotation interface for 2D-3D alignment.

3.4.2 Data generation

For pose estimation datasets with objects in the wild (e.g., Pascal3D+ [149], ObjectNet3D [148]), images are directly collected from the internet without either object pose annotations or camera intrinsic information. Thus, people manually label the pose of the observed object $(\mathbf{t}, \mathbf{R})$ by matching the rendered object against the observed object in a virtual camera system where the camera always points towards the object center and the camera focal length is constant across the dataset (cf. Figure 3.5). By using this strategy, the labeled annotations for an object are the rotation matrix $\mathbf{R}$, the distance d between the object and the camera, and the camera principal point. As the translation annotation $\mathbf{t} = (0, 0, d)$ and $\mathbf{t}$ is not the real 3D translation annotation between the object and the camera which captures the object, we therefore restrict our object pose refinement method to the object 3D rotation refinement method for objects in the wild.

Due to the model architecture property (i.e., fully connected layers predicting relative transformations), existing deep-learning-based methods [86, 72] do pose refinement on images of a fixed spatial size, however, the image spatial size is varied for images in the wild. To alleviate this constraint, here we propose a method to generate model input images from images of any spatial size. Given an image I containing objects, we first generate the observed image of the concerned object $I_{\mathcal{B}}$ by cropping I with the object ground-truth 2D bounding box $\mathcal{B}$. In practice, the area

of the cropped image $I_{\mathcal{B}}$ is larger than the area of $\mathcal{B}$ by a fixed ratio to include some image contexts around the object. Following the annotation convention, the camera still points towards the object center and the distance between the object and the camera is d. As the network needs a fixed size input image, $I_{\mathcal{B}}$ is scaled by a scaling factor s_1 and then pasted on the center of a black image of fixed size (e.g., 640×480) to generate the new object image I_o. The scaling factor s_1 is calculated automatically to ensue that the object image is smaller than the black image of fixed size. For the efficiency of training and test, the virtual camera intrinsic is fixed across images and we set the distance between the object and the camera to d/s_1. With this process, an image I of any spatial size can be converted to the object image of a fixed size I_o while the object image and the annotation are still aligned. For generating the rendered object I_r, we render the concerned object in the same virtual camera system with the ground-truth translation $\mathbf{t} = (0, 0, d/s_1)$ and an initial rotation estimate $\bar{\mathbf{R}}$. Similar to [72], a zoom-in mechanism is used on I_o, I_r afterwards to zoom into the object region based on the ground-truth bounding box $\mathcal{B}$ during training or the estimated bounding box $\hat{\mathcal{B}}$ from the rendered object during test. The zoom-in scaling factor is s_2 and the corresponding ground-truth translation $\mathbf{t} = (0, 0, d/s_1 s_2)$. The model input images are zoom-in versions of I_o, I_r which share the same spatial size with I_o, I_r.

For objects in the wild, object CAD models usually do not have the exact surface textures of the objects in images while objects in 6D pose datasets (e.g., LINEMOD [46] and YCB [150]) have instance-wise high-quality textures. Compared to existing methods [86, 72] which train their models with the rendered objects using instance-specific textures, our models are trained completely with an uniform gray texture to make our model have a better generalization ability. By this way, our models will learn to focus more on comparing the difference between the observed object shapes and the rendered object shapes, instead of the difference between the object textures.

For the generation of the observed objects, we also propose an alternative choice

Figure 3.6: Generated synthetic data for the training of our object pose refinement models.

to validate the synthetic-to-real setup. Except for using the cropped real image I_o as stated previously, the observed object image can also be a synthetic rendered object using a generated ground-truth pose and a domain randomization as in [77]. Concretely, the rendered object is first generated from its 3D model and a randomly selected surface texture from Unreal Engine to offer a great number of random object appearances. Then the rendered object is pasted on a background image which is randomly picked from either SUN397 [151] dataset or PASCAL-VOC [28]. As shown in Figure 3.6, this domain randomization strategy finally offers a great amount of samples for training. As shown in Section 3.5.4, our model trained only on the generated synthetic data can also achieve a pose refinement performance on real images, despite an a priori domain gap between the real images and the synthetic ones.

3.4.3 Training procedure

To train the network, we add noise following Gaussian distribution to the ground-truth poses as the initial pose estimates and feed the observed object image and the rendered object image into the network as input. The network predictions $\hat{\mathbf{v}}, \hat{\mathbf{q}}$ introduced in Section 3.4.1 representing the estimated relative transformations between the ground-truth pose and its noisy version. As stated in [55, 72], given the ground-truth pose $[\mathbf{R}|\mathbf{t}]$ and the refined pose $[\hat{\mathbf{R}}|\hat{\mathbf{t}}]$, we use a point-wise matching loss between the object

3D points to train the refinement model:

$$\mathcal{L}_{pose} = \frac{1}{n}\sum_{i=1}^{n}\left|(\mathbf{R}\mathbf{P}_i + \mathbf{t}) - (\hat{\mathbf{R}}\mathbf{P}_i + \hat{\mathbf{t}})\right|_{L^1}, \tag{3.6}$$

where $\mathbf{P}_i$ is a randomly selected 3D point on the object model surface, we set $n = 3000$ in our experiments. By this way, it measures how the transformed 3D models match against each other for the pose estimation. In our experiments about the object rotation refinement for objects in the wild, we always set $\hat{\mathbf{t}} = \mathbf{t}$ due to the dataset property introduced in Section 3.4.2. During training, we also add the decoder of FlowNetSimple [24] on the top of our model encoder to predict optical flow (between the observed object and the rendered object) and binary object mask. As observed in [72], the auxiliary predictions stabilize the training process. For training, we additionally use the optical flow loss $\mathcal{L}_{flow}$ as in FlowNet [24] and the sigmoid cross-entropy loss as the mask loss $\mathcal{L}_{mask}$. The entire loss $\mathcal{L}_{refine}$ for the model backpropagation is:

$$\mathcal{L}_{refine} = \alpha\mathcal{L}_{pose} + \beta\mathcal{L}_{flow} + \gamma\mathcal{L}_{mask}, \tag{3.7}$$

where $\alpha = 0.1$, $\beta = 0.25$ and $\beta = 0.03$ over all experiments.

3.4.4 Implementation details

For all our experiments, we set the batch size to 16 and train our network using the SGD optimizer [128] with a learning rate of 10^{-4} during the first half of train epochs, then we divide the learning rate by 10 for the rest of training epochs. For the experiments on ObjectNet3D [148], the number of training epochs is set to 16. We train and test our models on a single TITAN X GPU with 12 GB memory, the speed during test is around 12 fps with two refinement iterations. For training, the rotation noise added on the ground-truth rotation is an combination of angular noises added on all three Euler angles (azimuth, elevation and in-plane rotation).

Concretely, the angular noise for each Euler angle follows a Gaussian distribution with $\mu = 0, \sigma = 15$ degrees.

3.5 Experimental Evaluation

3.5.1 Datasets

LINEMOD [46] is a de facto standard benchmark for 6D object pose estimation. The central object in each RGB image of the videos is assigned a ground-truth 6D pose and an object class ID. A textured 3D mesh representing the object shape is also provided. The datasets contains 13 objects and there are around 1200 instances for each object.

ObjectNet3D [148] is a large-scale 3D dataset (90,127 images collected from ImageNet [22]) containing 100 object categories selected from the ShapeNet repository [13] (e.g., airplane, table, car, etc.) with a big variety of object shapes (791 CAD models). For each category, there exist many object instances with shape variations. As the images were captured by different cameras with different camera intrinsics which were not available, a virtual camera system was used to manually adjust the rendered object (using approximated object CAD models selected from ShapeNet [13]) to the observed object, yielding object pose annotations. Thus it only provides approximate object models and relatively rough alignments between the observed objects and the object pose annotations. As the true 3D translation annotation from the camera to the object is not available on ObjectNet3D, we evaluate the object 3D rotation refinement on object images based on the ground-truth object 2D bounding box in the experiments.

3.5.2 Evaluation metrics

The 2D Projection Error measures the average Euclidean distance in pixels between the 2D projections of 3D points using the ground-truth pose and the ones estimated by models.

ADD-0.1d. As introduced in [12, 54, 109, 130], the ADD metric evaluates the accuracy of pose estimation in 3D. For ADD-0.1d metric, a pose estimate is regarded as correct if the mean distance between the coordinates of 3D mesh vertices transformed by the ground-truth pose and those transformed by the estimated pose is less than 10% of the object 3D bounding box diameter. The final score is an average over all pose estimates for each object. For objects with rotational symmetry, we use a modified version ADD-S-0.1d as in [12, 54, 109, 130].

Acc$_{\frac{\pi}{6}}$ evaluates the correctness of object 3D rotation estimation. A pose estimate is regarded as correct if the angular distance between the ground-truth rotation and the estimated rotation is less than 30 degrees. The final score is an average over all pose estimates for each object category.

MedErr is the median number of angular distances between ground-truth rotations and estimated ones across all pose estimates for each object category.

3.5.3 Pose estimation with rich 3D-to-2D point correspondences

We first evaluate the precision of 2D coordinates estimation on LINEMOD [46] to study the impact of selecting different 3D control points. Compared to our baseline approach Tekin [130] which estimates the 3D bounding box corners and the object centroid, adding the 3D bounding box face centroids (6 centroids on 6 faces), i.e., 'Ours (15 points)' in Table 3.1, can reduce the 2D coordinates estimation errors.

LINEMOD	ape	bvise	cam	can	cat	drill	duck	ebox	glue	holep	iron	lamp	phone	**mean**
2D Projection Error in pixels (lower is better)														
Baseline [130]	4.05	5.00	4.59	4.08	3.84	5.58	4.89	4.38	3.57	5.62	7.08	7.05	7.86	5.20
Ours (15 points)	3.52	4.53	4.35	3.61	3.55	6.03	5.28	3.92	3.33	4.92	6.50	6.26	6.67	4.81
Ours (100 points)	**2.28**	**3.06**	**3.26**	**2.59**	**2.38**	**4.32**	**3.71**	**3.76**	**2.51**	**4.01**	**5.97**	**4.26**	**5.41**	**3.66**

Table 3.1: 2D error of estimated projections of 3D points on LINEMOD [46]

This is because the 2D projections of 3D bounding box face centroids are usually closer to the object region than the ones of bounding box corners in the image. Meanwhile, estimating the 2D projections of object surface points which are always in the object region of the image, i.e., 'Ours (100 points)' in Table 3.1, achieves the best performance among all three methods. The results demonstrate that estimating 2D projections near the object region is indeed easier for deep models to learn and predict, as discussed in Section 3.3.1. In the following paragraphs, we evaluate our method, i.e., 'Ours (100 points)' in Table 3.1, on 6D pose estimation metrics and compare it with other existing methods [12, 54, 109, 130].

The performance of our method (cf. 'Ours') is displayed in Table 3.2. Compared to other state-of-the-art methods without an additional object pose refinement stage, our method achieves the best performance. When comparing with the methods using an object pose refinement stage, we adopt the recent pose refinement method proposed by DeepIM [72] and do pose refinement on the initial pose estimates predicted by our method. As shown in the lower part of Table 3.2, our method also achieves the best performance. We also report the accuracy numbers without the refinement using the ADD metric in Table 3.3 for different thresholds (cf. 'ADD-0.1d', 'ADD-0.3d', 'ADD-0.5d'). Compared to our baseline method 'Tekin' [130], our method consistently improve the pose estimation performance across metrics. These results demonstrate the effectiveness of our method and the benefits of using rich 3D-to-2D point correspondences in object pose estimation task. Some qualitative results are shown in Figure 3.8. As an ablation study of our method, we also investigate the pose estimation performance in function with the number of sampled points using the

LINEMOD		ape	bvise	cam	can	cat	drill	duck	ebox*	glue*	holep	iron	lamp	phone	**mean**
ADD-0.1d in percentages (higher is better)															
w/o Refine.	Branchmann [12]	-	-	-	-	-	-	-	-	-	-	-	-	-	32.2
	SSD-6D [54]	0	0.2	0.4	1.4	0.5	2.6	0	8.9	0	0.3	8.9	8.2	0.2	2.4
	BB8 [109]	27.9	62.0	40.1	48.1	**45.2**	58.6	**32.8**	40.0	27.0	42.4	67.0	39.9	35.2	43.6
	Tekin [130]	21.6	78.7	33.4	67.5	41.8	63.5	25.6	65.5	80.1	42.3	**69.3**	67.9	**47.7**	54.2
	Ours	**33.9**	**80.6**	**43.9**	**75.2**	44.2	**66.1**	29.2	**66.7**	**81.1**	**48.6**	63.9	**71.4**	42.0	**57.5**
w/ Refine.	Branchmann [12]	33.2	64.8	38.4	62.9	42.7	61.9	30.2	49.9	31.2	52.8	80.0	67.0	38.1	50.2
	SSD-6D [54]	65	80	78	86	70	73	66	**100**	**100**	49	78	73	79	79.0
	BB8 [109]	40.4	91.8	55.7	64.1	62.6	74.4	44.3	57.8	41.2	**67.2**	84.7	76.5	54.0	62.7
	Ours	**78.7**	**97.7**	**93.8**	**97.2**	**82.7**	**95.0**	**77.6**	98.7	99.3	53.1	**98.4**	**97.6**	**88.8**	**89.1**

Table 3.2: Quantitative results of object pose estimation on LINEMOD [46]. * ADD-S-0.1d used for symmetric objects eggbox and glue.

LINEMOD		ape	bvise	cam	can	cat	drill	duck	ebox*	glue*	holep	iron	lamp	phone	**mean**
ADD-0.1d in percentages (higher is better)															
w/o Refine.	Tekin [130]	21.6	78.7	33.4	67.5	41.8	63.5	25.6	65.5	80.1	42.3	**69.3**	67.9	**47.7**	54.2
	Ours	**33.9**	**80.6**	**43.9**	**75.2**	44.2	**66.1**	29.2	**66.7**	**81.1**	**48.6**	63.9	**71.4**	42.0	**57.5**
ADD-0.3d in percentages (higher is better)															
w/o Refine.	Tekin [130]	70.7	91.1	81.6	99.0	90.6	97.4	70.7	81.3	89.0	85.5	**98.9**	98.9	**91.1**	88.1
	Ours	**75.4**	**99.5**	**87.3**	**99.1**	90.6	**97.8**	**73.5**	**97.7**	**95.3**	**90.4**	98.2	98.9	91.0	**91.9**
ADD-0.5d in percentages (higher is better)															
w/o Refine.	Tekin [130]	88.1	98.9	94.8	99.9	**98.1**	99.2	85.1	98.3	**97.2**	96.3	99.4	99.6	**98.9**	96.4
	Ours	**90.0**	**99.9**	**97.1**	99.9	97.9	**99.7**	**88.9**	**99.0**	96.9	**97.5**	99.4	**99.8**	98.8	**97.3**

Table 3.3: Quantitative results of object pose estimation on LINEMOD [46] using ADD metrics with different thresholds. * ADD-S used for symmetric objects eggbox and glue.

ADD-0.1d metric on LINEMOD [46]. The performance plot displayed in Figure 3.7 shows that our method achieves the best performance when the number of points is 100. One explanation of this phenomenon is that the sampled point cloud should represent approximately the object 3D shape and be easy to be discriminated by deep models in images, as discussed in Section 3.3.1.

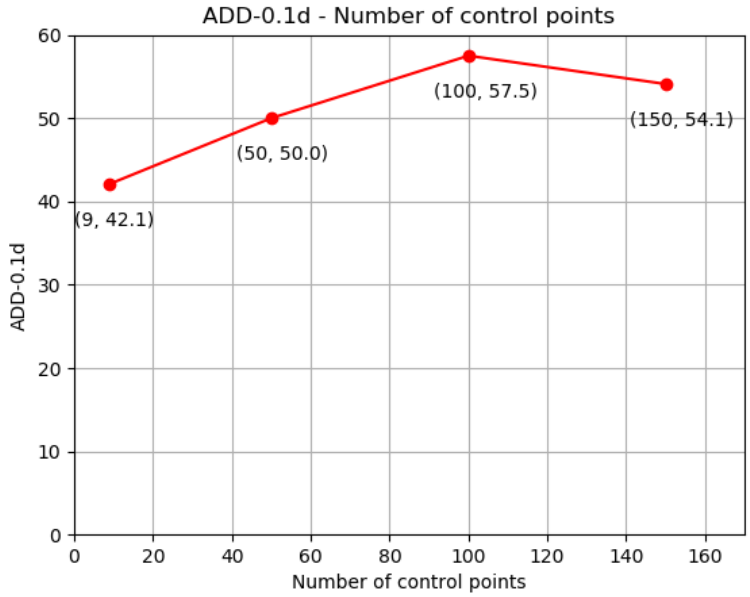


Figure 3.7: The pose estimation performance in function with the number of sampled points using the ADD-0.1d metric on LINEMOD [46].

Figure 3.8: Qualitative results of our pose estimation method on LINEMOD [46]. **green**: ground-truth 3D bounding box; **blue**: 3D bounding box estimated by our method.

ObjectNet3D	bed	bookcase	calc	cellphone	comp	door	cabinet	guitar	iron	knife	micro
***Acc*$_{\frac{\pi}{6}}$ in percentages (higher is better)**											
Init	76	73	79	80	80	78	77	82	76	**72**	81
Ours (synthetic)	82	74	77	74	82	80	83	84	82	71	**94**
Ours (real)	**85**	**85**	**91**	**85**	**90**	**92**	**89**	**86**	**93**	**72**	91
***MedErr* in degrees (lower is better)**											
Init	21	24	22	21	22	21	21	19	19	24	23
Ours (synthetic)	20	23	20	21	16	19	17	18	20	24	**14**
Ours (real)	**16**	**19**	**17**	**18**	**15**	**15**	**15**	**16**	**17**	**21**	**14**

	pen	pot	rifle	shoe	slipper	stove	toilet	tub	wheelchair	**mean**
***Acc*$_{\frac{\pi}{6}}$ in percentages (higher is better)**										
Init	79	76	79	76	**80**	72	76	72	80	77
Ours (synthetic)	77	74	86	78	74	84	73	75	76	79
Ours (real)	**81**	**80**	**89**	**85**	78	**88**	**82**	**84**	**84**	**86**
***MedErr* in degrees (lower is better)**										
Init	22	21	18	23	**21**	22	22	24	22	22
Ours (synthetic)	21	19	17	19	22	17	23	21	20	20
Ours (real)	**18**	**18**	**16**	**17**	**21**	**15**	**18**	**19**	**18**	**17**

[images: 90,127 in the wild | objects: 201,888 | categories: 100 | 3D models: 791]

Table 3.4: Pose refinement for unseen objects on ObjectNet3D [148]. Train and test are the same as [163]; Training is on 80 object categories and test is on the other 20 unseen object categories.

3.5.4 Pose refinement for objects in the Wild

We evaluate our method on ObjectNet3D [148]. To verify the generalization ability of our method for unseen object categories, we follow the protocol of StarMap [163]: we evenly hold out 20 categories (every 5 categories sorted in the alphabetical order) from the training data and only use them for testing. The initial 3D rotation for test is obtained by adding angular noises to the Euler angles of the ground-truth rotation label where the angular noise for each Euler angle follows a Gaussian distribution with $\mu = 0, \sigma = 15$ degrees. As shown in Table 3.4, our models trained on either synthetic data (cf. 'Ours (synthetic)') or real data (cf. 'Ours (real)') can achieve a good pose refinement performance despite the challenge of applying it to images in the wild. The results demonstrate the effectiveness of our refinement method for objects in the wild and the good generalization ability of our models on novel objects categories. Some qualitative results are shown in Figure 3.9.

LINEMOD [46]	ape	bvise	cam	can	cat	drill	duck	ebox*	glue*	holep	iron	lamp	phone	**mean**
ADD-0.1d in percentages (higher is better)														
Init	4.3	19.0	14.9	8.8	12.2	18.8	6.4	100	**81.85**	20.9	9.8	**25.2**	10.8	25.6
w/ our refinement	**24.7**	**24.7**	**21.1**	**15.9**	**17.5**	**36.1**	**13.0**	100	54.63	**32.9**	**27.6**	25.0	**20.1**	**31.8**

Table 3.5: Pose refinement for unseen objects on LINEMOD [46] using initial pose estimates provided by our initial pose estimation model [152]. Our initial pose estimation model [152] is trained on ShapeNetCore [13] and our refinement model is trained on ObjectNet3D [148]. * ADD-S-0.1d used for symmetric objects eggbox and glue.

In order to further prove the generalization ability of our pose refinement method, we choose a cross-domain unseen pose refinement setup where the two domains are significantly different in images and objects. Concretely, we train our model on synthetic data generated from ObjectNet3D [148] and test it on LINEMOD [46] given initial pose estimates by our initial pose estimation model [152] which is trained on synthetic data generated from ShapeNetCore [13]. This setup means that both the initial pose estimator and the pose refiner are trained only on synthetic data and tested on a new real dataset used for evaluation. As shown in Table 3.5, our refinement method can consistently refine our initial pose estimates on novel objects from LINEMOD [46]. The results in this hard setup further demonstrate the generalization ability of our category-agnostic pose refinement method. Some qualitative results are displayed in Figure 3.10.

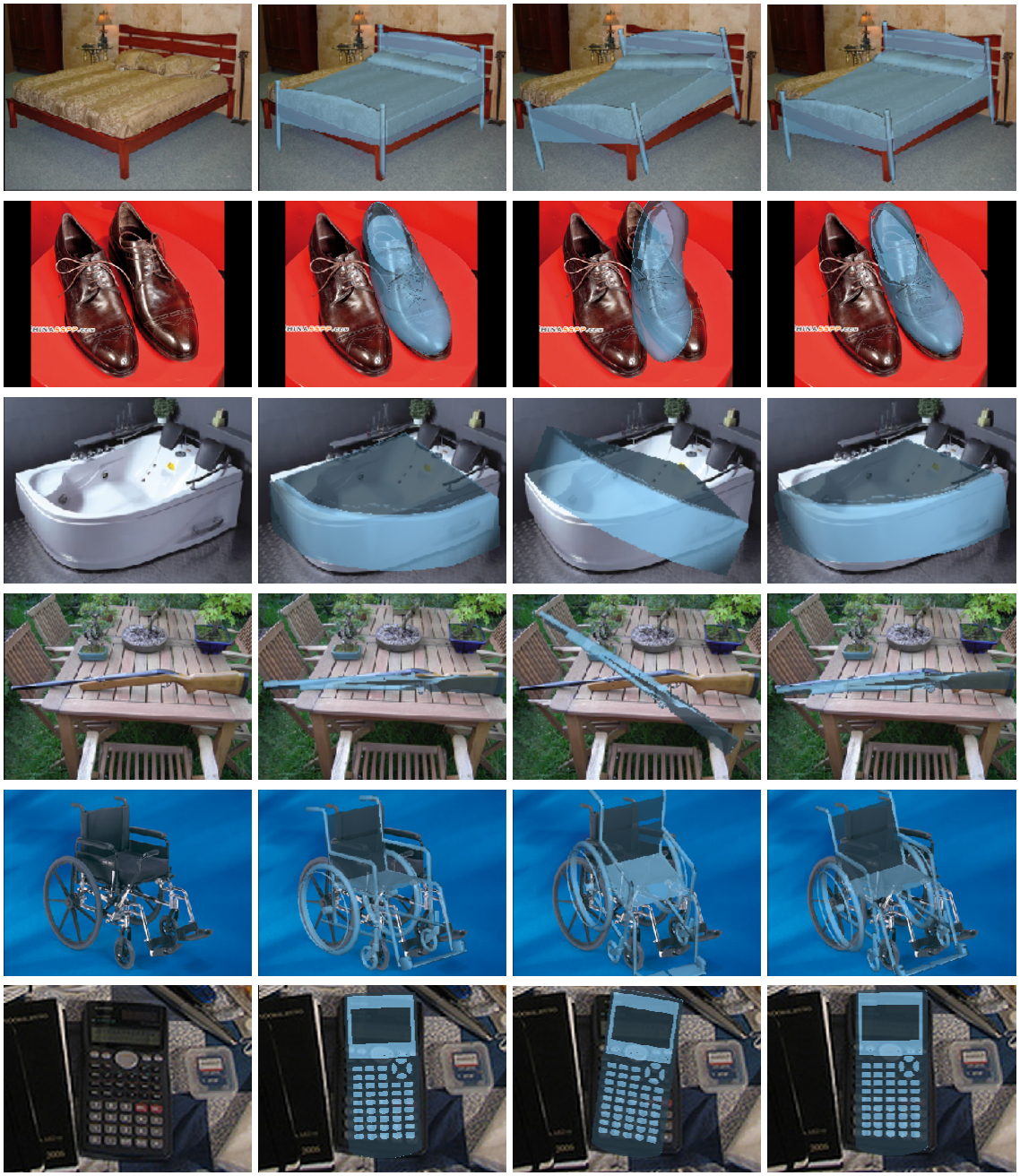


Figure 3.9: Qualitative results of our pose refinement method on ObjectNet3D [148]. For each sample, the four columns from left to right: the input image, the rendered object with the ground-truth pose, the rendered object with the initial pose estimate and the rendered object with the refined pose.

Figure 3.10: Qualitative results of our pose refinement method on LINEMOD [46]. For each sample, the four columns from left to right represent: the input image, the rendered object with the ground-truth pose, the rendered object with the initial pose estimate by our initial pose estimator [152] and the rendered object with refined pose.

3.6 Conclusion

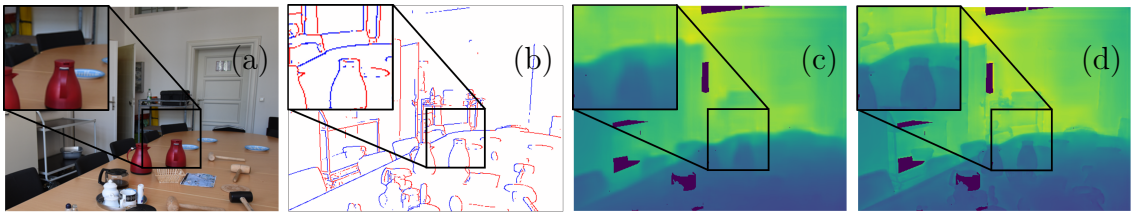
In this chapter, we analyze the use of object shape information in object pose estimation tasks. With this information, we first propose to use rich 3D-to-2D point correspondences between the object 3D shape and the image of object for a more accurate object pose estimation. Then we propose a category-agnostic pose refinement method to refine coarse pose estimates for objects in the wild. Experiments prove that the proposed monocular pose estimation method can boost the performance of pose estimation and our refinement method can achieve good pose refinement results on novel object categories which are not seen during training.

Chapter 4

Scene Occlusion Relationship and Depth Estimation

4.1 Introduction

Occlusions are ubiquitous in 2D images (cf. Figure 4.1(a)) and constitute a major obstacle to address scene understanding rigorously and efficiently. Besides the joint treatment of occlusion when developing techniques for specific tasks [112, 51, 143, 103, 97, 109, 49], task-independent occlusion reasoning [116, 63, 131, 141, 138, 81] offers valuable occlusion-related features for high-level scene understanding tasks.



(a) (b) (c) (d)

Figure 4.1: Illustration of the proposed methods: (a) input image, (b) estimated horizontal occlusion relationship (a part of P2ORM) where red (resp. blue) pixels occlude (resp. are occluded by) their right-hand pixel, (c) depth estimation obtained by a state-of-the-art method [112], (d) our depth refinement method based on occlusion relationships.

In this work, we are interested in one most valuable but challenging scenario of task-independent occlusion reasoning where the input is a single image and the output is the corresponding pixel-level occlusion relationship in the whole image domain (cf. Figure 4.1(b)); the goal is to capture both the localization and orientation of the occlusion boundaries, similar to previous work such as [131, 141, 138, 81]. In this context, informative cues are missing compared to other usual scenarios of occlusion reasoning, in particular semantics [110], stereo geometry [164] and inter-frame motion [31]. Moreover, the additional estimation of orientation further increases the difficulty compared to usual occlusion boundary estimation [3, 40, 31]. Despite of recent progress achieved via deep learning [141, 138, 81], the study on pixel-level occlusion relationship in monocular images is still relatively limited and the state-of-the-art performance is still lagging.

Here, we formalize concepts around geometric occlusion in 2D images (i.e., ignoring semantics), and propose a unified formulation, called *Pixel-Pair Occlusion Relationship Map (P2ORM)*, that captures both localization and orientation information of occlusion boundaries. Our representation simplifies the development of estimation methods, compared to previous works [131, 141, 138, 81]: a common ResNet-based [39] U-Net [119] outperforms carefully-crafted state-of-the-art architectures on both indoor and outdoor datasets, with either low-quality or high-quality ground truth. Besides, thanks to the modularity regarding pixel-level classification methods, better classifiers can be adopted to further improve the performance of our method. In addition, P2ORM can be easily used in scene understanding tasks to increase their performance. As an illustration, we develop a depth map refinement module based on P2ORM for monocular depth estimation (Figure 4.1(c-d)). Experiments demonstrate that it significantly and consistently sharpens the edges of depth maps generated by a wide range of methods [26, 74, 61, 68, 30, 73, 53, 112, 156], including the method targeted at sharp edges [112].

Moreover, our representation derives from a 3D geometry study that involves a

first-order approximation of the observed 3D scene, offering a way to create high-quality occlusion annotations from a depth map with given or estimated surface normals. This allows the automated generation of large-scale, accurate datasets from synthetic data [70] (possibly with domain adaptation [161] for more realistic images) or from laser scanners [58]. Compared to manually annotated dataset that is commonly used [116], we generate a high-quality synthetic dataset of that is two orders of magnitude larger.

Owing to our P2ORM formulation and generated annotations, we further propose a one-stage monocular depth estimation method that recovers high-quality depth discontinuities without depth map refinement procedures. Experiments demonstrate that our method improves the state-of-the-art performance on depth boundary regions.

Our contributions are:

- A formalization of geometric occlusion in 2D images and a relevant occlusion annotation generation method that creates three datasets: InteriorNet-OR, iBims-1-OR and NYUv2-OR.

- A new formulation capturing occlusion relationship at pixel-pair level, from which usual boundaries and orientations can be computed.

- An occlusion estimation method that outperforms the state-of-the-art on several datasets.

- The illustration of the relevance of this formulation with an application to depth map refinement that consistently improves the performance of state-of-the-art monocular depth estimation methods.

- A single-stage monocular depth estimation method that recovers high quality depth discontinuities.

- Our code and data are available at http://imagine.enpc.fr/~qiux/P2ORM/.

The remainder of the chapter is structured as follows: We discuss related work in Section 4.2. We formalize geometric occlusion existing in single images in Section 4.3. We propose the new formulation capturing occlusion relationship at pixel-pair level and the corresponding occlusion estimation method in Section 4.4. We introduce a depth map refinement method using estimated occlusions in Section 4.5. The proposed single-stage monocular depth estimation method focusing on depth discontinuities is presented in Section 4.6. Implementation details and experimental evaluation are presented in Section 4.7 and Section 4.8 respectively. We make our conclusion in Section 4.9.

4.2 Related Work

Task-independent occlusion relationship in monocular images has long been studied due to the importance of occlusion reasoning in scene understanding. Early work often estimates occlusion relationship between simplified 2D models of the underlying 3D scene, such as blocks world [118], line drawings [126, 17] and 2.1-D sketches [94]. Likewise, [47] estimates figure/ground labels using an estimated 3D scene layout. Another approach combines contour/junction structure and local shapes using a Conditional Random Field (CRF) to represent and estimate figure/ground assignment [116]. [131] learns border ownership cues and impose a border ownership structure with structured random forests. Specific devices, e.g., with multi-flash imaging [113], have also been developed.

Recently, an important representation was used in several deep models to estimate occlusion relationship [141, 138, 81]: a pixel-level binary map encoding the localization of the occlusion boundary and an angle representing the oriented occlusion direction, indicating where the foreground lies w.r.t. the pixel.

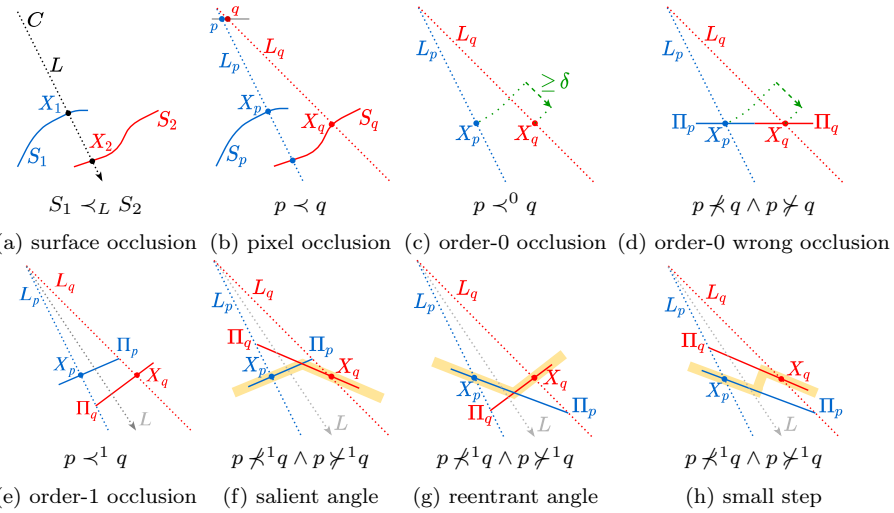
This theme is also closely related to *occlusion boundary detection*, which ignores orientation. Existing methods often estimate occlusion boundaries from images sequences. To name a few, [3] detects T-junctions in space-time as a strong cue to estimate occlusion boundaries; [123] adds relative motion cues to detect occlusion boundaries based on an initial edge detector [88]; [31] further exploits both spatial and temporal contextual information in video sequences. Also, [158, 159, 76, 1] detect object boundaries between specific semantic classes.

Monocular depth estimation is extremely valuable for geometric scene understanding, but very challenging due to its high ill-posedness. Yet significant progress has been made with the development of deep learning and large labeled datasets. Multi-scale networks better explore the global image context [26, 25, 61]. Depth estimation also is converted into an ordinal regression task to increase accuracy [30, 62]. Other approaches propose a better regression loss [53] or the inclusion of geometric constraints from either single images [107, 156] or stereo image pairs [41, 34].

Depth map refinement is often treated as a post-processing step, using CRFs [140, 154, 44, 117]: an initial depth prediction is regularized based on pixel-wise and pairwise energy terms depending on various guidance signals. These methods now underperform state-of-the-art deep-learning-based methods without refinement [107, 53, 156] while being more computationally expensive. Recently, [111] predicts image displacement fields to sharpen initial depth predictions.

4.3 Formalizing and Representing Geometric Occlusion

In this section, we provide formal definitions and representations of occlusion in single images based on scene geometry information. It enables the generation of accurate



(a) surface occlusion (b) pixel occlusion (c) order-0 occlusion (d) order-0 wrong occlusion

(e) order-1 occlusion (f) salient angle (g) reentrant angle (h) small step

Figure 4.2: Occlusion configurations (solid lines represent real or tangent surfaces, dotted lines are imaginary lines): (a) S_1 occludes S_2 along L; (b) p occludes q as S_p occludes S_q along L_p; (c) p occludes q at order 0 as $\|X_q\| - \|X_p\| \geq \delta > 0$, cf. Equation (4.1); (d) no occlusion despite order-0 occlusion as Π_p, Π_q do not occlude one another; (e) p occludes q at order 1 as tangent plane Π_p occludes tangent plane Π_q in the $[L_p, L_q]$ cone, cf. Equation (4.2); no occlusion for a (f) salient or (g) reentrant angle between tangent planes Π_p, Π_q, cf. Equation (4.2); (h) tangent plane occlusion superseded by order-0 non-occlusion, cf. Equation (4.2).

datasets and the development of an efficient inference method.

We consider a camera located at C observing the surface $\mathcal{S}$ of a 3D scene. Without loss of generality, we assume $C = \mathbf{0}$. We note L a ray from C, and L_X the ray from C through 3D point X. For any surface patch S on $\mathcal{S}$ intersecting L, we note $L \cap S$ the closest intersection point to C, and $\|L \cap S\|$ it distance to C.

4.3.1 Approximating occlusion at order 0

Given two surface patches S_1, S_2 on $\mathcal{S}$ and a ray L (cf. Figure 4.2(a)), we say that S_1 *occludes* S_2 *along* L, noted $S_1 \prec_L S_2$ (meaning S_1 comes before S_2 along L), iff L intersects both S_1 and S_2, and the intersection $X_1 = L \cap S_1$ is closer to C than $X_2 = L \cap S_2$, i.e., $\|X_1\| < \|X_2\|$.

Now given neighbor pixels $p, q \in \mathcal{P}$, which are also the 2D projections of 3D points

in the image plane, we say that p *occludes* q, noted $p \prec q$, iff there are surface patches S_p, S_q on $\mathcal{S}$ containing respectively X_p, X_q such that S_p occludes S_q along L_p, (cf. Figure 4.2(b)). Assuming $L_p \cap S_q$ exists and $\|L_p \cap S_q\|$ can be approximated by $\|L_q \cap S_q\| = \|X_q\|$, it leads to a common definition that we qualify as "order-0". We say that p *occludes* q *at order 0*, noted $p \prec^0 q$ iff X_q is deeper than X_p (cf. Figure 4.2(c)):

$$p \prec^0 q \quad \text{iff} \quad \|X_p\| < \|X_q\| \,. \tag{4.1}$$

The depth here is w.r.t. the camera center ($d_p = \|X_p\|$), not to the image plane. This definition is constructive (can be tested) and the relation is antisymmetric. The case of a minimum margin $\|X_q\| - \|X_p\| \geq \delta > 0$ is considered as the existence of occlusion.

However, when looking at the same continuous surface patch $S_p = S_q$, the incidence angles of L_p, L_q on S_p, S_q may be such that order-0 occlusion is satisfied whereas there is no actual occlusion, as S_q does not pass behind S_p (cf. Figure 4.2(d)). This yields many false positives, e.g., when the camera observes planar surfaces such as walls.

4.3.2 Approximating occlusion at order 1

To address this issue, we consider an order-1 approximation of the surface. We assume the scene surface $\mathcal{S}$ is regular enough for a normal $\mathbf{n}_X$ to be defined at every point X on $\mathcal{S}$. For any pixel p, we consider Π_p the tangent plane at X_p with normal $\mathbf{n}_p = \mathbf{n}_{X_p}$. Then to assess if p *occludes* q *at order 1*, noted $p \prec^1 q$, we approximate locally S_p by Π_p and S_q by Π_q, and study the relative occlusion of Π_p and Π_q, cf. Figure 4.2(d-h).

Looking at a planar surface as in Figure 4.2(d), we now have $p \prec^0 q$ as $\|X_p\| < \|X_q\|$, but $p \not\prec^1 q$ because Π_p does not occlude Π_q, thus defeating the false positive at order 0. A question, however, is on which ray L to test surface occlusion, cf. Figure 4.2(a). If we choose $L = L_p$, cf. Figure 4.2(b), only Π_q (approximating S_q) is actually consid-

ered, which is less robust and can lead to inconsistencies due to the asymmetry. If we choose $L = L_{(p+q)/2}$, which passes through an imaginary middle pixel $(p+q)/2$, the formulation is symmetrical but there are issues when Π_p, Π_q form a sharp edge (salient or reentrant) lying between L_p and L_q, cf. Figure 4.2(f-g), which is a common situation in man-made environments. Indeed, the occlusion status then depends on the edge shape and location w.r.t. $L_{(p+q)/2}$, which is little satisfactory. Besides, such declared occlusions are false positives.

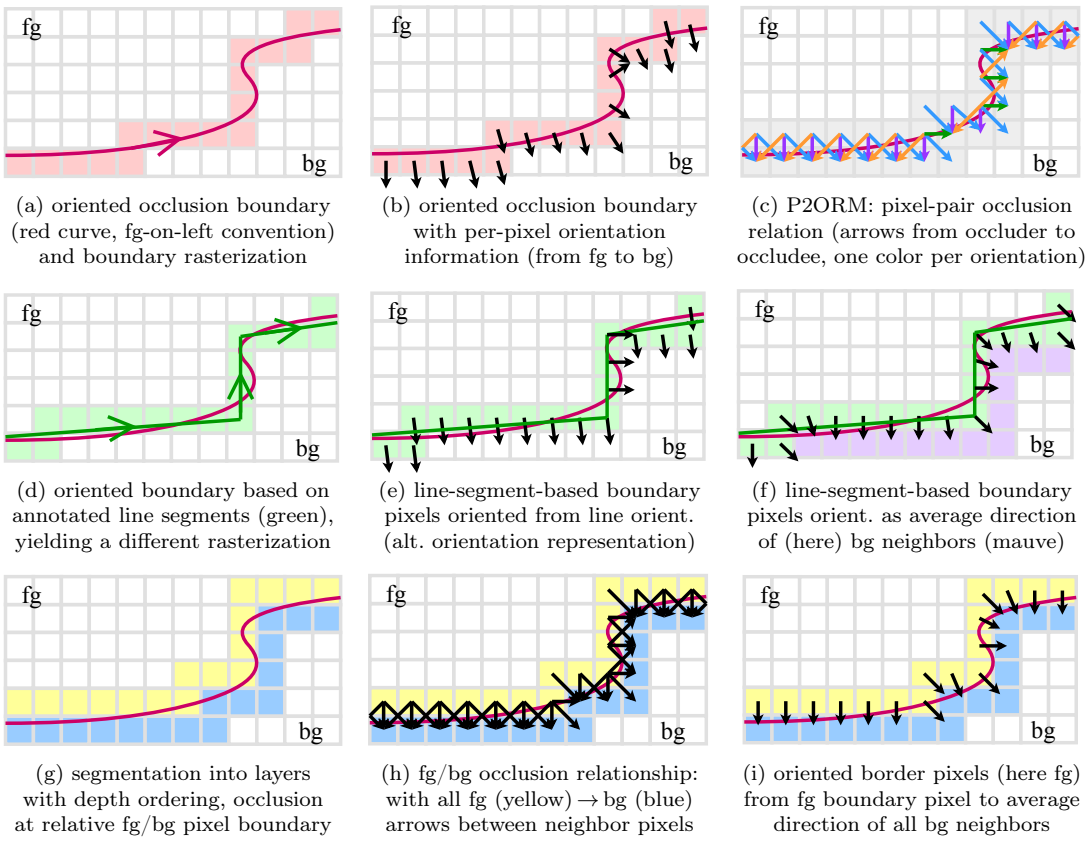
To solve this problem, we define order-1 occlusion $p \prec^1 q$ as a situation where Π_p occludes Π_q along all rays L between L_p and L_q, which can simply be tested as $\|X_p\| < \|\Pi_q \cap L_p\|$ and $\|X_q\| > \|\Pi_p \cap L_q\|$. However, it raises yet another issue: there are cases where $\|X_p\| < \|X_q\|$, thus $p \prec^0 q$, and yet $\|\Pi_p \cap L\| > \|\Pi_q \cap L\|$ for all L between L_p and L_q, implying the inverse occlusion $p \succ^1 q$, cf. Figure 4.2(h). This small-step configuration exists ubiquitously (e.g., book on a table, frame on a wall) but does not correspond to an actual occlusion. To prevent this paradoxical situation and also to introduce some robustness, as normals can be wrong due to estimation errors, we actually define order-1 occlusion so that it also implies order-0 occlusion. In the end, we say that *p occlude q at order 1* iff (i) p occludes q at order 0, (ii) Π_p occludes Π_q along all rays L between L_p and L_q, i.e.,

$$p \prec^1 q \;\text{ iff }\; \|X_p\| < \|X_q\| \;\wedge\; \|X_p\| < \|\Pi_q \cap L_p\| \;\wedge\; \|X_q\| > \|\Pi_p \cap L_q\|\,. \tag{4.2}$$

4.3.3 Discretized occlusion

In practice, we resort to a discrete formulation where p, q are neighboring pixels in image $\mathcal{P}$ and L_p passes through the center of p. We note $\mathcal{N}_p$ the immediate neighbors of p, considering either only the 4 horizontal and vertical neighbors $\mathcal{N}_p^4$, or including also in $\mathcal{N}_p^8$ the 4 diagonal pixels.

As distances (depths) $d_p = \|X_p\|$ can only be measured approximately, we require a minimum discontinuity threshold $\delta > 0$ to test any depth difference. A condition



(a) oriented occlusion boundary (red curve, fg-on-left convention) and boundary rasterization

(b) oriented occlusion boundary with per-pixel orientation information (from fg to bg)

(c) P2ORM: pixel-pair occlusion relation (arrows from occluder to occludee, one color per orientation)

(d) oriented boundary based on annotated line segments (green), yielding a different rasterization

(e) line-segment-based boundary pixels oriented from line orient. (alt. orientation representation)

(f) line-segment-based boundary pixels orient. as average direction of (here) bg neighbors (mauve)

(g) segmentation into layers with depth ordering, occlusion at relative fg/bg pixel boundary

(h) fg/bg occlusion relationship: with all fg (yellow) $\rightarrow$ bg (blue) arrows between neighbor pixels

(i) oriented border pixels (here fg) from fg boundary pixel to average direction of all bg neighbors

Figure 4.3: Some representations of occlusion and oriented occlusion.

$d_p < d_q$ thus translates as $d_q - d_p \geq \delta$. However, to treat equally all pairs of neighboring pixels p, q, the margin δ has to be relative to the pixel distance $\|p-q\|$, which can be 1 or $\sqrt{2}$ due to the diagonal neighbors. Extending the first-order approximation, the relation $d_p < d_q$ is thus actually tested as $d_{pq} > \delta$ where $d_{pq} \stackrel{\text{def}}{=} (d_q - d_p)/\|q-p\|$, making δ a pixel-wise depth increasing rate.

4.3.4 Occlusion relationship and occlusion boundary

Most of the literature on occlusion in images focuses on *occlusion boundaries*, that are imaginary lines separating locally a foreground (fg) from a background (bg). A problem is that they are often materialized as rasterized, 1-pixel-wide contours, that are not well defined, cf. Figure 4.3(a). The fact is that vectorized occlusion

delineations are not generally available in existing datasets, except for handmade annotations, that are coarse as they are made with line segments, with endpoints at discrete positions, only approximating the actual, ideal curve, cf. Figure 4.3(d). An alternative representation [116, 47, 141] considers occlusion boundaries at the border pixels of two relative fg/bg segments (regions) rather than on a separating line (Figure 4.3(g)).

Inspired by this pixel-border representation but departing from the notion of fg/bg segments, we model occlusion at pixel-level between a fg and a bg pixel, yielding *pixel-pair occclusion relationship maps (P2ORM)* at image level, cf. Figure 4.3(c). An important advantage is that it allows the generation of relatively reliable occlusion information from depth maps, cf. Equation (4.2), assuming the depth maps are accurate enough, e.g., generated from synthetic scenes or obtained by high-end depth sensors. Together with photometric data, this occlusion information can then be used as ground truth to train an occlusion relationship estimator from images (see Section 4.4). Besides, it can model more occlusion configurations, i.e., when a pixel is both occluder and occludee (of different neighbor pixels).

Still, to enable the comparison with existing methods, we provide a way to construct traditional boundaries from P2ORM. Boundary-based methods represent occlusion as a mask $(\omega_p)_{p\in\mathcal{P}}$ such that $\omega_p = 1$ if pixel p is on an occlusion boundary, and $\omega_p = 0$ otherwise, with associated predicate $\dot{\omega}_p \stackrel{\text{def}}{=} (\omega_p = 1)$. We say that a pixel p *is on an occlusion boundary*, noted $\dot{\omega}_p$, iff it is an occluder or occludee:

$$\dot{\omega}_p \;\text{ iff }\; \exists q \in \mathcal{N}_p,\; p \prec q \;\vee\; p \succ q. \tag{4.3}$$

This defines a 2-pixel-wide boundary, illustrated as the grey region in Figure 4.3(c). As we actually estimate occlusion probabilities rather than certain occlusions, this width may be thinned by thresholding or non-maximum suppression (NMS).

4.3.5 Occlusion relationship and oriented occlusion boundary

Related to the notions of segment-level occlusion relationship, figure/ground representation and boundary ownership [116, 141], occlusion boundaries may be oriented to indicate which side is fg vs bg, cf. Figure 4.3(b). It is generally modeled as the direction of the tangent to the boundary, conventionally oriented [47] (fg on the left, Figure 4.3(a)). In practice, the boundary is modeled with line segments (Figure 4.3(d)), whose orientation θ is transferred to their rasterized pixels [141] (Figure 4.3(e)). Inaccuracies matter little here as the angle is only used to identify a boundary side.

The occlusion border formulation, based on fg/bg pixels (Figure 4.3(g)), implicitly captures orientation information: from each fg pixel to each neighbor bg pixel (Figure 4.3(h)). So does our modeling (Figure 4.3(c)). To compare with boundary-based approaches, we define a notion of pixel occlusion orientation (that could apply to occlusion borders too (Figure 4.3(i)), or even boundaries (Figure 4.3(f)). We say that a pixel p is oriented as the sum v_p of the unitary directions of occluded or occluding neighboring pixels q, with angle $\theta_p = \text{atan2}(u_p^y, u_p^x) - \frac{\pi}{2}$ where $u_p = v_p / \|v_p\|$ and

$$v_p = \sum_{q \in \mathcal{N}_p} (\mathbb{1}(p \prec q) - \mathbb{1}(p \succ q)) \frac{q - p}{\|q - p\|}. \tag{4.4}$$

4.4 Pixel-Pair Occlusion Relationship Estimation

4.4.1 Modeling

The occlusion relation is a binary property that is antisymmetric: $p \prec q \Rightarrow q \not\prec p$. Hence, to model the occlusion relationship of neighbor pair pq, we use a random variable $\omega_{p,q}$ with only three possible values $r \in \{-1, 0, 1\}$ representing respectively: $p \succ q$ (p is occluded by q), $p \not\prec q \wedge p \not\succ q$ (no occlusion between p and q), and $p \prec q$ (p occludes q).

Since $\omega_{p,q} = -\omega_{q,p}$, a single variable per pair is enough. We assume a fixed total ordering $<$ on pixels (e.g., lexicographic order on image coordinates) and note $\omega_{pq} = \omega_{p,q}$ if $p < q$. We also define the probability for each possible value: $\omega_{pqr} = \mathbb{P}(\omega_{pq} = r)$.

Concretely, we consider 4 inclinations, horizontal, vertical, diagonal, antidiagonal, with canonical displacements $\mathsf{h} = (1, 0)$, $\mathsf{v} = (0, 1)$, $\mathsf{d} = (1, 1)$, $\mathsf{a} = (1, -1)$, and we call $\mathcal{Q}_i = \{pq \mid p, q \in \mathcal{P}, q = p + i\}$ the set of pixel pairs with inclination $i \in \mathcal{I}^4 = \{\mathsf{h}, \mathsf{v}, \mathsf{d}, \mathsf{a}\}$. For the 4-connectivity, we only consider $i \in \mathcal{I}^2 = \{\mathsf{h}, \mathsf{v}\}$.

4.4.1.1 Estimating the occlusion relation

For occlusion relationship estimation, we adopt a segmentation approach: we classify each valid pixel pair pq by scoring its 3 possible statuses $r \in \{-1, 0, 1\}$, from which we extract estimated probabilities $\hat{\omega}_{pqr}$. The estimated occlusion status is obtained as $\hat{\omega}_{pq} = \operatorname{argmax}_r \hat{\omega}_{pqr}$.

Our architecture is sketched in Figure 4.4 (left). The P2ORM estimator (named *P2ORNet*) takes an RGB image as input, and outputs its pixel-pair occlusion relationship map for the different inclinations. We use a ResNet-based [39] U-Net-like auto-encoder with skip-connections [119]. It must be noted that this architecture is strikingly simple compared to more complex problem-specific architectures that have been proposed in the past [141, 138, 81]. Besides, our approach is not specifically bound to U-Net or ResNet; in the future, we may benefit from improvements in general segmentation methods.

4.4.1.2 From occlusion relations to occlusion boundaries.

As discussed with Equation (4.3), occlusion boundaries can be generated from an occlusion relation. In case the occlusion relation is available with probabilities (i.e., $\{\hat{\omega}_{pqr} \mid r \in \{-1, 0, 1\}\}$), we define the probabilistic variant of being on an occlusion boundary for a pixel p as $\omega_p \in [0, 1]$. The estimated probability $\hat{\omega}_p$ is an average considering the pairs between the pixel p and its neighbor pixels: $\hat{\omega}_p = \frac{1}{|\mathcal{N}_p|} \sum_{q \in \mathcal{N}_p} (\hat{\omega}_{p,q,-1} +$

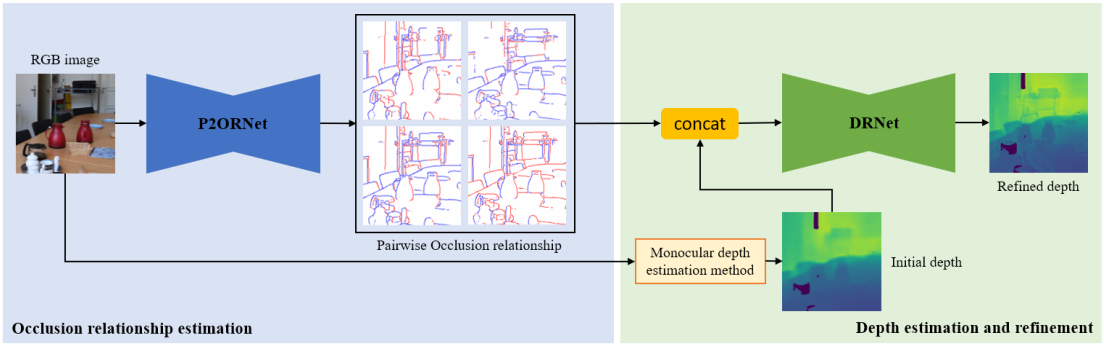


Figure 4.4: Overview of our method. Left: a encoder-decoder structure followed by softmax takes an RGB image as input and outputs 4 classification maps (ω_p^i) where each pixel p in a map for inclination i actually represents a pair of pixels pq with $q = p + i$. The map $\omega_{pq}^i = \omega_p^i = r$ classifies p as occluded ($r = -1$), not involved in occlusion ($r = 0$) or occluding ($r = 1$), with probability ω_{pqr}^i. (If $\mathcal{N} = \mathcal{N}^4$, only 2 inclination maps are generated.) Colors blue, white and red represent respectively $r = -1$, 0 or 1. The top two images presents occlusion relationships along inclinations horizontal ($i = \mathsf{h}$) and vertical ($i = \mathsf{v}$); the bottom two, along inclinations diagonal ($i = \mathsf{d}$) and antidiagonal ($i = \mathsf{a}$). Right: A direct use of the occlusion relationship for depth map refinement.

$\hat{\omega}_{p,q,1}$). Here, $\hat{\omega}_{p,q,r} = \mathbb{P}(\hat{\omega}_{p,q} = r)$ and it can be calculated from $\hat{\omega}_{pqr}$.

As proposed in [23] and performed in many other methods, we operate a non-maximum suppression to get thinner boundaries. The final occlusion boundary map is given by thresholding $\mathrm{NMS}((\omega_p)_{p\in\mathcal{P}})$ with a probability, e.g., 0.5.

Boundary orientations can then be generated as defined in Equation (4.4). Given our representation, it has the following simpler formulation: $v_p = \sum_{q\in\mathcal{N}_p} \hat{\omega}_{p,q} \frac{q-p}{\|q-p\|}$.

4.4.2 Training procedure

We train our model with a *class-balanced cross-entropy loss* [153], taking into account the low probability for a pair pq to be labeled 1 (p occludes q) or -1 (q occludes p), given that most pixel pairs do not feature any occlusion. Our global loss $\mathcal{L}_{\mathsf{occrel}}$ is a sum of $|\mathcal{I}|$ losses for each kind of pair inclination $i \in \mathcal{I}$, averaged over the number of

pairs $|\mathcal{Q}_i|$ to balance each task $i \in \mathcal{I}$:

$$\mathcal{L}_{\text{occrel}} = \sum_{i \in \mathcal{I}} \frac{1}{|\mathcal{Q}_i|} \sum_{\substack{pq \in \mathcal{Q}_i \\ r \in \{-1,0,1\}}} -\alpha_r \, \omega_{pqr} \log(\hat{\omega}_{pqr}). \tag{4.5}$$

where $\hat{\omega}_{pqr}$ is the estimated probability that pair pq has occlusion status r, $\omega_{pqr} = \mathbb{1}(\omega_{pq} = r)$ where ω_{pq} is the ground-truth occlusion status of pair pq, $\alpha_r = \mathbb{1}(r = 0) + \alpha \mathbb{1}(r \neq 0)$ and α accounts for the disparity in label frequency. In the early stage of our research, we also tried regularization terms considering the geometric constraints of physical world (e.g., one pixel can not be occluded by all its neighbor pixels). However, our experiments shows that the involvement of these regularization terms slows down the learning of our models and does not contribute to a performance boost. Thus, we only use class-balanced cross-entropy loss [153] during training.

4.5 Depth Map Refinement with Occlusion Relationship

Given an image, a depth map $(\tilde{d}_p)_{p \in \mathcal{P}}$ estimated by some method, and an occlusion relationship $(\hat{\omega}_{p,p+i})_{p \in \mathcal{P}, i \in \mathcal{I}}$ as estimated in Section 4.4, we produce a refined, more accurate depth map $(\hat{d}_p)_{p \in \mathcal{P}}$ with sharper edges. To this end, we propose a U-Net architecture [119] (Figure 4.4 (right)), named DRNet, where $(\tilde{d}_p)_{p \in \mathcal{P}}$ and the 8 maps $((\hat{\omega}_{p,p+i})_{p \in \mathcal{P}})_{i \in \mathcal{I} \cup (-\mathcal{I})}$ are stacked as a multi-channel input of the network.

As a pre-processing, we first use the ground-truth depth map $(d_p)_{p \in \mathcal{P}}$ and normals $(\mathbf{n}_p)_{p \in \mathcal{P}}$ to compute the ground-truth occlusion relationship $(p \prec_{\text{gt}} q)_{p \in \mathcal{P}, q \in \mathcal{N}_p}$. We

then train the network via the following loss:

$$\mathcal{L}_{\mathsf{refine}} = \mathcal{L}_{\mathsf{occonsist}} + \lambda \mathcal{L}_{\mathsf{regul}} \tag{4.6}$$

$$\mathcal{L}_{\mathsf{occonsist}} = \frac{1}{N} \sum_{p \in \mathcal{P}} \sum_{q \in \mathcal{N}_p^8} \begin{cases} \mathcal{B}(\log \delta, \log \hat{d}_{pq}) & \text{if } \; p \prec_{\mathsf{gt}} q \text{ and } \hat{d}_{pq} < \delta \\ \mathcal{B}(\log \delta, \log \hat{D}_{pq}) & \text{if } \; p \not\prec_{\mathsf{gt}} q \text{ and } \hat{D}_{pq} \geq \delta \\ 0 & \text{otherwise} \end{cases} \tag{4.7}$$

$$\mathcal{L}_{\mathsf{regul}} = \frac{1}{|\mathcal{P}|} \sum_{p \in \mathcal{P}} \left(\mathcal{B}(\log \tilde{d}_p, \log \hat{d}_p) + \left\| \nabla \log \tilde{d}_p - \nabla \log \hat{d}_p \right\|^2 \right) \tag{4.8}$$

where $\mathcal{B}$ is the berHu loss [61], δ is the depth discontinuity threshold introduced in Section 4.3, N is the number of pixels p having a non-zero contribution to $\mathcal{L}_{\mathsf{occonsist}}$, and $\hat{D}_{pq}$ is the order-1 depth difference at mid-pixel $(p+q)/2$, i.e., $\hat{D}_{pq} = \min(\hat{d}_{pq}, \hat{m}_{pq})$ where $\hat{m}_{pq} = (\left\| \hat{\Pi}_q \cap L_{(p+q)/2} \right\| - \left\| \hat{\Pi}_p \cap L_{(p+q)/2} \right\|) / \left\| q - p \right\|$ is the signed distance between predicted tangent planes $\hat{\Pi}_p, \hat{\Pi}_q$ (using $\hat{d}_p, \hat{d}_q$ and $\mathbf{n}_p, \mathbf{n}_q$) along $L_{(p+q)/2}$.

$\mathcal{L}_{\mathsf{occonsist}}$ penalizes refined depths d_p that are inconsistent with ground-truth occlusion relationship $\prec_{\mathsf{gt}}$, i.e., when p occludes q in the GT but not in the refinement, or when p does not occlude q in the ground truth but does it in the refinement. $\mathcal{L}_{\mathsf{regul}}$ penalizes differences between the rough input depth $(\tilde{d}_p)_{p \in \mathcal{P}}$ and the refined output depth $(\hat{d}_p)_{p \in \mathcal{P}}$, which makes refined depths conditioned on input depths. The total loss $\mathcal{L}_{\mathsf{refine}}$ tends to change depths only close to occlusion boundaries, preventing excessive drifts.

To provide occlusion information that has the same size as the depth map, as pixel-pair information is not perfectly aligned on the pixel grid, we turn pixel-pair data $(\omega_{p,p+i})_{p \in \mathcal{P}, i \in \mathcal{I}, p+i \in \mathcal{P}}$ into a pixel-wise information: for a given inclination $i \in \mathcal{I}$, we define $\omega_p^i = \omega_{p,p+i}$. Thus, e.g., if $p \prec p+i$, then $\omega_p^i = 1$ and $\omega_{p+i}^i = -1$.

At test time, given the estimated occlusion relationships, we use NMS to sharpen depth edges. For this, we first generate pixel-wise occlusion boundaries from the estimated P2ORM $(\hat{\omega}_{p,p+i})_{p \in \mathcal{P}, i \in \mathcal{I}}$, pass them through NMS [23] and do thresholding

to get a binary occlusion boundary map $(\hat{\omega}_p)_{p\in\mathcal{P}}$ where $\hat{\omega}_p \in \{0,1\}$. We then thin the estimated directional maps $(\hat{\omega}^i_p)_{p\in\mathcal{P}}$ by setting $\hat{\omega}^i_p \leftarrow 0$ if $\hat{\omega}_p = 0$.

4.6 Accurate Depth Estimation on Boundaries

In section 4.5, our model predicts the refined depths conditioned on the input depths with the help of P2ORM. As P2ORM features directed depth discontinuities, another natural thinking is to use the generated P2ORM annotations as an additional supervision signal in the learning of monocular depth estimation methods to make models pay more attention to the depth boundaries regions. Instead of adopting a multi-task learning strategy, we propose to use the estimated P2ORM conditioned on the predicted depths to evaluate the quality of predicted depths and guide our models towards better depth predictions.

4.6.1 Modeling

We first make the generation of P2ORM differentiable. In other words, we train a deep network G_{d2o} which predicts pairwise occlusion relationship directly from the depth map d with the supervision of P2ORM ground truth. Concretely, G_{d2o} adopts a ResNet-based [39] U-Net-like auto-encoder with skip-connections [119] as the network architecture. It takes the ground-truth depth map d as input and classifies each valid pixel pair pq by scoring its 3 possible statuses $r \in \{-1,0,1\}$, from which we extract the estimated probabilities $\hat{\omega}_{pqr}$. The final classification map is obtained as $\hat{\omega}_{pq} = \text{argmax}_r\, \hat{\omega}_{pqr}$. By this way, the mapping from scene geometry information to occlusion relationship is learned by G_{d2o} and can be used in an end-to-end learning scheme. Once the learning converges, we fix all parameters of G_{d2o} and use it in our monocular depth estimation approach.

For monocular depth estimation, we adopt a DenseNet-based [50] U-Net-like encoder-decoder G_{i2d} to predict metric depth $\hat{d}$ from each image with the super-

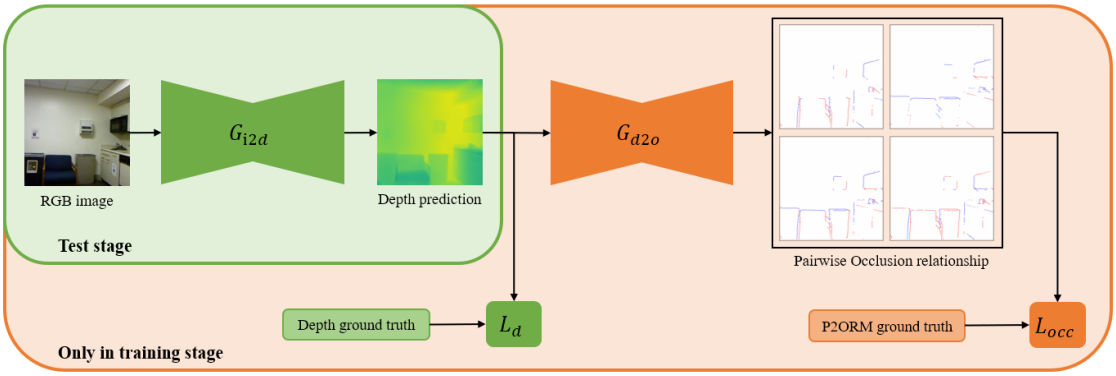


Figure 4.5: Overview of our monocular depth estimation method with the guidance of P2ORM

vision of ground-truth depth d. Then the predicted depth $\hat{d}$ is fed into G_{d2o} as input to estimate the pixel-pair occlusion relationship map $\hat{\omega}_{pq}$ with the supervision of P2ORM ground truth ω_{pq}. As mentioned before, the mapping from depth map to occlusion is learnt by G_{d2o} and the parameters of it are fixed during the learning of depth estimation. Thus, the predicted $P2ORM$ by G_{d2o} in fact reflects the quality of depth prediction $\hat{d}$ by G_{i2d} about depth discontinuities. As G_{d2o} is differentiable for back-propagation, the supervision about occlusion will let our depth estimator G_{i2d} pay more attention to the regions with depth discontinuities during training. By this way, our single-stage depth estimator G_{i2o} can recovers high-quality depth discontinuities without an additional depth refiner. More importantly, our method is independent of the choice of G_{i2d} architecture so that more powerful choice of G_{i2d} can be used with the advancement of research in the community. As shown in Figure 4.5, we train our monocular depth estimator G_{i2d} with the supervision of d and of P2ORM where our occlusion estimator G_{d2o} takes the predicted depth map $\hat{d}$ as input. During test, we only evaluate our monocular depth estimator G_{i2d} on benchmarks.

4.6.2 Training procedure

For the learning of the mapping from depth map to P2ORM, similar to Section 4.4, we train our model G_{d2o} with a *class-balanced cross-entropy loss* [153] $\mathcal{L}_{\text{occrel}}$ as in Equation (4.5), taking into account the low probability for a pair pq to be labeled 1 (p occludes q) or -1 (q occludes p).

For the learning of the mapping from color image to depth map, we train our model G_{i2d} with $\mathcal{L}_{\text{occrel}}$ mentioned before and loss terms between the predicted depth map $\hat{d}$ and the ground-truth depth map d. Regarding depth map as a set of pixels $\mathcal{P}$, the total loss $\mathcal{L}_{depth}$ for backpropagation is as follows:

$$\mathcal{L}_{\text{depth}} = \lambda_d \mathcal{L}_1 + \mathcal{L}_{\text{grad}} + \mathcal{L}_{\text{SSIM}} + \lambda_o \mathcal{L}_{\text{occrel}} \tag{4.9}$$

$$\mathcal{L}_1 = \frac{1}{N} \sum_{p \in \mathcal{P}} |d_p - \hat{d}_p| \tag{4.10}$$

$$\mathcal{L}_{\text{grad}} = \frac{1}{N} \sum_{p \in \mathcal{P}} |\nabla_x d_p - \nabla_x \hat{d}_p)| + |\nabla_y d_p - \nabla_y \hat{d}_p| \tag{4.11}$$

$$\mathcal{L}_{\text{SSIM}} = \frac{1 - SSIM(d, \hat{d})}{2}, \tag{4.12}$$

where $\mathcal{L}_1$ is the pixel-wise L1 loss, $\mathcal{L}_{\text{grad}}$ is the L1 loss between depth map gradients and $\mathcal{L}_{\text{SSIM}}$ is the Structural Similarity term [144]. Since SSIM has an upper bound of one, we define $\mathcal{L}_{\text{SSIM}}$ similar to [2]. In experiments, we set $\lambda_d = 0.1$ and $\lambda_o = 0.1$.

4.7 Implementation Details

Occlusion ground truth generation from depth acquired by laser scanner. In real datasets such as iBims-1 [58] whose depths are captured by laser scanners, the depth estimation noise varies with both actual depth, pixel spatial location and surface orientation due to laser scanner hardware properties. Therefore we propose the following formula (cf. Equation (4.13)) to calculate possible the depth estimation

error E_p relevant to a pixel p considering aforementioned factors:

$$E_p = \frac{\eta}{\tan(\gamma)} d_p \,, \tag{4.13}$$

where η is a constant representing laser scanner estimation angular noise, γ is the angle between ray L_p and tangent plane Π_p, d_p is the Euclidean distance between surface point X_p and camera center C and it's estimated by laser scanner. Then the discontinuity threshold δ as introduced in Section 4.3 between a pixel-pair p, q can be calculated with E_p, E_q and a constant C_δ that ensures a minimum discontinuity (cf. Equation (4.14)):

$$\delta = E_p + E_q + C_\delta \,. \tag{4.14}$$

As shown in Figure 4.7 in Section 4.8, by using the proposed occlusion definition and the proposed dynamic discontinuity threshold in Equation (4.14) for each pixel-pair, the generated occlusion ground truths are accurate in the scene with a large depth range. Specifically, for iBims-1 dataset, $\eta = 0.005\,rad$ in Equation (4.13) and $C_\delta = 25\,mm$ in Equation (4.14).

Occlusion relationship estimation. In Figure 4.4, we sketch the architecture of our network for pixel-pair occlusion relationship estimation (P2ORNet). Here we provide additional information. The detailed architecture is depicted in Figure 4.6, while the setup of each block is presented in Table 4.1. Blocks named as "Res" are residual convolution blocks introduced in [39] and blocks named as "Deconv" are transposed convolution layers.

We initialize the encoder model of the occlusion estimation module with the weights of a ResNet-50 model [39] pre-trained on ImageNet, and the remaining layers with random values (as defined by the PyTorch [100] default initialization). To train the network, we use the ADAM optimizer [57] with learning rate 10^{-4} and divide

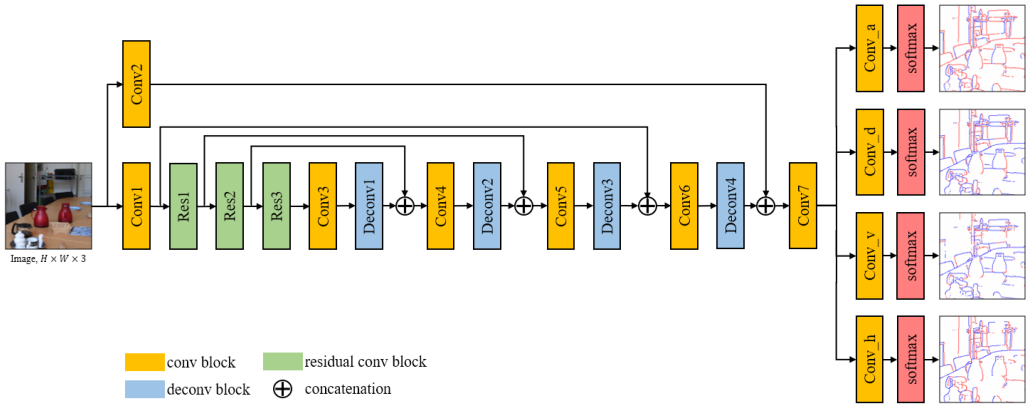


Figure 4.6: Architecture of P2ORNet, for occlusion relationship estimation.

Conv1	Conv2	Res1	Res2	Res3	Conv3
$[7\times7, 64]\times1$ stride 2	$\begin{bmatrix} 3\times3, 64 \\ 3\times3, 64 \end{bmatrix}\times1$	3×3 maxpool, stride 2 $\begin{bmatrix} 1\times1, 64 \\ 3\times3, 64 \\ 1\times1, 256 \end{bmatrix}\times3$	$\begin{bmatrix} 1\times1, 128 \\ 3\times3, 128 \\ 1\times1, 512 \end{bmatrix}\times4$	$\begin{bmatrix} 1\times1, 256 \\ 3\times3, 256 \\ 1\times1, 1024 \end{bmatrix}\times6$	$\begin{bmatrix} 3\times3, 512 \\ 3\times3, 512 \end{bmatrix}\times1$
Deconv1	Conv4	Deconv2	Conv5	Deconv3	Conv6
$[3\times3, 512]\times1$ stride 2	$\begin{bmatrix} 3\times3, 256 \\ 3\times3, 256 \end{bmatrix}\times1$	$[3\times3, 256]\times1$ stride 2	$\begin{bmatrix} 3\times3, 64 \\ 3\times3, 64 \end{bmatrix}\times1$	$[3\times3, 64]\times1$ stride 2	$\begin{bmatrix} 3\times3, 64 \\ 3\times3, 64 \end{bmatrix}\times1$
Deconv4	Conv7	Conv_h	Conv_v	Conv_d	Conv_a
$[3\times3, 64]\times1$ stride 2	$\begin{bmatrix} 3\times3, 64 \\ 3\times3, 64 \end{bmatrix}\times1$	$[1\times1, 3]\times1$	$[1\times1, 3]\times1$	$[1\times1, 3]\times1$	$[1\times1, 3]\times1$

Table 4.1: Detailed architecture of P2ORNet (pixel-pair occlusion relation estimation). For each block, as in [39], we give the kernel size and the number of output channels. The blocks Conv1, Res1, Res2, Res3 are the first four blocks of ResNet-50 [39].

it by 10 when half of the total training iterations (4000, 100000, 110000 for BSDS, NYUv2-OR, iBims-1-OR respectively) is reached. The input image size during training is 320×320, and the mini-batch size is 8.

Depth refinement. We initialize our network layers using the 'kaiming' initialization as in [39] and train the network from scratch. The training set contains 10k depth predictions of SharpNet [112] on the InteriorNet subset [70] that we consider, and corresponding ground truth P2ORM. We use the ADAM optimizer [57] with a fixed learning rate of 10^{-5} and stop the training after 40k iterations. The size of the input depth images size is 640×480 and the batch size is 8 for all experiments.

Accurate Depth Estimation on Boundries. We initialize the encoder model of the occlusion estimation network G_{d2o} with the weights of a ResNet-50 model [39] pre-trained on ImageNet, and the remaining layers with random values (as defined by the PyTorch [100] default initialization). To train the network, we use the ADAM optimizer [57] with a learning rate 10^{-4} for 50000 iterations and divide it by 10 for the remaining 50000 iterations. The input image size during training is 320×320, and the mini-batch size is 8. For the monocular depth estimation network G_{i2d}, we initialize the encoder model with the weights of a DenseNet-169 model [50] pre-trained on ImageNet and randomize the weights of remaining layers as what we do on G_{d2o}. To train the model, we first set $\lambda_o = 0$ in Equation (4.12) and use the ADAM optimizer [57] with a learning rate 10^{-4} for 250000 iterations. Then we set $\lambda_o = 0.1$ and train the model for the remaining 100000 iterations. The input image size during training is 640×480, and the mini-batch size is 4.

4.8 Experimental Evaluation

4.8.1 Datasets

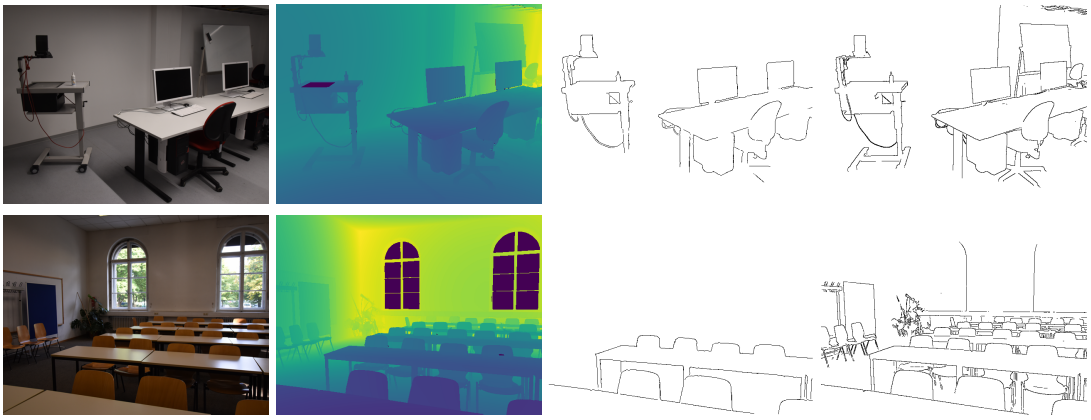
Evaluation datasets. We evaluate on 3 datasets: BSDS ownership [116], NYUv2-OR, iBims-1-OR (cf. Table 4.2). We keep the original training and testing data of BSDS. NYUv2-OR picks the test set of NYUv2 [93] and add occlusion boundaries from [111] and our generated occlusion labels. IBims-1-OR augments iBims-1 [58] with the occlusion ground truth that we generated automatically (cf. Section 4.3). As illustrated in Figure 4.7, this new accurate ground truth is much more complete than the "distinct depth transitions" offered by iBims-1 [58], that are first detected on depth maps with [23], then manually selected. For training, a subset of InteriorNet [70] (namely InteriorNet-OR) is used for NYUv2-OR and iBims-1-OR. For NYUv2-OR, because of the domain gap between sharp InteriorNet images and blurry NYUv2

Table 4.2: Used and created occlusion datasets. (a) We only use 500 scenes and 20 images per scene (not all 500M images). (b) Training on NYUv2-OR uses all InteriorNet-OR images adapted using [161] with the 795 training images of NYUv2 as target domain. (c) Training on iBims-1-OR uses all InteriorNet-OR images w/o domain adaptation.

Dataset	InteriorNet-OR	BSDS ownership	NYUv2-OR	iBim-1-OR
Origin	[70]	[116]	[93]	[58]
Type	synthetic	real	real	real
Scene	indoor	outdoor	indoor	indoor
Resolution	640×480	481×321	640×480	640×480
Depth	synthetic	N/A	Kinect v1	laser scanner
Normals	synthetic	N/A	N/A	computed [11]
Relation annot.	ours from depth and normals	ours from manual fig./ground [116]	ours from boundaries and depth	ours from depth and normals
Boundary annot.	from relation	manual [116]	manual [111]	from relation
Orient. annot.	from relation	manual [141]	manual (ours)	from relation
Annot. quality	high	low	medium	high
# train img. (orig.)	$10{,}000^{(a)}$	100	$795^{(b)}$	$0^{(c)}$
# train images	$10{,}000^{(a)}$	100	$10{,}000^{(b)}$	$10{,}000^{(c)}$
# testing images	0	100	654	100
α in $\mathcal{L}_{\text{occrel}}$	N/A	50	10	10

images, the InteriorNet-OR images are furthermore adapted with [161] using NYUv2 training images.

Samples of occlusion relationship in generated datasets. As described in introduction, we generated occlusion relationship annotations for three datasets, i.e., InteriorNet, iBims-1, NYUv2, and name the new datasets InteriorNet-OR, iBims-1-OR, NYUv2-OR. We illustrate here a few annotation samples: from InteriorNet-OR (cf. Figure 4.8), iBims-1-OR (cf. Figure 4.9) and NYUv2-OR (cf. Figure 4.10).



(a) (b) (c) (d)

Figure 4.7: iBims-1-OR: (a) RGB images, (b) GT depth (invalid is black), (c) provided "distinct depth transitions" [58], (d) our finer and more complete occlusion boundaries.

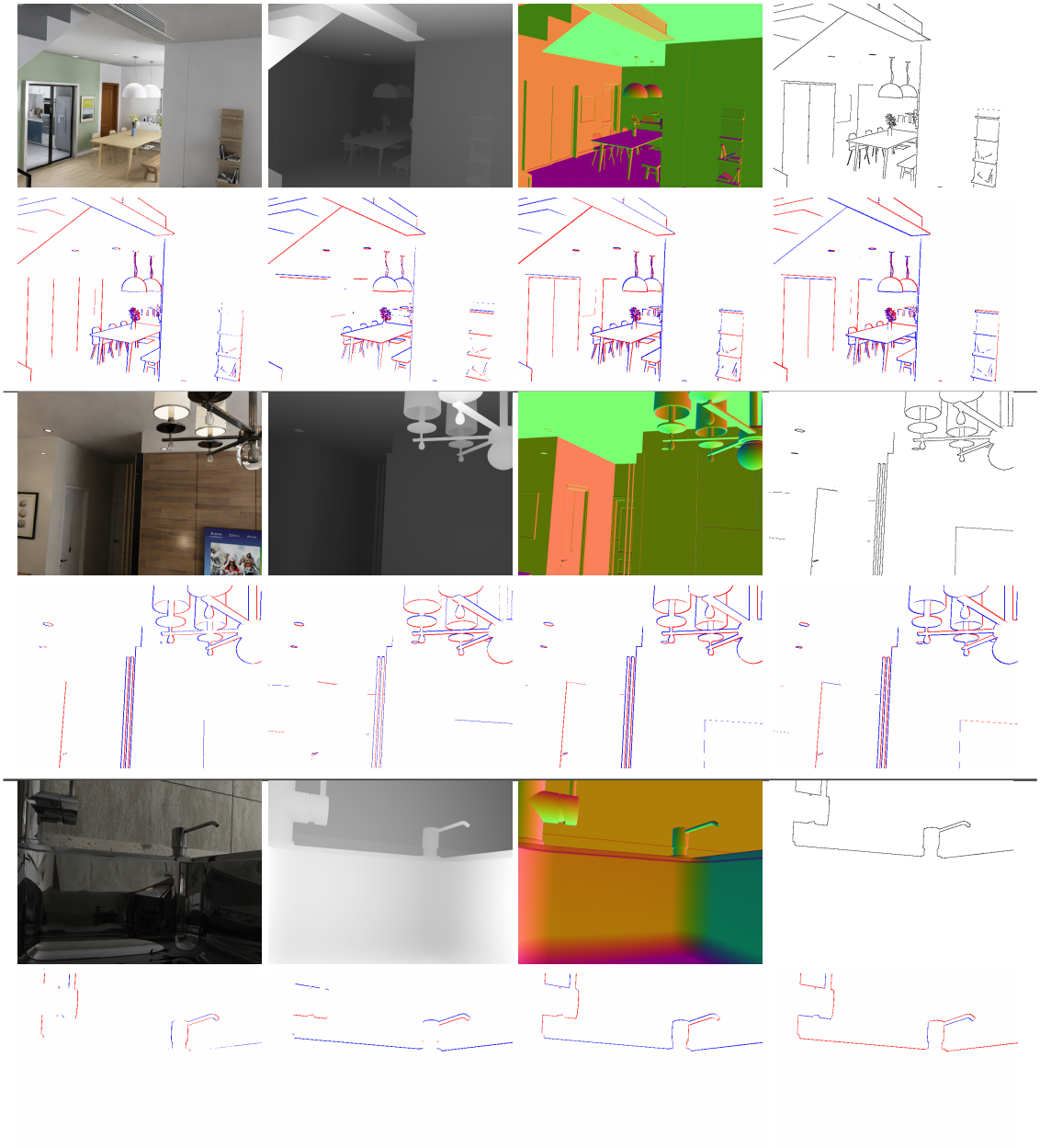


Figure 4.8: Samples from our InteriorNet-OR dataset. For each sample, first row, left to right: RGB image, depth map, normal map and generated occlusion boundaries; second row, left to right: generated occlusion relationships along inclinations horizontal ($i = \mathsf{h}$), vertical ($i = \mathsf{v}$), diagonal ($i = \mathsf{d}$) and antidiagonal ($i = \mathsf{a}$). Colors blue, white and red respectively represent pixel-pair occlusion status $r = -1$, 0 or 1.

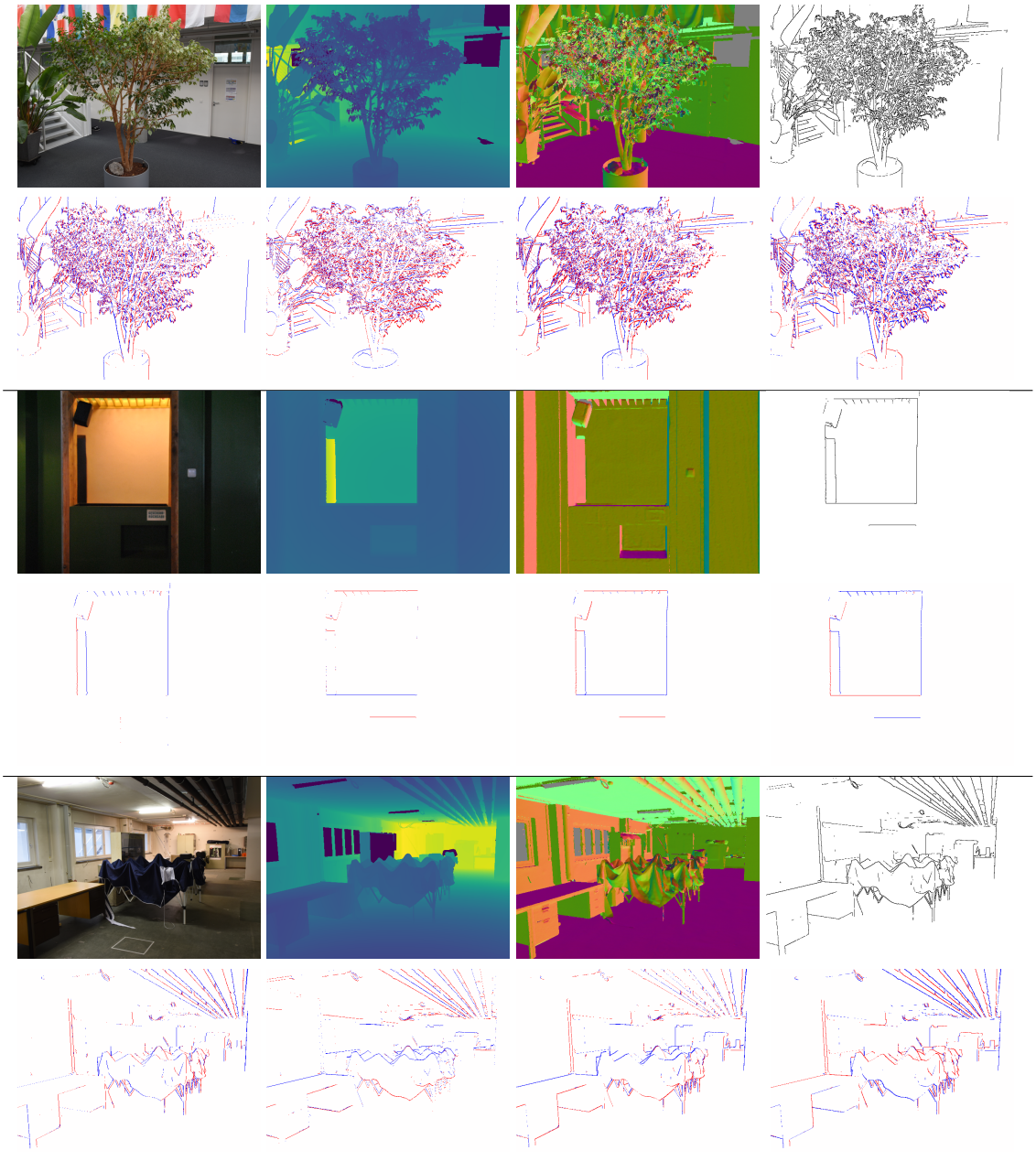


Figure 4.9: Samples from our iBims-1-OR dataset. For each sample, first row, left to right: RGB image, depth map, normal map and generated occlusion boundaries; second row, left to right: generated occlusion relationships along inclinations horizontal ($i = \mathsf{h}$), vertical ($i = \mathsf{v}$), diagonal ($i = \mathsf{d}$) and antidiagonal ($i = \mathsf{a}$). Colors blue, white and red respectively represent pixel-pair occlusion status $r = -1$, 0 or 1.

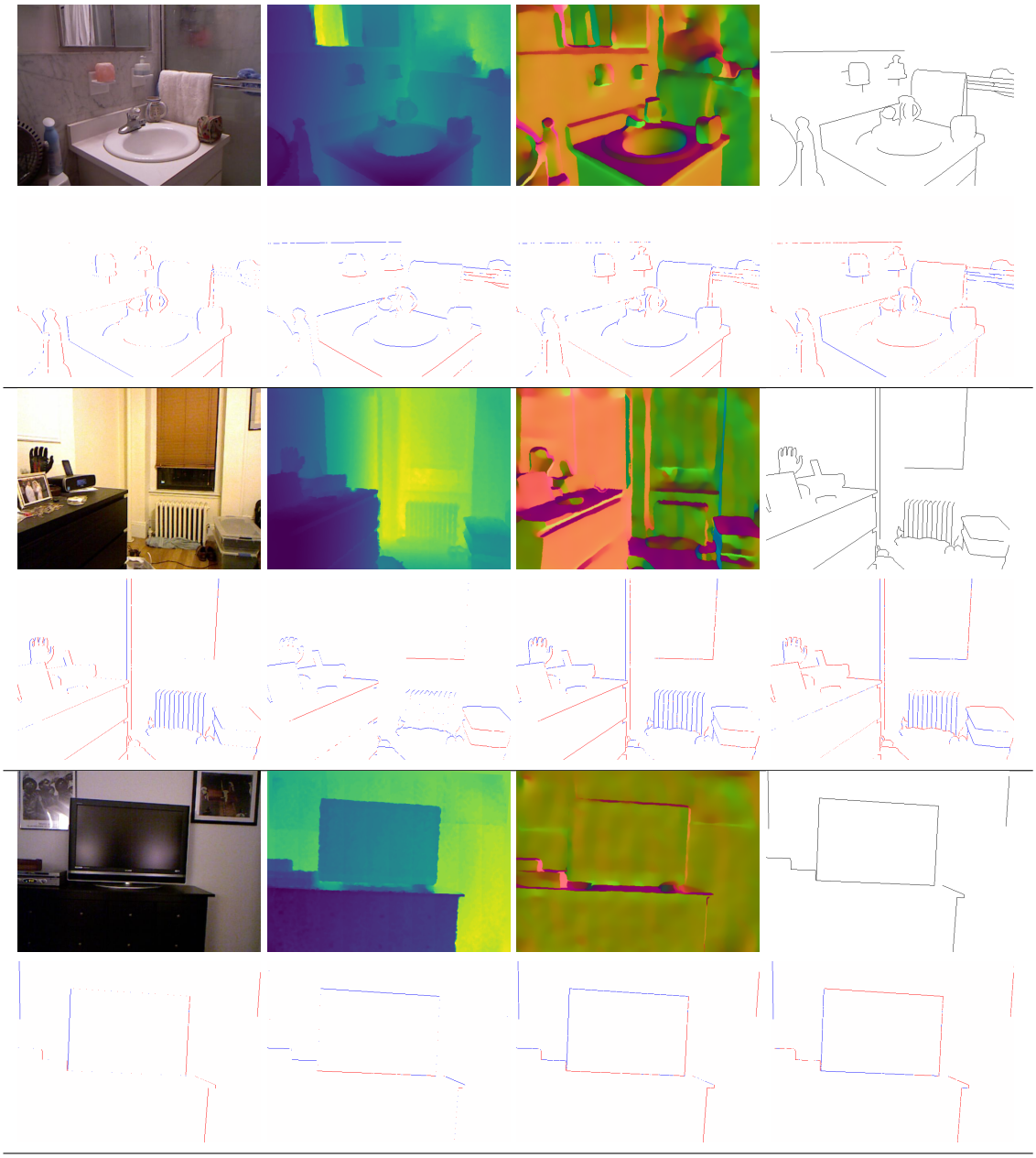


Figure 4.10: Samples from our NYUv2-OR dataset. For each sample, first row, left to right: RGB image, depth map and occlusion boundaries labeled by [111]; second row, left to right: generated occlusion relationships along inclinations horizontal ($i = \mathsf{h}$), vertical ($i = \mathsf{v}$), diagonal ($i = \mathsf{d}$) and antidiagonal ($i = \mathsf{a}$). Colors blue, white and red respectively represent pixel-pair occlusion status $r = -1$, 0 or 1.

4.8.2 Evaluation metrics

Oriented occlusion boundary metrics. We use the same protocol as [138, 81] to compute 3 standard evaluation metrics, based on the Occlusion-Precision-Recall graph (OPR): F-measure with best fixed occlusion probability threshold over the all dataset (ODS), F-measure with best occlusion probability threshold for each image (OIS), and average precision over all occlusion probability thresholds (AP). Recall (R) is the proportion of correct boundary detections, while Precision (P) is the proportion of pixels with correct occlusion orientation w.r.t. all pixels detected as occlusion boundary.

Depth map estimation metrics. We evaluate based on depth maps estimated by methods that offer results on depth-edge metrics: [26, 61, 30, 112, 53, 156] on NYUv2, and [26, 74, 68, 61, 112, 73] on iBims-1. We train our network on InteriorNet-OR for ground truth, with input depth maps to refine estimated by SharpNet [112]. For a fair comparison, we follow the evaluation protocol of [111]. To assess general depth accuracy, we measure: mean absolute relative error (rel), mean $\log_{10}$ error ($\log_{10}$), Root Mean Squared linear Error (RMSE(lin)), Root Mean Squared log Error (RMSE(log)), and accuracy under threshold $(\sigma_i < 1.25^i)_{i=1,2,3}$. For depth-edge, following [58], we measure the accuracy ϵ_{acc} and completion ϵ_{comp} of predicted boundaries.

4.8.3 Occlusion relationship estimation

4.8.3.1 Evaluation on oriented occlusion boundary

Because of the originality of our approach, there is no other method to directly compare with. Yet to demonstrate its significance in task-independent occlusion reasoning, we translate our relation maps into oriented occlusion boundaries (cf. Section 4.3) to compare with SRF-OCC [131], DOC-DMLFOV [141], DOC-HED [141], DOOB-

Net [138][1], OFNet [81][1].

To disentangle the respective contributions of the P2ORM formulation and the network architecture, we also evaluate a "baseline" variant of our architecture, that relies on the usual paradigm of estimating separately boundaries and orientations [141, 138, 81]: we replace the last layer of our pixel-pair classifier by two separate heads, one for classifying the boundary and the other one for regressing the orientation, and we use the same loss as [138, 81].

Table 4.3 summarizes quantitative results. Our baseline is on par with the state-of-the-art on the standard BSDS ownership benchmark as well as on the two new datasets, hinting that complex specific architectures maybe buy little as a common ResNet-based U-Net is at least as efficient. More importantly, our method with 8-connectivity outperforms existing methods on all metrics by a large margin (up to 15 points), demonstrating the significance of our formulation on higher-quality annotations, as opposed to BSDS whose lower quality levels up performances. It could also be an illustration that classification is often superior to regression [90] as it does not average ambiguities. Lastly, the 4-connectivity variant shows that the ablation of diagonal neighbors decreases the performance, thus assessing the relevance of 8-connectivity.

To allow a deeper assessment of the performance of our approach, compared to other state-of-the-art methods, we plot two graphs (cf. Figure 4.11):

(a) the Occlusion Accuracy w.r.t. boundary Recall (AOR) curve, as introduced in [141], represents accuracy as a function of recall;

(b) the Occlusion Precision w.r.t. boundary Recall (OPR) curve, as later proposed in [138], represents precision as a function of recall — a harder metric;

[1]As DOOBNet and OFNet are coded in Caffe, in order to have an unified platform for experimenting them on new datasets, we carefully re-implemented them in PyTorch (following the Caffe code). We could not reproduce exactly the same quantitative values provided in the original papers (ODS and OIS metrics are a bit less while AP is a bit better), probably due to some intrinsic differences between frameworks Caffe and PyTorch, however, the difference is very small (less than 0.03, cf. Table 4.3).

Table 4.3: Oriented occlusion boundary estimation. *Our re-implementation.

Method	BSDS ownership			NYUv2-OR			iBims-1-OR		
Metric	ODS	OIS	AP	ODS	OIS	AP	ODS	OIS	AP
SRF-OCC [131]	.419	.448	.337	-	-	-	-	-	-
DOC-DMLFOV [141]	.463	.491	.369	-	-	-	-	-	-
DOC-HED [141]	.522	.545	.428	-	-	-	-	-	-
DOOBNet [138]	.555	.570	.440	-	-	-	-	-	-
OFNet [81]	.583	.607	.501	-	-	-	-	-	-
DOOBNet*	.529	.543	.433	.343	.370	.263	.421	.440	312
OFNet*	.553	.577	.520	.402	.431	.342	.488	.513	.432
baseline	.571	.605	.524	.396	.428	.343	.482	.507	.431
ours (4-connectivity)	.590	.612	.512	.500	.522	.477	.575	.599	.508
ours (8-connectivity)	**.607**	**.632**	**.598**	**.520**	**.540**	**.497**	**.581**	**.603**	**.525**

where:

- (R)ecall is the proportion of pixels with correct boundary detections;
- (P)recision is the proportion of pixels with correct occlusion orientation w.r.t. all pixels detected as occlusion boundary;
- (A)ccuracy is the proportion of pixels with correct occlusion orientation w.r.t. all pixels *correctly* detected as occlusion boundary.

We compared all methods using the BSDS ownership dataset. This dataset has become a de facto standard benchmark regarding oriented occlusion boundary estimation, despite its moderate size and its coarse manually-annotated ground truth. We use exactly the same dataset (same training data and same test data) for all methods, including ours. On both AOR and OPR curves, we largely outperform all other existing methods, i.e., SRF-OCC [131], DOC-DMLFOV [141], DOC-HED [141], DOOBNet [138] and OFNet [81]. We show some qualitative results in Figure 4.12.

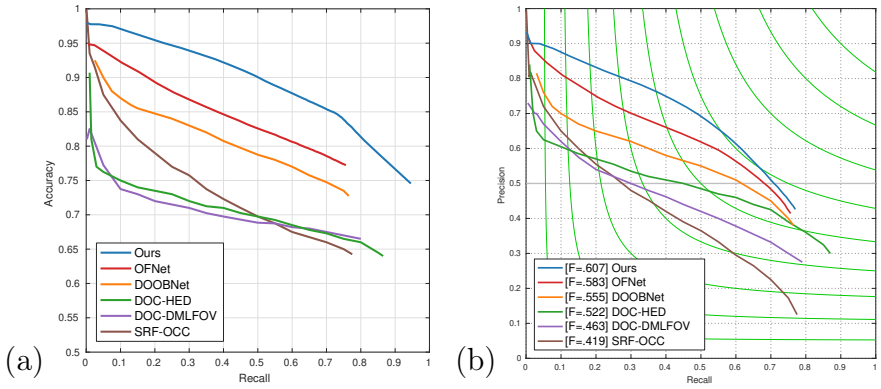


Figure 4.11: Oriented occlusion boundary estimation on BSDS ownership: (a) Occlusion-Accuracy-Recall curve (AOR) [141], (b) Occlusion-Precision-Recall curve (OPR) [138].

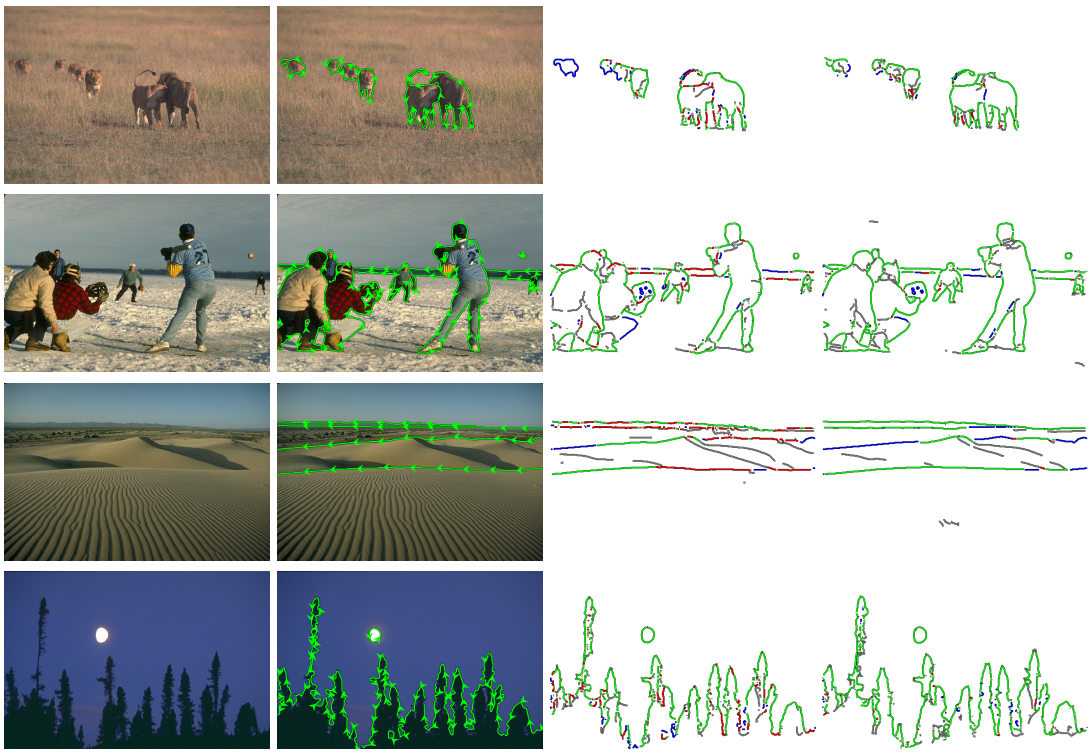


Figure 4.12: Occlusion estimation on BSDS ownership dataset: (a) input RGB image, (b) ground-truth occlusion orientation, (c) OFNet estimation [81], (d) our estimation. **green**: correct boundary and orientation; **red**: correct boundary, incorrect orientation; **blue**: missed boundaries; **gray**: incorrect boundaries.

NYUv2-OR (adapt. & test)	w/o adaptation			with adaptation			gain		
Method \ Metric	ODS	OIS	AP	ODS	OIS	AP	ODS	OIS	AP
DOOBNet*	.292	.324	.204	.343	.370	.263	.051	.046	.059
OFNet*	.339	.366	.255	.402	.431	.342	.063	.065	.087
baseline	.394	.418	.336	.396	.428	.343	.002	.010	.007
ours (4-connectivity)	.425	.446	.369	.500	.522	.477	.075	.076	.108
ours (8-connectivity)	.452	.477	.424	.520	.540	.497	.068	.063	.073

Table 4.4: Ablation study about domain adaptation using [161] with NYUv2-OR images as target for training on synthetic images of InteriorNet-OR and testing on NYUv2-OR. *Our re-implementation (cf. footnote in Section 4.8.3.1). In blue, the minimum gain; in red, the maximum gain.

4.8.3.2 Ablation study on training data

Ablation study on domain adaptation for synthetic images. To evaluate on NYUv2-OR and iBims-1-OR [58], we train on 10^4 synthetic images of InteriorNet-OR (cf. Table 4.2). The pictures in InteriorNet are not totally photorealistic, but still fairly good. In our experiments on iBims-1-OR, whose test images are of good quality, we train directly on InteriorNet-OR images and get good results. However, on NYUv2-OR, the test images are of low quality, with some amount of blur. To get better results, we do domain adaptation on the InteriorNet-OR images using the training images of NYUv2-OR as target domain and the method proposed by [161].

The quantitative results in Table 4.4 show that this domain adaptation is worthwhile: except for the "baseline" method, for which the gains are limited, we gain on all other methods at least 4.6 points and up to 10.8 points, depending on the considered metric.

Ablation study on ground truth generation. To illustrate the value of defining occlusion at order-1 as introduced in Section 4.3, here we show quantitative results where the occlusion ground truths are generated by occlusion at order-0 definition. We train on 10^4 domain-adapted images of InteriorNet-OR (cf. Table 4.2) and evaluate on NYUv2-OR dataset. As shown in Table 4.5, if the ground truths are generated without considering occlusion at order-1, the performance of the model degrades

Method	NYUv2-OR		
Metric	ODS	OIS	AP
ours (w/o order-1)	.404	.439	.368
ours	**.520**	**.540**	**.497**

Table 4.5: Ablation study about P2ORM ground truth generation.

greatly due to the inaccurate supervision signals.

4.8.4 Depth map refinement

The implementation and experiments of depth map refinement are mainly done by Yang Xiao, we are very grateful for his contributions in this section.

4.8.4.1 Evaluation on RGBD datasets

To assess our refinement approach, we compare with [111], which is the current state-of-the-art method for depth refinement on boundaries.

Figure 4.13 summarizes quantitative results on depth boundaries. We significantly improve edge metrics $\epsilon_{acc}, \epsilon_{comp}$ on NYUv2 [93] and iBims-1 [58], systematically outperforming [111] and showing consistency across the two different datasets. The differences on general metrics after refinement are negligible ($< 1\%$), i.e., we improve sharpness without degrading the overall depth.

Quantitative results of depth refinement on NYUv2 are shown in Table 4.6 for all metrics and for input depth maps obtained from a wide range of state-of-the-art depth estimation methods. After refinement, the improvement or degradation of general accuracy metrics (i.e., "Depth Error" and "Depth Accuracy") are negligible ($\leq$ 0.006 difference). This result is similar to the other depth refinement method, namely DispField [111]. However, we significantly and systematically outperform DispField on "Boundaries" metrics for the whole range of depth estimation methods.

We evaluate and compare our method to DispField [111] on iBims-1, in the same setting as for NYUv2 above, cf. Table 4.7. The results are similar: after refinement,

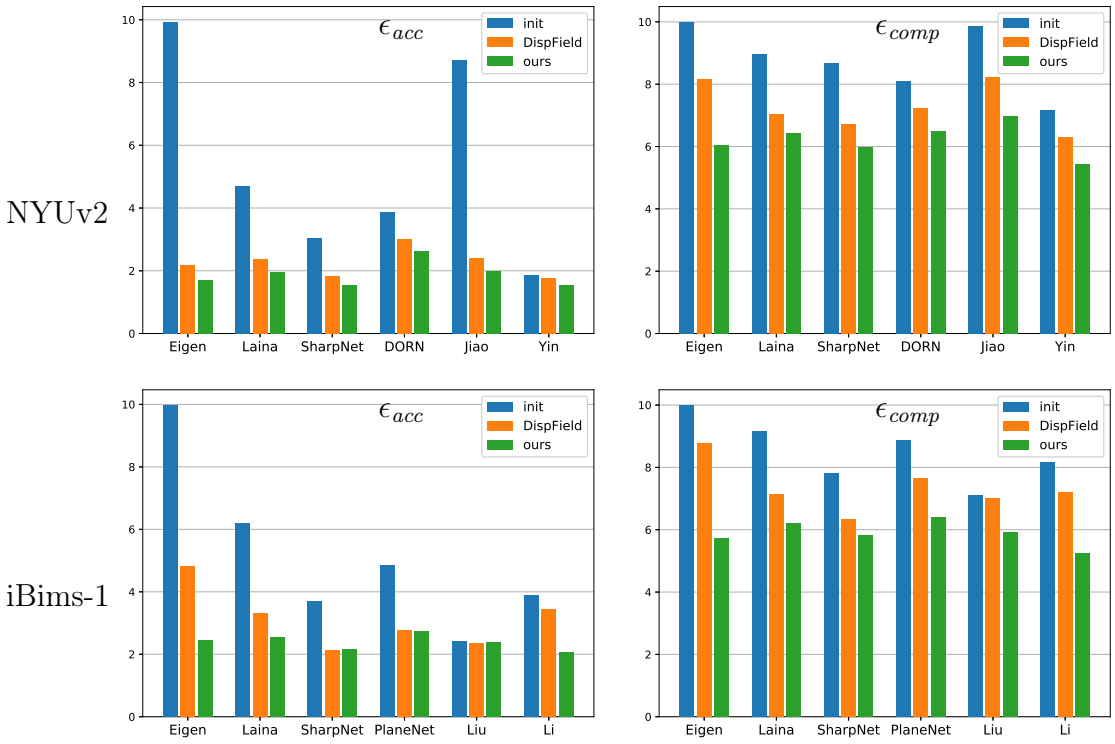


Figure 4.13: Gain in edge quality after depth refinement for metrics ϵ_{acc} (left) and ϵ_{comp} (right) on NYUv2 (top) for respectively [26, 61, 112, 30, 53, 156] and on iBimis-1 (bottom) for [26, 61, 112, 73, 74, 68]: metric on input depth maps (blue), after refining with [111] (orange), and after our refinement (green). Lower metric value is better.

the improvement or degradation of general accuracy metrics are negligible (≤ 0.02 difference); however, we significantly and almost systematically outperform DispField [111] on "Boundaries" metrics for the whole range of depth estimation methods.

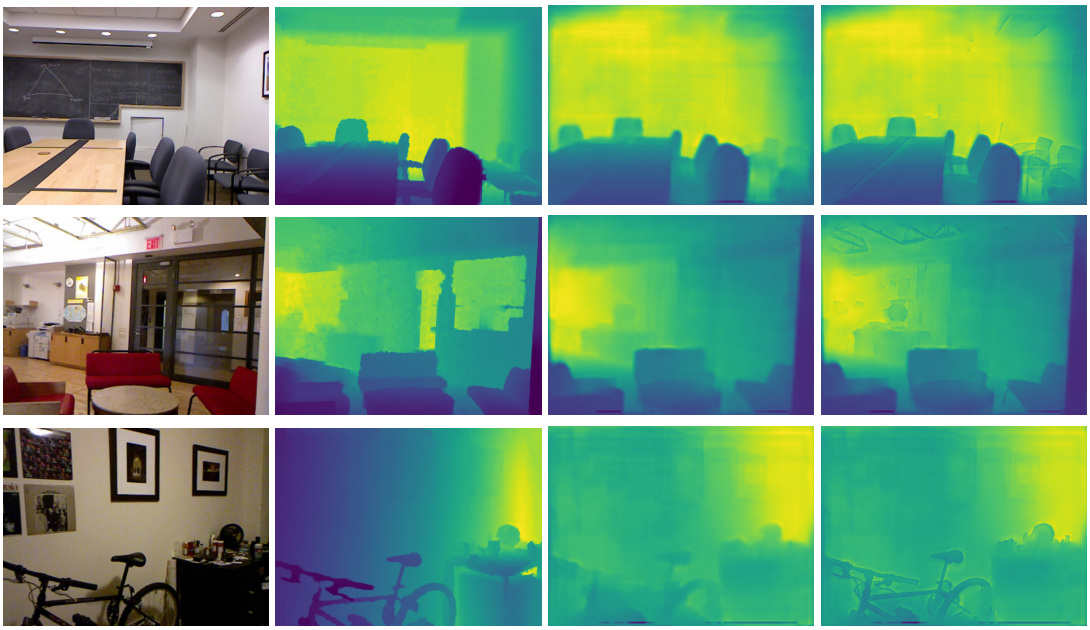
We illustrate here, in Figure 4.14, examples of refinements on NYUv2 with initial depth map estimation by SharpNet [112] as input, which is the second best method regarding boundary metrics ϵ_{acc} and ϵ_{comp}. Figure 4.15 illustrates the refinement on depth maps of iBims-1.

Depth estim. method	Refinement method	Boundaries(↓)		Depth Error(↓)				Depth Accuracy(↑)		
		ϵ_{acc}	ϵ_{comp}	rel	$\log_{10}$	RMS_{lin}	$\text{RMS}_{\log}$	σ_1	σ_2	σ_3
Eigen et al. [26]	—	9.926	9.993	0.236	0.095	0.765	0.265	0.611	0.887	0.971
	DispField [111]	2.168	8.173	0.232	0.094	0.758	0.263	0.615	0.889	0.971
	ours	**1.715**	**6.048**	0.231	0.095	0.761	0.264	0.615	0.888	0.970
Laina et al. [61]	—	4.702	8.982	0.142	0.059	0.510	0.181	0.818	0.955	0.988
	DispField [111]	2.372	7.041	0.140	0.059	0.509	0.180	0.819	0.956	0.989
	ours	**1.976**	**6.423**	0.142	0.059	0.508	0.181	0.818	0.955	0.988
Fu et al. [30]	—	3.872	8.117	0.131	0.053	0.493	0.174	0.848	0.956	0.984
	DispField [111]	3.001	7.242	0.136	0.054	0.502	0.178	0.844	0.954	0.983
	ours	**2.631**	**6.507**	0.132	0.053	0.487	0.173	0.848	0.957	0.985
Ramamonjisoa and Lepetit [112]	—	3.041	8.692	0.116	0.053	0.448	0.163	0.853	0.970	0.993
	DispField [111]	1.838	6.730	0.117	0.054	0.457	0.165	0.848	0.970	0.993
	ours	**1.546**	**5.988**	0.116	0.053	0.448	0.163	0.852	0.970	0.993
Jiao et al. [53]	—	8.730	9.864	0.093	0.043	0.356	0.134	0.908	0.981	0.995
	DispField [111]	2.410	8.230	0.092	0.042	0.352	0.132	0.910	0.981	0.995
	ours	**1.985**	**6.990**	0.093	0.042	0.351	0.133	0.909	0.981	0.995
Yin et al. [156]	—	1.854	7.188	0.112	0.047	0.417	0.144	0.880	0.975	0.994
	DispField [111]	1.762	6.307	0.112	0.047	0.419	0.144	0.879	0.975	0.994
	ours	**1.544**	**5.453**	0.113	0.047	0.421	0.145	0.878	0.975	0.994

Table 4.6: Evaluation of depth refinement on the output of several state-of-the-art methods on NYUv2 [93], cropped within valid region as in [26]. Best results in **bold**.

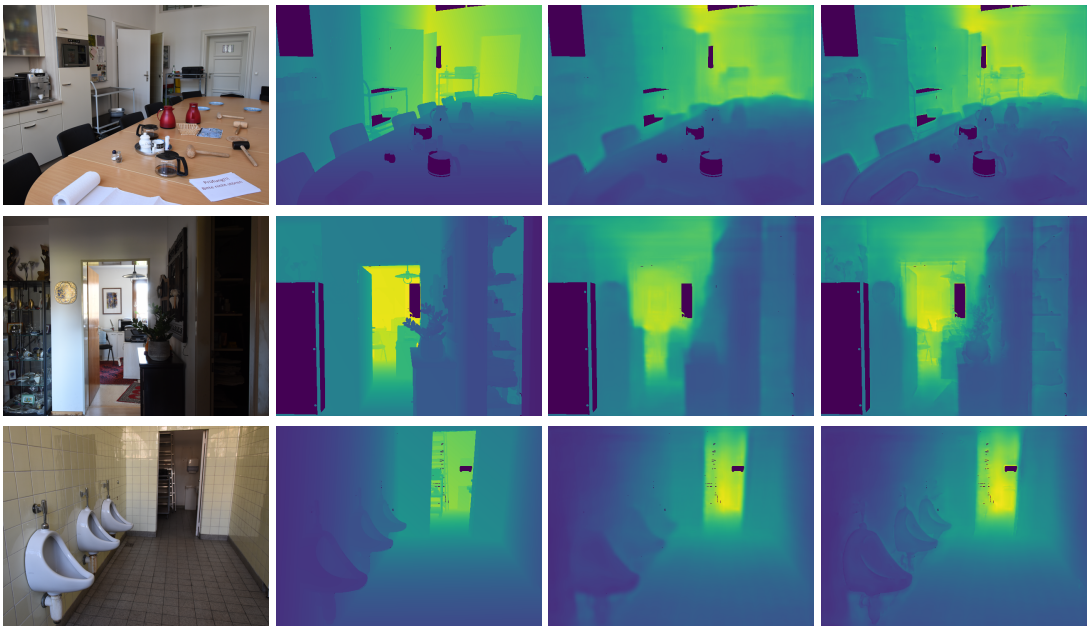
Depth estimation method	Refinement method	Boundaries(↓)		Depth Error(↓)			Depth Accuracy(↑)		
		ϵ_{acc}	ϵ_{comp}	rel	$\log_{10}$	RMS_{lin}	σ_1	σ_2	σ_3
Eigen et al. [26]	—	9.97	9.99	0.32	0.17	1.55	0.36	0.65	0.84
	DispField	4.83	8.78	0.32	0.17	1.54	0.37	0.66	0.85
	ours	**2.46**	**5.74**	0.32	0.17	1.55	0.36	0.65	0.84
Laina et al. [61]	—	6.19	9.17	0.26	0.13	1.20	0.50	0.78	0.91
	DispField	3.32	7.15	0.25	0.13	1.20	0.51	0.79	0.91
	ours	**2.56**	**6.20**	0.26	0.13	1.20	0.50	0.78	0.90
Liu et al. [74]	—	2.42	7.11	0.30	0.13	1.26	0.48	0.78	0.91
	DispField	**2.36**	7.00	0.30	0.13	1.26	0.48	0.77	0.91
	ours	2.37	**5.91**	0.30	0.13	1.26	0.48	0.78	0.91
Li et al. [68]	—	3.90	8.17	0.22	0.11	1.09	0.58	0.85	0.94
	DispField	3.43	7.19	0.22	0.11	1.10	0.58	0.84	0.94
	ours	**2.07**	**5.26**	0.22	0.11	1.10	0.58	0.84	0.94
Liu et al. [73]	—	4.84	8.86	0.29	0.17	1.45	0.41	0.70	0.86
	DispField	2.78	7.65	0.29	0.17	1.47	0.40	0.69	0.86
	ours	**2.75**	**6.40**	0.29	0.17	1.45	0.41	0.69	0.86
Ramamonjisoa and Lepetit [112]	—	3.69	7.82	0.27	0.11	1.08	0.59	0.83	0.93
	DispField	**2.13**	6.33	0.27	0.11	1.08	0.59	0.83	0.93
	ours	2.16	**5.82**	0.27	0.11	1.08	0.59	0.83	0.93

Table 4.7: Evaluation of depth refinement on the output of several state-of-the-art methods on iBims-1 [58], cropped within valid region as in [26]. Best results in **bold**.



(a) (b) (c) (d)

Figure 4.14: Qualitative results of depth refinement on NYUv2 [93]: (a) input RGB image from NYUv2, (b) ground-truth depth, (c) SharpNet depth estimation [112], (d) our refined depth.



(a) (b) (c) (d)

Figure 4.15: Qualitative results of depth refinement on iBims-1 [58]: (a) input RGB image from iBims-1, (b) ground-truth depth, (c) SharpNet depth prediction [112], (d) our refined depth.

Method	Boundaries(↓)		Depth Error(↓)				Depth Accuracy(↑)		
	ϵ_{acc}	ϵ_{comp}	rel	$\log_{10}$	RMS_{lin}	$\text{RMS}_{\log}$	σ_1	σ_2	σ_3
Initial estimation [26]	9.926	9.993	0.236	0.095	0.765	0.265	0.611	0.887	**0.971**
Bilateral Filter [134]	9.313	9.940	0.236	0.095	0.765	0.265	0.611	0.887	**0.971**
GF [38]	6.106	9.617	0.237	0.095	0.767	0.265	0.610	0.885	**0.971**
FBS [5]	5.428	9.454	0.236	0.095	0.765	0.264	0.611	0.887	**0.971**
Deep GF [145]	4.318	9.597	0.306	0.116	0.917	0.362	0.508	0.823	0.948
PACNet [124]	4.681	9.702	0.238	0.096	0.771	0.267	0.608	0.885	**0.971**
Ours	**1.715**	**6.048**	**0.231**	**0.095**	**0.761**	**0.264**	**0.615**	**0.888**	**0.971**

Table 4.8: Comparison with existing methods for image enhancement, adapted to the depth map refinement problems on NYUv2 [93] where the initial depth estimations are given by [26]. Best results in **bold**.

To further validate the effectiveness of P2ORM as depth refinement guidance, we also compare many existing methods using image intensity as guidance signal [134, 38, 5, 145, 124] where the initial depth prediction is given by [26]. As shown in Table 4.8, our method (line "Ours") achieves the best quantitative performance on NYUv2 [93]. The results show the superiority of P2ORM as a guidance signal for depth refinement w.r.t. image intensity.

4.8.4.2 Ablation study for depth refinement

Many variants and alternatives are possible to exploit our pixel-pair occlusion relationships for depth map refinement. We report here quantitative results justifying the particular choice we made in Section 4.5. To evaluate the effectiveness of different variants, we consider the estimation by [53] as input because this method provides the depth maps with the best accuracy on NYUv2 dataset [93]. But the conclusion is still valid for other methods.

Alternative network inputs. We first explore the influence of other types of input, in place of our pixel-pair occlusion relationships: the original RGB image, a normal map estimated using [11], and a classical occlusion edge mask (i.e., a binary map). The occlusion edge masks are created by thresholding the occlusion boundaries derived from the estimated occlusion relationships after Non Maximal Suppression (NMS), as described in Section 4.4. The network architecture and loss function are unchanged w.r.t. our proposed method, except that the first convolutional layer is adapted according to the number of input channels (1 more the edge map, 3 more for the RGB image or the normal map).

As shown in the top part of Table 4.9, using the RGB image as input (line "RGB") instead of our pixel-pair occlusion relationships (line "Refined (ours)") hardly improves the quality of the output depth map, which is not surprising as most the cues that can be directly exploited from the RGB image have already been exploited by [53]. Using the normal map leads to slightly lower depth errors but much worse depth boundaries. Last, using binary occlusion edges leads to a slightly higher depth accuracy but poor depth boundaries too. In the end, our estimated occlusion relationships as guidance achieves the lowest boundary errors without a noticeable degradation or improvement of the depth error and accuracy, which we believe is the best compromise.

Variant	Boundaries(↓) ϵ_{acc}	Boundaries(↓) ϵ_{comp}	Depth error(↓) rel	Depth error(↓) $\log_{10}$	Depth error(↓) RMS_{lin}	Depth error(↓) $\text{RMS}_{\log}$	Depth accuracy(↑) σ_1	Depth accuracy(↑) σ_2	Depth accuracy(↑) σ_3
Initial depth [53]	8.730	9.864	0.093	0.043	0.356	0.134	0.908	0.981	**0.995**
Refined with DispField [111]	2.410	8.230	0.092	0.042	0.352	0.132	0.910	0.981	**0.995**
Refined (ours)	**1.985**	**6.990**	0.093	0.042	0.351	0.133	0.909	0.981	**0.995**
Alternative network inputs (in addition to the rough depth map)									
RGB image	8.816	9.887	0.092	0.042	0.352	0.132	0.910	0.982	**0.995**
Normal map	9.437	9.937	**0.087**	**0.038**	**0.333**	**0.125**	0.917	0.982	0.996
Binary edges	5.619	9.397	0.096	0.044	0.362	0.138	**0.902**	**0.980**	**0.995**
Different loss functions $\mathcal{L}$ exploiting the ground-truth depth									
$\mathcal{L}_{\text{gtdepth}} + \mathcal{L}_{\text{regul}}$	8.756	9.866	0.093	0.043	0.356	0.134	0.908	0.981	**0.995**
$\mathcal{L}_{\text{occonsist}} + \mathcal{L}_{\text{gtdepth}}$	2.778	8.006	0.092	0.042	0.356	0.133	0.909	0.981	**0.995**
$\mathcal{L}_{\text{occonsist}} + \mathcal{L}_{\text{regul}} + \mathcal{L}_{\text{gtdepth}}$	3.090	7.291	0.093	0.042	0.351	0.132	0.910	0.982	**0.995**
Different depth combinations used in $\mathcal{L}_{\text{occonsist}}$									
dd (order-0 depth only)	2.375	7.406	0.094	0.043	0.356	0.135	0.907	0.981	**0.995**
DD (order-1 depth only)	2.401	7.373	0.093	0.042	0.352	0.133	0.909	0.981	**0.995**

Table 4.9: Ablation study about depth map refinement: (a) using alternative network inputs, (b) using different loss functions exploiting the ground-truth depth, (c) using a different combination of order-0 and order-1 depth difference in $\mathcal{L}_{\text{occonsist}}$. See details in text. Best results in **bold**.

Different loss functions exploiting the ground-truth depth. Then we study variations in the loss function when adding ground-truth depth information at training time. We introduce the loss function $\mathcal{L}_{\text{gtdepth}}$, which is the counterpart of $\mathcal{L}_{\text{regul}}$ using the ground-truth depth d instead of the rough input depth $\tilde{d}$: it penalizes the difference between the refined depth $\hat{d}$ and the ground-truth depth d as defined in Equation (4.10):

$$\mathcal{L}_{\text{gtdepth}} = \frac{1}{|\mathcal{P}|} \sum_{p \in \mathcal{P}} \left(\mathcal{B}(\log d_p, \log \hat{d}_p) + \left\| \nabla \log d_p - \nabla \log \hat{d}_p \right\|^2 \right) \tag{4.10}$$

We study different combinations of partial losses, i.e., $\mathcal{L}_{\text{gtdepth}} + \mathcal{L}_{\text{regul}}$ (which ignores occlusion information), $\mathcal{L}_{\text{occonsist}} + \mathcal{L}_{\text{gtdepth}}$ (which does not penalize difference between $\tilde{d}$ and $\hat{d}$), and $\mathcal{L}_{\text{occonsist}} + \mathcal{L}_{\text{regul}} + \mathcal{L}_{\text{gtdepth}}$ (which combines both the rough input depth and ground-truth depth information), comparing to the loss $\mathcal{L}_{\text{refine}}$ defined in Equation (4.8) (which uses only the rough input depth), i.e., line "Refined (ours)" in Table 4.9.

As shown in the middle part of Table 4.9, $\mathcal{L}_{\text{gtdepth}} + \mathcal{L}_{\text{regul}}$ does not improve or

degrade the input depth map noticeably; information about edges [111] or occlusions is missing to yield any significant improvement. Replacing the rough input depth map by the ground truth as $\mathcal{L}_{\mathsf{occonsist}} + \mathcal{L}_{\mathsf{gtdepth}}$ significantly improves ϵ_{acc}, slightly improves ϵ_{comp} and does not affect much the general depth error and accuracy metrics. But it is not as good as the performance of our method. Finally, using both the rough input depth map and the ground-truth depth map as $\mathcal{L}_{\mathsf{occonsist}} + \mathcal{L}_{\mathsf{regul}} + \mathcal{L}_{\mathsf{gtdepth}}$, i.e., adding ground-truth information to our setting, is not as good as not using it.

Different combinations or order-0 and order-1 depths. Last, we also consider variations in the depths used in $\mathcal{L}_{\mathsf{occonsist}}$, using either the refined order-0 depth difference $\hat{d}_{pq}$ or the tangent-adjusted order-1 depth difference $\hat{D}_{pq}$. More precisely, we consider the cases where the signed distances in Equation (4.8) are both $\hat{d}_{pq}$ (named "dd") or both $\hat{D}_{pq}$ (named "DD"), instead of $\hat{d}_{pq}$ then $\hat{D}_{pq}$ as defined in Equation (4.8).

As can be seen in the bottom part of Table 4.9, the performance of both variants is not as good as the loss function we define in Equation (4.8).

Method	Boundaries(↓)		Depth Error(↓)			Depth Accuracy(↑)		
	ϵ_{acc}	ϵ_{comp}	rel	$\log_{10}$	RMS_{lin}	σ_1	σ_2	σ_3
Baseline	2.171	6.387	0.116	0.048	0.526	0.888	0.980	0.993
Ours	**1.830**	**5.965**	0.110	0.044	0.492	0.891	0.981	0.993

Table 4.10: Comparison with our baseline method for monocular depth estimation on SceneNet [91]. Best depth boundaries results in **bold**.

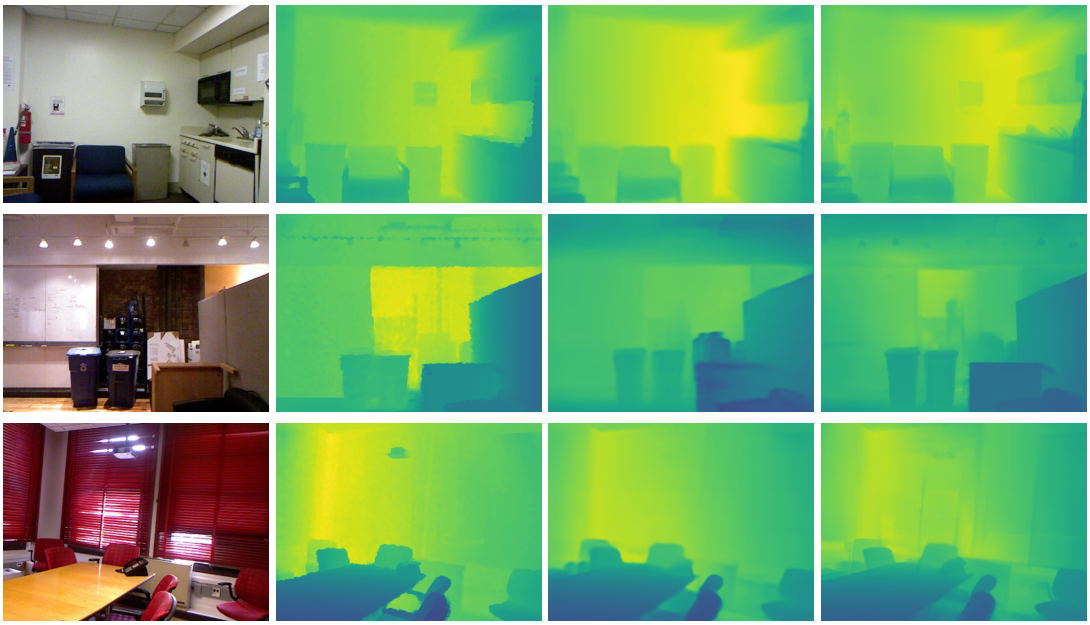
Depth estimation method	Boundaries(↓)		Depth Error(↓)			Depth Accuracy(↑)		
	ϵ_{acc}	ϵ_{comp}	rel	$\log_{10}$	RMS_{lin}	σ_1	σ_2	σ_3
Eigen et al. [26]	9.926	9.993	0.236	0.095	0.765	0.611	0.887	0.971
Laina et al. [61]	4.702	8.982	0.142	0.059	0.510	0.818	0.955	0.988
Fu et al. [30]	3.872	8.117	0.131	0.053	0.493	0.848	0.956	0.984
Ramamonjisoa and Lepetit [112]	3.041	8.692	0.116	0.053	0.448	0.853	0.970	0.993
Yin et al. [156]	1.854	7.188	0.112	0.047	0.417	0.880	0.975	0.994
Ours	**1.398**	**6.414**	0.130	0.056	0.483	0.834	0.966	0.992

Table 4.11: Evaluation of monocular depth estimation on NYUv2 [93], cropped within valid region as in [26]. Best depth boundaries results in **bold**.

4.8.5 Accurate depth estimation on boundaries

We first evaluate our method as introduced in Section 4.6 on the synthetic dataset SceneNet [91]. To disentangle the respective contribution of the proposed method, we evaluate a 'baseline' method which means training our monocular depth estimation module G_{i2d} without the supervision of P2ORM introduced in Section 4.6. As shown in Table 4.10, compared to our baseline method, our method successfully recover more precise depth boundaries.

Then we evaluate our method quantitatively on NYUv2 dataset [93] in Table 4.11. The proposed method also successfully recovers more precise depth boundaries compared to other state-of-the-art methods. Figure 4.16 offers qualitative results of our method compared to one state-of-the-art method Yin et al. [156], it is evident that our method recovers better depth discontinuities while a depth refinement stage is not needed.



(a) (b) (c) (d)

Figure 4.16: Qualitative results of monocular depth estimation on NYUv2 [93]: (a) input RGB image, (b) ground-truth depth, (c) depth prediction by Yin et al. [156], (d) our depth prediction.

4.9 Conclusion

In this chapter, we propose a new representation of occlusion relationship based on pixel pairs and design a simple network architecture to estimate it. Translating our results into standard occlusion boundaries for comparison, we significantly outperform the state-of-the-art for oriented occlusion boundary estimation. To illustrate the potential of our representation, we also propose a depth map refinement model that exploits our estimated occlusion relationships. It also consistently outperforms the state-of-the-art regarding depth edge sharpness, without degrading accuracy in the rest of the depth image. By using P2ORM as a supervision signal, we further propose a one-stage monocular depth estimation method that recovers high-quality depth discontinuities without a depth map refinement stage. These results are made possible thanks to a our method which automatically generate accurate occlusion relationship labels from depth maps, on a large scale.

Chapter 5

Discussion

In this chapter, we conclude the thesis by providing in Section 5.1 a summary of its contributions and outlining in Section 5.2 directions of future work.

5.1 Summary of Contributions

This thesis has addressed 2D and 3D geometric attributes estimation in images via deep learning. More concretely, we developed a series of deep-learning-based approaches that aim to improve the performance of geometric attributes estimation tasks such as object visual tracking, object 6D pose estimation, scene occlusion reasoning and monocular depth estimation. Meanwhile, automatic labeling approaches and synthetic data generation methods are also proposed to make our models less dependent on manually labeled training data. To summarize, our contributions are threefold:

- In Chapter 2, we involved semantics information offered by semantic segmentation models in the DCF tracking framework, which allowed the proposed tracker to make use of the semantics of both the tracking target and its surrounding environment. Experiments on public benchmarks demonstrated that our method could improve both the accuracy and the robustness of tracking.

- In Chapter 3, we first introduced our monocular object 6D pose estimation method that exploits the 3D-to-2D correspondences between the object image and the object 3D shape. Our experiments demonstrated the performance boost of using these correspondences. Then we proposed an object pose refinement method for images in the wild while existing methods work on images captured with experimental setup. To make our method more general, we also investigated the generalization ability of our models trained on generated synthetic data and evaluated our models on objects which are not seen during training. Evaluation on pubic datasets showed that our models could achieve good pose refinement results.

- In Chapter 4, we formalized the notion of geometric occlusion in single images and proposed an automatic labeling method for the generation of high-quality occlusion annotations. Based on our occlusion definition, we further proposed a new pixel-pair occlusion relationship formulation and the corresponding inference method using deep models. Experiments on occlusion reasoning benchmarks demonstrated the superiority of the proposed formulation and method. In order to recover accurate depth discontinuities, we also proposed a depth map refinement method and a single-stage monocular depth estimation method. By using our occlusion formulation as a guidance signal, both these methods significantly improve depth estimation by making occlusion edges in the scene much sharper. Besides, we released three datasets containing generated occlusion annotations (i.e., InteriorNet-OR, NYUv2-OR and iBims-1-OR) to contribute to the advancement of research in the community.

5.2 Future Work

Synthetic-real domain gap. Compared to real data, synthetic data is more controllable and enables the training on large-scale datasets without expensive manual

labeling. However, the appearance difference between real images and synthetic images still limits the wide applicability of synthetic data in training tasks. In Chapter 3, we still observed a performance gap between the models trained on real images and the ones trained on synthetic images. In Chapter 4, we compared the models trained on synthetic data and the ones trained on synthetic data with a synthetic-to-real domain adaption, the difference of performance illustrates the value of photo-realism in synthetic data generation. A valuable avenue of study is image synthesis methods and domain adaption methods to further reduce the appearance difference between synthetic images and real images, and improve the performance of models trained on synthetic data.

Another potential extension is the exploration of methods that bring synthetic data distributions closer to real data distributions. For example, in Chapter 4, the authors of the synthetic dataset [70] use real indoor room layouts created by interior designers to approximate room layouts existing in real data. A less expensive and more general way to do so can be the estimation of rough room layouts directly from real images acquired from the internet and the use of these estimated room layouts for synthetic data generation. The diversity of synthetic data is important and worth a further study.

Occlusion reasoning in object 2D/3D detection and segmentation. In Chapter 4, we studied low-level occlusion reasoning following a pure geometric definition. As shown in Chapter 2 and Chapter 3, occlusion is ubiquitous in object 2D/3D detection and segmentation, and can induce a performance degradation by misleading corresponding estimators into focusing on neighboring objects. The proposed occlusion formulation can be a useful guidance signal to discriminate different objects in the scene and therefore helps determine the region of individual object instances. By paying more attention to the correctly estimated object region, object 2D/3D detectors have a higher chance to achieve more accurate estimates. Due to the lack of

time, we haven't conducted any experiment relevant to this topic. We believe that this is an interesting next step and it will be one of our future work.

Bibliography

[1] Acuna, D., Kar, A., Fidler, S.: Devil is in the edges: Learning semantic boundaries from noisy annotations. In: Conference on Computer Vision and Pattern Recognition (CVPR). pp. 11075–11083 (2019)

[2] Alhashim, I., Wonka, P.: High quality monocular depth estimation via transfer learning. arXiv:1812.11941 (2018)

[3] Apostoloff, N., Fitzgibbon, A.: Learning spatiotemporal t-junctions for occlusion detection. In: Conference on Computer Vision and Pattern Recognition (CVPR). vol. 2, pp. 553–559. IEEE (2005)

[4] Badrinarayanan, V., Kendall, A., Cipolla, R.: Segnet: A deep convolutional encoder-decoder architecture for image segmentation. IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI) **39**(12), 2481–2495 (2017)

[5] Barron, J.T., Poole, B.: The fast bilateral solver. In: European Conference on Computer Vision (ECCV). pp. 617–632 (2016)

[6] Bay, H., Tuytelaars, T., Van Gool, L.: Surf: Speeded up robust features. In: European Conference on Computer Vision (ECCV). pp. 404–417. Springer (2006)

[7] Bertinetto, L., Valmadre, J., Golodetz, S., Miksik, O., Torr, P.H.: Staple: Complementary learners for real-time tracking. In: Conference on Computer Vision and Pattern Recognition (CVPR). pp. 1401–1409 (2016)

[8] Bertinetto, L., Valmadre, J., Henriques, J.F., Vedaldi, A., Torr, P.H.: Fully-convolutional siamese networks for object tracking. In: European Conference on Computer Vision (ECCV). pp. 850–865. Springer (2016)

[9] Bhat, G., Danelljan, M., Gool, L.V., Timofte, R.: Learning discriminative model prediction for tracking. In: International Conference on Computer Vision (ICCV). pp. 6182–6191 (2019)

[10] Bolme, D.S., Beveridge, J.R., Draper, B.A., Lui, Y.M.: Visual object tracking using adaptive correlation filters. In: Conference on Computer Vision and Pattern Recognition (CVPR). pp. 2544–2550. IEEE (2010)

[11] Boulch, A., Marlet, R.: Fast and robust normal estimation for point clouds with sharp features. Computer Graphics Forum (CGF) **31**(5), 1765–1774 (2012)

[12] 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: Conference on Computer Vision and Pattern Recognition (CVPR) (2016)

[13] Chang, A.X., Funkhouser, T., Guibas, L., Hanrahan, P., Huang, Q., Li, Z., Savarese, S., Savva, M., Song, S., Su, H., Xiao, J., Yi, L., Yu, F.: ShapeNet: An Information-Rich 3D Model Repository. Tech. Rep. arXiv:1512.03012, Stanford University – Princeton University – Toyota Technological Institute at Chicago (2015)

[14] Chatfield, K., Simonyan, K., Vedaldi, A., Zisserman, A.: Return of the devil in the details: Delving deep into convolutional nets. British Machine Vision Conference (BMVC) (2014)

[15] Chen, L.C., Papandreou, G., Kokkinos, I., Murphy, K., Yuille, A.L.: Semantic image segmentation with deep convolutional nets and fully connected crfs. ICLR (2014)

[16] Chen, L.C., Papandreou, G., Kokkinos, I., Murphy, K., Yuille, A.L.: DeepLab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected CRFs. IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI) **40**(4), 834–848 (2017)

[17] Cooper, M.C.: Interpreting line drawings of curved objects with tangential edges and surfaces. Image and Vision Computing **15**(4), 263–276 (1997)

[18] Danelljan, M., Bhat, G., Shahbaz Khan, F., Felsberg, M.: Eco: Efficient convolution operators for tracking. In: Conference on Computer Vision and Pattern Recognition (CVPR). pp. 6638–6646 (2017)

[19] Danelljan, M., Hager, G., Shahbaz Khan, F., Felsberg, M.: Convolutional features for correlation filter based visual tracking. In: International Conference on Computer Vision Workshops (ICCV Workshops). pp. 58–66 (2015)

[20] Danelljan, M., Hager, G., Shahbaz Khan, F., Felsberg, M.: Learning spatially regularized correlation filters for visual tracking. In: International Conference on Computer Vision (ICCV). pp. 4310–4318 (2015)

[21] Danelljan, M., Robinson, A., Khan, F.S., Felsberg, M.: Beyond correlation filters: Learning continuous convolution operators for visual tracking. In: European Conference on Computer Vision (ECCV). pp. 472–488. Springer (2016)

[22] Deng, J., Dong, W., Socher, R., Li, L.J., Li, K., Fei-Fei, L.: Imagenet: A large-scale hierarchical image database. In: Conference on Computer Vision and Pattern Recognition (CVPR). pp. 248–255 (2009)

[23] Dollár, P., Zitnick, C.L.: Fast edge detection using structured forests. IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI) **37**(8), 1558–1570 (2014)

[24] Dosovitskiy, A., Fischer, P., Ilg, E., Hausser, P., Hazirbas, C., Golkov, V., Van Der Smagt, P., Cremers, D., Brox, T.: Flownet: Learning optical flow with convolutional networks. In: International Conference on Computer Vision (ICCV). pp. 2758–2766 (2015)

[25] Eigen, D., Fergus, R.: Predicting depth, surface normals and semantic labels with a common multi-scale convolutional architecture. In: Conference on Computer Vision and Pattern Recognition (CVPR). pp. 2650–2658 (2015)

[26] Eigen, D., Puhrsch, C., Fergus, R.: Depth map prediction from a single image using a multi-scale deep network. In: Ghahramani, Z., Welling, M., Cortes, C., Lawrence, N.D., Weinberger, K.Q. (eds.) Conference on Neural Information Processing Systems (NeurIPS), pp. 2366–2374. Curran Associates, Inc. (2014)

[27] Elhoseiny, M., El-Gaaly, T., Bakry, A., Elgammal, A.M.: A comparative analysis and study of multiview CNN models for joint object categorization and pose estimation. In: International Conference on Machine Learning (ICML) (2016)

[28] Everingham, M., Van Gool, L., Williams, C.K.I., Winn, J., Zisserman, A.: The PASCAL Visual Object Classes Challenge 2011 (VOC2011) Results. http://www.pascal-network.org/challenges/VOC/voc2011/workshop/index.html

[29] Ferraz, L., Binefa, X., Moreno-Noguer, F.: Very fast solution to the PnP problem with algebraic outlier rejection. In: Conference on Computer Vision and Pattern Recognition (CVPR) (2014)

[30] Fu, H., Gong, M., Wang, C., Batmanghelich, K., Tao, D.: Deep ordinal regression network for monocular depth estimation. In: Conference on Computer Vision and Pattern Recognition (CVPR). pp. 2002–2011 (2018)

[31] Fu, H., Wang, C., Tao, D., Black, M.J.: Occlusion boundary detection via deep exploration of context. In: Conference on Computer Vision and Pattern Recognition (CVPR). pp. 241–250 (2016)

[32] Geiger, A., Lenz, P., Urtasun, R.: Are we ready for autonomous driving? the KITTI vision benchmark suite. In: Conference on Computer Vision and Pattern Recognition (CVPR) (2012)

[33] Georgakis, G., Karanam, S., Wu, Z., Kosecka, J.: Matching RGB images to CAD models for object pose estimation. CoRR **abs/1811.07249** (2018)

[34] Godard, C., Mac Aodha, O., Brostow, G.J.: Unsupervised monocular depth estimation with left-right consistency. In: Conference on Computer Vision and Pattern Recognition (CVPR). pp. 270–279 (2017)

[35] Grabner, A., Roth, P.M., Lepetit, V.: 3D pose estimation and 3D model retrieval for objects in the wild. In: Conference on Computer Vision and Pattern Recognition (CVPR) (2018)

[36] Güler, R.A., Trigeorgis, G., Antonakos, E., Snape, P., Zafeiriou, S., Kokkinos, I.: Densereg: Fully convolutional dense shape regression in-the-wild. In: Conference on Computer Vision and Pattern Recognition (CVPR) (2017)

[37] He, A., Luo, C., Tian, X., Zeng, W.: A twofold siamese network for real-time object tracking. In: Conference on Computer Vision and Pattern Recognition (CVPR). pp. 4834–4843 (2018)

[38] He, K., Sun, J., Tang, X.: Guided image filtering. In: European Conference on Computer Vision (ECCV). pp. 1–14 (2010)

[39] He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: Conference on Computer Vision and Pattern Recognition (CVPR). pp. 770–778 (2016)

[40] He, X., Yuille, A.: Occlusion boundary detection using pseudo-depth. In: European Conference on Computer Vision (ECCV). pp. 539–552. Springer (2010)

[41] Heise, P., Klose, S., Jensen, B., Knoll, A.: PM-Huber: Patchmatch with Huber regularization for stereo matching. In: International Conference on Computer Vision (ICCV). pp. 2360–2367 (2013)

[42] Henriques, J.F., Caseiro, R., Martins, P., Batista, J.: Exploiting the circulant structure of tracking-by-detection with kernels. In: European Conference on Computer Vision (ECCV). pp. 702–715. Springer (2012)

[43] Henriques, J.F., Caseiro, R., Martins, P., Batista, J.: High-speed tracking with kernelized correlation filters. IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI) **37**(3), 583–596 (2014)

[44] Heo, M., Lee, J., Kim, K.R., Kim, H.U., Kim, C.S.: Monocular depth estimation using whole strip masking and reliability-based refinement. In: European Conference on Computer Vision (ECCV). pp. 36–51 (2018)

[45] Hinterstoisser, S., Cagniart, C., Ilic, S., Sturm, P., Navab, N., Fua, P., Lepetit, V.: Gradient response maps for real-time detection of textureless objects. IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI) (2012)

[46] 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: Asian Conference on Computer Vision (ACCV) (2012)

[47] Hoiem, D., Efros, A.A., Hebert, M.: Recovering occlusion boundaries from an image. International Journal of Computer Vision (IJCV) **91**, 328–346 (2010)

[48] Hong, S., You, T., Kwak, S., Han, B.: Online tracking by learning discriminative saliency map with convolutional neural network. In: International Conference on Machine Learning (ICML). pp. 597–606 (2015)

[49] Hong, Z., Chen, Z., Wang, C., Mei, X., Prokhorov, D., Tao, D.: Multi-store tracker (muster): A cognitive psychology inspired approach to object tracking. In: Conference on Computer Vision and Pattern Recognition (CVPR). pp. 749–758 (2015)

[50] Huang, G., Liu, Z., Van Der Maaten, L., Weinberger, K.Q.: Densely connected convolutional networks. In: Conference on Computer Vision and Pattern Recognition (CVPR). pp. 4700–4708 (2017)

[51] Ilg, E., Saikia, T., Keuper, M., Brox, T.: Occlusions, motion and depth boundaries with a generic network for disparity, optical flow or scene flow estimation. In: European Conference on Computer Vision (ECCV). pp. 614–630 (2018)

[52] Jégou, S., Drozdzal, M., Vazquez, D., Romero, A., Bengio, Y.: The one hundred layers tiramisu: Fully convolutional densenets for semantic segmentation. In: Conference on Computer Vision and Pattern Recognition Worshops(CVPR Worshops). pp. 11–19 (2017)

[53] Jiao, J., Cao, Y., Song, Y., Lau, R.W.H.: Look deeper into depth: Monocular depth estimation with semantic booster and attention-driven loss. In: European Conference on Computer Vision (ECCV) (2018)

[54] Kehl, W., Manhardt, F., Tombari, F., Ilic, S., Navab, N.: SSD-6D: Making RGB-based 3D detection and 6D pose estimation great again. In: International Conference on Computer Vision (ICCV) (2017)

[55] Kendall, A., Cipolla, R.: Geometric loss functions for camera pose regression with deep learning. In: Conference on Computer Vision and Pattern Recognition (CVPR) (2017)

[56] Kendall, A., Grimes, M.K., Cipolla, R.: PoseNet: A convolutional network for real-time 6-DOF camera relocalization. In: International Conference on Computer Vision (ICCV) (2015)

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

[58] Koch, T., Liebel, L., Fraundorfer, F., Körner, M.: Evaluation of CNN-based single-image depth estimation methods. pp. 331–348. Springer International Publishing (2019)

[59] Kristan, M., Matas, J., Leonardis, A., Vojir, T., Pflugfelder, R., Fernandez, G., Nebehay, G., Porikli, F., Čehovin, L.: A novel performance evaluation methodology for single-target trackers. IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI) **38**(11), 2137–2155 (2016)

[60] Krizhevsky, A., Sutskever, I., Hinton, G.E.: Imagenet classification with deep convolutional neural networks. In: Conference on Neural Information Processing Systems (NeurIPS). pp. 1097–1105 (2012)

[61] Laina, I., Rupprecht, C., Belagiannis, V., Tombari, F., Navab, N.: Deeper depth prediction with fully convolutional residual networks. In: International Conference on 3D Vision (3DV). pp. 239–248. IEEE (2016)

[62] Lee, J.H., Kim, C.S.: Monocular depth estimation using relative depth maps. In: Conference on Computer Vision and Pattern Recognition (CVPR). pp. 9729–9738 (2019)

[63] Leichter, I., Lindenbaum, M.: Boundary ownership by lifting to 2.1-D. In: International Conference on Computer Vision (ICCV). pp. 9–16. IEEE (2008)

[64] Lepetit, V., Moreno-Noguer, F., Fua, P.: EPnP: An accurate O(n) solution to the PnP problem. International Journal of Computer Vision (IJCV) (2009)

[65] Li, C., Bai, J., Hager, G.D.: A unified framework for multi-view multi-class object pose estimation. In: European Conference on Computer Vision (ECCV) (2018)

[66] Li, D., Wang, H., Yin, Y., Wang, X.: Deformable registration using edge-preserving scale space for adaptive image-guided radiation therapy. Journal of Applied Clinical Medical Physics (JACMP) (2011)

[67] Li, F., Tian, C., Zuo, W., Zhang, L., Yang, M.H.: Learning spatial-temporal regularized correlation filters for visual tracking. In: Conference on Computer Vision and Pattern Recognition (CVPR). pp. 4904–4913 (2018)

[68] Li, J.Y., Klein, R., Yao, A.: A two-streamed network for estimating fine-scaled depth maps from single RGB images. In: International Conference on Computer Vision (ICCV). pp. 3392–3400 (2016)

[69] Li, S., Xu, C., Xie, M.: A robust O(n) solution to the perspective-n-point problem. IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI) (2012)

[70] Li, W., Saeedi, S., McCormac, J., Clark, R., Tzoumanikas, D., Ye, Q., Huang, Y., Tang, R., Leutenegger, S.: InteriorNet: Mega-scale multi-sensor photo-realistic indoor scenes dataset. In: British Machine Vision Conference (BMVC) (2018)

[71] Li, Y., Zhu, J.: A scale adaptive kernel correlation filter tracker with feature integration. In: European Conference on Computer Vision (ECCV). pp. 254–265. Springer (2014)

[72] Li, Y., Wang, G., Ji, X., Xiang, Y., Fox, D.: DeepIM: Deep iterative matching for 6D pose estimation. In: European Conference on Computer Vision (ECCV) (2018)

[73] Liu, C., Yang, J., Ceylan, D., Yumer, E., Furukawa, Y.: PlaneNet: Piece-wise planar reconstruction from a single RGB image. In: Conference on Computer Vision and Pattern Recognition (CVPR). pp. 2579–2588 (2018)

[74] Liu, F., Shen, C., Lin, G.: Deep convolutional neural fields for depth estimation from a single image. In: Conference on Computer Vision and Pattern Recognition (CVPR) (2015)

[75] Liu, T., Wang, G., Yang, Q.: Real-time part-based visual tracking via adaptive correlation filters. In: Conference on Computer Vision and Pattern Recognition (CVPR). pp. 4902–4912 (2015)

[76] Liu, Y., Cheng, M.M., Fan, D.P., Zhang, L., Bian, J., Tao, D.: Semantic edge detection with diverse deep supervision. arXiv:1804.02864 (2018)

[77] Loing, V., Marlet, R., Aubry, M.: Virtual training for a real application: Accurate object-robot relative localization without calibration. International Journal of Computer Vision (IJCV) **126**(9), 1045–1060 (2018)

[78] Long, J., Shelhamer, E., Darrell, T.: Fully convolutional networks for semantic segmentation. In: Conference on Computer Vision and Pattern Recognition (CVPR). pp. 3431–3440 (2015)

[79] Lowe, D.G.: Fitting parameterized three-dimensional models to images. IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI) (1991)

[80] Lowe, D.G.: Distinctive image features from scale-invariant keypoints. International Journal of Computer Vision (IJCV) (2004)

[81] Lu, R., Xue, F., Zhou, M., Ming, A., Zhou, Y.: Occlusion-shared and feature-separated network for occlusion relationship reasoning. In: International Conference on Computer Vision (ICCV) (2019)

[82] Ma, C., Huang, J.B., Yang, X., Yang, M.H.: Hierarchical convolutional features for visual tracking. In: International Conference on Computer Vision (ICCV). pp. 3074–3082 (2015)

[83] Ma, C., Yang, X., Zhang, C., Yang, M.H.: Long-term correlation tracking. In: Conference on Computer Vision and Pattern Recognition (CVPR). pp. 5388–5396 (2015)

[84] Mahendran, S., Ali, H., Vidal, R.: A mixed classification-regression framework for 3D pose estimation from 2D images. In: British Machine Vision Conference (BMVC) (2018)

[85] Manhardt, F., Kehl, W., Navab, N., Tombari, F.: Deep model-based 6D pose refinement in RGB. In: European Conference on Computer Vision (ECCV) (2018)

[86] Manhardt, F., Kehl, W., Navab, N., Tombari, F.: Deep model-based 6d pose refinement in rgb. In: European Conference on Computer Vision (ECCV). pp. 800–815 (2018)

[87] Maninis, K.K., Caelles, S., Chen, Y., Pont-Tuset, J., Leal-Taixé, L., Cremers, D., Van Gool, L.: Video object segmentation without temporal information. IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI) **41**(6), 1515–1530 (2018)

[88] Martin, D.R., Fowlkes, C.C., Malik, J.: Learning to detect natural image boundaries using local brightness, color, and texture cues. IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI) **26**(5), 530–549 (2004)

[89] Massa, F., Marlet, R., Aubry, M.: Crafting a multi-task CNN for viewpoint estimation. In: British Machine Vision Conference (BMVC) (2016)

[90] Massa, F., Marlet, R., Aubry, M.: Crafting a multi-task CNN for viewpoint estimation. In: British Machine Vision Conference (BMVC) (2016)

[91] McCormac, J., Handa, A., Leutenegger, S., Davison, A.J.: Scenenet rgb-d: Can 5m synthetic images beat generic imagenet pre-training on indoor segmentation? In: International Conference on Computer Vision (ICCV). pp. 2678–2687 (2017)

[92] Mousavian, A., Anguelov, D., Flynn, J., Kosecka, J.: 3D bounding box estimation using deep learning and geometry. In: Conference on Computer Vision and Pattern Recognition (CVPR) (2017)

[93] Nathan Silberman, Derek Hoiem, P.K., Fergus, R.: Indoor segmentation and support inference from RGBD images. In: European Conference on Computer Vision (ECCV) (2012)

[94] Nitzberg, M., Mumford, D.B.: The 2.1-D sketch. IEEE Computer Society Press (1990)

[95] Nocedal, J., Wright, S.: Numerical optimization. Springer Science & Business Media (2006)

[96] Oberweger, M., Rad, M., Lepetit, V.: Making deep heatmaps robust to partial occlusions for 3D object pose estimation. In: European Conference on Computer Vision (ECCV) (2018)

[97] Oberweger, M., Rad, M., Lepetit, V.: Making deep heatmaps robust to partial occlusions for 3D object pose estimation. In: European Conference on Computer Vision (ECCV). pp. 119–134 (2018)

[98] Oh, S.W., Lee, J.Y., Sunkavalli, K., Kim, S.J.: Fast video object segmentation by reference-guided mask propagation. In: Conference on Computer Vision and Pattern Recognition (CVPR). pp. 7376–7385 (2018)

[99] Osadchy, M., Cun, Y.L., Miller, M.L.: Synergistic face detection and pose estimation with energy-based models. Journal of Machine Learning Research (JMLR) (2007)

[100] Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., Killeen, T., Lin, Z., Gimelshein, N., Antiga, L., et al.: PyTorch: An imperative style, high-performance deep learning library. In: Conference on Neural Information Processing Systems (NeurIPS). pp. 8024–8035 (2019)

[101] Pavlakos, G., Zhou, X., Chan, A., Derpanis, K.G., Daniilidis, K.: 6-dof object pose from semantic keypoints. In: International Conference on Robotics and Automation (ICRA) (2017)

[102] Penedones, H., Collobert, R., Fleuret, F., Grangier, D.: Improving object classification using pose information. Tech. rep., Idiap Research Institute (2012)

[103] Peng, S., Liu, Y., Huang, Q., Zhou, X., Bao, H.: PVNet: Pixel-wise voting network for 6DoF pose estimation. In: Conference on Computer Vision and Pattern Recognition (CVPR). pp. 4561–4570 (2019)

[104] Perazzi, F., Khoreva, A., Benenson, R., Schiele, B., Sorkine-Hornung, A.: Learning video object segmentation from static images. In: Conference on Computer Vision and Pattern Recognition (CVPR). pp. 2663–2672 (2017)

[105] Pinheiro, P.O., Collobert, R., Dollár, P.: Learning to segment object candidates. In: Conference on Neural Information Processing Systems (NeurIPS) (2015)

[106] Pohlen, T., Hermans, A., Mathias, M., Leibe, B.: Full-resolution residual networks for semantic segmentation in street scenes. In: Conference on Computer Vision and Pattern Recognition (CVPR). pp. 4151–4160 (2017)

[107] Qi, X., Liao, R., Liu, Z., Urtasun, R., Jia, J.: Geonet: Geometric neural network for joint depth and surface normal estimation. In: Conference on Computer Vision and Pattern Recognition (CVPR). pp. 283–291 (2018)

[108] Qiu, X., Xiao, Y., Wang, C., Marlet, R.: Pixel-Pair occlusion relationship map (P2ORM): Formulation, inference & application. In: European Conference on Computer Vision (ECCV) (2020)

[109] 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: International Conference on Computer Vision (ICCV) (2017)

[110] Rafi, U., Gall, J., Leibe, B.: A semantic occlusion model for human pose estimation from a single depth image. In: Conference on Computer Vision and Pattern Recognition Worshops(CVPR Worshops). pp. 67–74 (2015)

[111] Ramamonjisoa, M., Du, Y., Lepetit, V.: Predicting sharp and accurate occlusion boundaries in monocular depth estimation using displacement fields. In: Conference on Computer Vision and Pattern Recognition (CVPR). pp. 14648–14657 (2020)

[112] Ramamonjisoa, M., Lepetit, V.: Sharpnet: Fast and accurate recovery of occluding contours in monocular depth estimation. In: International Conference on Computer Vision Workshops (ICCV Workshops) (2019)

[113] Raskar, R., Tan, K.H., Feris, R., Yu, J., Turk, M.: Non-photorealistic camera: depth edge detection and stylized rendering using multi-flash imaging. ACM Transactions on Graphics (TOG) **23**(3), 679–688 (2004)

[114] Redmon, J., Divvala, S., Girshick, R., Farhadi, A.: You only look once: Unified, real-time object detection. In: Conference on Computer Vision and Pattern Recognition (CVPR). pp. 779–788 (2016)

[115] Redmon, J., Farhadi, A.: Yolo9000: better, faster, stronger. In: Conference on Computer Vision and Pattern Recognition (CVPR). pp. 7263–7271 (2017)

[116] Ren, X., Fowlkes, C.C., Malik, J.: Figure/ground assignment in natural images. In: European Conference on Computer Vision (ECCV). pp. 614–627. Springer (2006)

[117] Ricci, E., Ouyang, W., Wang, X., Sebe, N., et al.: Monocular depth estimation using multi-scale continuous CRFs as sequential deep networks. IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI) **41**(6), 1426–1440 (2018)

[118] Roberts, L.G.: Machine perception of three-dimensional solids. Ph.D. thesis, Massachusetts Institute of Technology (1963)

[119] Ronneberger, O., Fischer, P., Brox, T.: U-Net: Convolutional networks for biomedical image segmentation. In: International Conference on Medical Image Computing & Computer Assisted Intervention (MICCAI) (2015)

[120] Rublee, E., Rabaud, V., Konolige, K., Bradski, G.: Orb: An efficient alternative to sift or surf. In: International Conference on Computer Vision (ICCV). pp. 2564–2571. Ieee (2011)

[121] Simonyan, K., Zisserman, A.: Very deep convolutional networks for large-scale image recognition. In: International Conference on Learning Representations (ICLR) (2015)

[122] Song, Y., Ma, C., Wu, X., Gong, L., Bao, L., Zuo, W., Shen, C., Lau, R.W., Yang, M.H.: Vital: Visual tracking via adversarial learning. In: Conference on Computer Vision and Pattern Recognition (CVPR). pp. 8990–8999 (2018)

[123] Stein, A.N., Hebert, M.: Occlusion boundaries from motion: Low-level detection and mid-level reasoning. International Journal of Computer Vision (IJCV) **82**, 325–357 (2008)

[124] Su, H., Jampani, V., Sun, D., Gallo, O., Learned-Miller, E., Kautz, J.: Pixel-adaptive convolutional neural networks. In: Conference on Computer Vision and Pattern Recognition (CVPR). pp. 11166–11175 (2019)

[125] Su, H., Qi, C.R., Li, Y., Guibas, L.J.: Render for CNN: Viewpoint estimation in images using CNNs trained with rendered 3D model views. In: International Conference on Computer Vision (ICCV) (2015)

[126] Sugihara, K.: Machine interpretation of line drawings, vol. 1. MIT press Cambridge (1986)

[127] Sun, X., Wu, J., Zhang, X., Zhang, Z., Zhang, C., Xue, T., Tenenbaum, J.B., Freeman, W.T.: Pix3D: Dataset and methods for single-image 3d shape modeling. In: Conference on Computer Vision and Pattern Recognition (CVPR) (2018)

[128] Sutskever, I., Martens, J., Dahl, G., Hinton, G.: On the importance of initialization and momentum in deep learning. In: International Conference on Machine Learning (ICML). pp. 1139–1147 (2013)

[129] Tao, R., Gavves, E., Smeulders, A.W.: Siamese instance search for tracking. In: Conference on Computer Vision and Pattern Recognition (CVPR). pp. 1420–1429 (2016)

[130] Tekin, B., Sinha, S.N., Fua, P.: Real-time seamless single shot 6D object pose prediction. In: Conference on Computer Vision and Pattern Recognition (CVPR) (2018)

[131] Teo, C., Fermuller, C., Aloimonos, Y.: Fast 2D border ownership assignment. In: Conference on Computer Vision and Pattern Recognition (CVPR). pp. 5117–5125 (2015)

[132] Tjaden, H., Schwanecke, U., Schömer, E.: Real-time monocular pose estimation of 3D objects using temporally consistent local color histograms. In: International Conference on Computer Vision (ICCV) (2017)

[133] Tola, E., Lepetit, V., Fua, P.: Daisy: An efficient dense descriptor applied to wide-baseline stereo. IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI) (2010)

[134] Tomasi, C., Manduchi, R.: Bilateral filtering for gray and color images. In: International Conference on Computer Vision (ICCV). pp. 839–846 (1998)

[135] Tulsiani, S., Malik, J.: Viewpoints and keypoints. In: Conference on Computer Vision and Pattern Recognition (CVPR) (2015)

[136] Vedaldi, A., Lenc, K.: Matconvnet – convolutional neural networks for matlab. In: ACM International Conference on Multimedia (ACM-MM) (2015)

[137] Voigtlaender, P., Leibe, B.: Online adaptation of convolutional neural networks for video object segmentation. British Machine Vision Conference (BMVC) (2017)

[138] Wang, G., Liang, X., Li, F.W.B.: DOOBNet: Deep object occlusion boundary detection from an image. In: Asian Conference on Computer Vision (ACCV) (2018)

[139] Wang, N., Yeung, D.Y.: Learning a deep compact image representation for visual tracking. In: Conference on Neural Information Processing Systems (NeurIPS). pp. 809–817 (2013)

[140] Wang, P., Shen, X., Russell, B., Cohen, S., Price, B., Yuille, A.L.: Surge: Surface regularized geometry estimation from a single image. In: Conference on Neural Information Processing Systems (NeurIPS). pp. 172–180 (2016)

[141] Wang, P., Yuille, A.: DOC: Deep occlusion estimation from a single image. In: European Conference on Computer Vision (ECCV) (2016)

[142] Wang, Q., Gao, J., Xing, J., Zhang, M., Hu, W.: Dcfnet: Discriminant correlation filters network for visual tracking. arXiv:1704.04057 (2017)

[143] Wang, Y., Yang, Y., Yang, Z., Zhao, L., Wang, P., Xu, W.: Occlusion aware unsupervised learning of optical flow. In: Conference on Computer Vision and Pattern Recognition (CVPR). pp. 4884–4893 (2018)

[144] Wang, Z., Bovik, A.C., Sheikh, H.R., Simoncelli, E.P.: Image quality assessment: from error visibility to structural similarity. IEEE Transactions on Image Processing (TIP) **13**(4), 600–612 (2004)

[145] Wu, H., Zheng, S., Zhang, J., Huang, K.: Fast end-to-end trainable guided filter. In: Conference on Computer Vision and Pattern Recognition (CVPR). pp. 1838–1847 (2018)

[146] Wu, Y., Lim, J., Yang, M.H.: Online object tracking: A benchmark. In: Conference on Computer Vision and Pattern Recognition (CVPR). pp. 2411–2418 (2013)

[147] Wu, Z., Song, S., Khosla, A., Yu, F., Zhang, L., Tang, X., Xiao, J.: 3d shapenets: A deep representation for volumetric shapes. In: Conference on Computer Vision and Pattern Recognition (CVPR). pp. 1912–1920 (2015)

[148] Xiang, Y., Kim, W., Chen, W., Ji, J., Choy, C., Su, H., Mottaghi, R., Guibas, L., Savarese, S.: ObjectNet3D: A large scale database for 3D object recognition. In: European Conference on Computer Vision (ECCV) (2016)

[149] Xiang, Y., Mottaghi, R., Savarese, S.: Beyond PASCAL: A benchmark for 3D object detection in the wild. In: Winter Conference on Applications of Computer Vision (WACV) (2014)

[150] Xiang, Y., Schmidt, T., Narayanan, V., Fox, D.: PoseCNN: A convolutional neural network for 6D object pose estimation in cluttered scenes. In: Robotics: Science and Systems (RSS) (2018)

[151] Xiao, J., Hays, J., Ehinger, K.A., Oliva, A., Torralba, A.: SUN database: Large-scale scene recognition from abbey to zoo. In: Conference on Computer Vision and Pattern Recognition (CVPR) (2010)

[152] Xiao, Y., Qiu, X., Langlois, P., Aubry, M., Marlet, R.: Pose from shape: Deep pose estimation for arbitrary 3D objects. In: British Machine Vision Conference (BMVC) (2019)

[153] Xie, S., Tu, Z.: Holistically-nested edge detection. In: International Conference on Computer Vision (ICCV). pp. 1395–1403 (2015)

[154] Xu, D., Ricci, E., Ouyang, W., Wang, X., Sebe, N.: Multi-scale continuous CRFs as sequential deep networks for monocular depth estimation. In: Conference on Computer Vision and Pattern Recognition (CVPR). pp. 5354–5362 (2017)

[155] Yang, L., Wang, Y., Xiong, X., Yang, J., Katsaggelos, A.K.: Efficient video object segmentation via network modulation. In: Conference on Computer Vision and Pattern Recognition (CVPR). pp. 6499–6507 (2018)

[156] Yin, W., Liu, Y., Shen, C., Yan, Y.: Enforcing geometric constraints of virtual normal for depth prediction. In: International Conference on Computer Vision (ICCV) (2019)

[157] Yu, F., Koltun, V.: Multi-scale context aggregation by dilated convolutions. International Conference on Learning Representations (ICLR) (2015)

[158] Yu, Z., Feng, C., Liu, M.Y., Ramalingam, S.: CASENet: Deep category-aware semantic edge detection. In: Conference on Computer Vision and Pattern Recognition (CVPR). pp. 5964–5973 (2017)

[159] Yu, Z., Liu, W., Zou, Y., Feng, C., Ramalingam, S., Vijaya Kumar, B., Kautz, J.: Simultaneous edge alignment and learning. In: European Conference on Computer Vision (ECCV). pp. 388–404 (2018)

[160] Zhang, J., Ma, S., Sclaroff, S.: Meem: robust tracking via multiple experts using entropy minimization. In: European Conference on Computer Vision (ECCV). pp. 188–203. Springer (2014)

[161] Zheng, C., Cham, T.J., Cai, J.: T2Net: Synthetic-to-realistic translation for solving single-image depth estimation tasks. In: European Conference on Computer Vision (ECCV). pp. 767–783 (2018)

[162] Zheng, Y., Kuang, Y., Sugimoto, S., Astrom, K., Okutomi, M.: Revisiting the PnP problem: A fast, general and optimal solution. In: International Conference on Computer Vision (ICCV) (2013)

[163] Zhou, X., Karpur, A., Luo, L., Huang, Q.: Starmap for category-agnostic keypoint and viewpoint estimation. In: European Conference on Computer Vision (ECCV) (2018)

[164] Zitnick, C.L., Kanade, T.: A cooperative algorithm for stereo matching and occlusion detection. IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI) **22**(7), 675–684 (2000)