ROBUST SCENE UNDERSTANDING IN CHALLENGING SCENARIOS

by

TUAN-ANH VU

A Thesis Submitted to
The Hong Kong University of Science and Technology
in Partial Fulfillment of the Requirements for
the Degree of Doctor of Philosophy
in Computer Science and Engineering

August 2024, Hong Kong

Authorization

I hereby declare that I am the sole author of the thesis.

I authorize the Hong Kong University of Science and Technology to lend this thesis to other institutions or individuals for the purpose of scholarly research.

I further authorize the Hong Kong University of Science and Technology to reproduce the thesis by photocopying or by other means, in total or in part, at the request of other institutions or individuals for the purpose of scholarly research.

_____

TUAN-ANH VU

ROBUST SCENE UNDERSTANDING IN CHALLENGING SCENARIOS

by

TUAN-ANH VU

This is to certify that I have examined the above Ph.D. thesis

and have found that it is complete and satisfactory in all respects,

and that any and all revisions required by

the thesis examination committee have been made.

Prof. Sai-Kit Yeung, Thesis Supervisor

Prof. Xiaofang Zhou, Head of Department

Department of Computer Science and Engineering

20 August 2024

ACKNOWLEDGMENTS

As I reflect on my PhD journey, I am deeply grateful for the remarkable individuals and groups who have played a significant role in making this experience unforgettable.

Advisors. I wish to express my deepest gratitude to my PhD supervisor, Prof. Sai-Kit Yeung, for offering me the incredible opportunity to pursue my PhD at one of the world's leading research groups. Your trust in me and the autonomy you granted allowed me to approach my research with creativity and independence. I am also immensely thankful to my other advisors, Prof. Duc Thanh Nguyen (Deakin University) and Prof. Binh-Son Hua (Trinity College Dublin). Your remarkable vision in identifying key research questions and your insightful critiques have consistently ensured that our work remained focused, ultimately leading to research of the highest quality.

Collaborators. I am deeply indebted to all my brilliant collaborators for the invaluable discussions that have served as lessons and inspirations throughout this journey. Thank you, Ziqiang, Hai, Susan, Sang, Trung, Jason, Huajian (HKUST VGD); Prof. Zhiyuan Zhang (Singapore Management University); Quang-Hieu Pham (Woven by Toyota USA); Nhat Chung, E-Ro, Prof. Minh-Triet Tran (Vietnam National University). Your sincere advice, both within and beyond the realm of research, has been invaluable.

Special thanks go to Ziqiang and Hai, my cherished collaborators. Whenever I faced doubts, you always provided multiple solutions. During challenging times, your comforting words and encouragement were a guiding light, turning obstacles into opportunities.

My internships and research attachments were made truly unforgettable thanks to the immense support from numerous collaborators, especially Prof. Guo Qing from CFAR, ASTAR and NUS; Prof. Ivor W. Tsang, also from CFAR, ASTAR and NTU; and Colin Yap and Shirley Wong for their tireless support. I also want to thank to Yunrui, Sensen, ZhangJie from CFAR, A*STAR for their discussions and collaborations.

Colleagues, Friends, and Family. Reflecting on these five years of my PhD journey, I realize that research is only part of the story. I would not have come this far without the unwavering support of incredible colleagues, dear friends, and my loving family.

My heartfelt gratitude goes to my peers at HKUST VGD – Jaeyeon, Dung, Nhan, Mendis, Hilary, Yipeng. Our countless brainstorming sessions, inspiring discussions, coffee breaks, and late-night

gatherings are memories I will treasure forever. VGD has been more than just a research group to me; it has been a home where I've formed special bonds.

Outside the lab, my life would have been incomplete without the wonderful friends from HKUST and CFAR, A*STAR. I cherish all the precious moments shared with Prof. Howard, Prof. Tuan Anh, Prof. Quoc, Huong, Nu, NhuY, Long, Duc, Truc, Quan, Khoi, Giang, Minh, Kenny, Bohao, Yun, Molly, Gabriel, CaoYue, Houyang, YangYing, Chen, Bin, and many more.

Finally, to my beloved family, your unwavering love and faith in me have been my greatest source of strength. Your kindness, tolerance, support, and patience have shaped me into the person I am today.

TABLE OF CONTENTS

ROBUST SCENE UNDERSTANDING IN CHALLENGING SCENARIOS

by

TUAN-ANH VU

Department of Computer Science and Engineering

The Hong Kong University of Science and Technology

ABSTRACT

In recent years, computer vision and graphics fields have witnessed significant progress with the emergence of novel techniques and architectures to address complex challenges. This thesis presents novel methodologies and advancements to enhance the accuracy and robustness of current methodologies in scene understanding. The study focuses on developing algorithms for 2D transparent object segmentation in challenging indoor environments, aiming to significantly improve detection capabilities in scenarios with cluttered backgrounds and variable lighting conditions.

Additionally, it explores 3D test time augmentation methods for classification and segmentation, targeting both static and dynamic objects in indoor and outdoor scenes to enhance model resilience and accuracy. By implementing these methods, the research aims to provide robust solutions that adapt to variations in data, thereby improving the overall performance of 3D models.

Furthermore, the research combines 3D reconstruction with motion flow estimation to achieve a comprehensive understanding of dynamic objects, such as humans and animals, in indoor settings. This approach aims to accurately track and predict object movements, enhancing the analysis of dynamic scenes.

The study also utilizes text-to-image diffusion models for 2D open vocabulary camouflage instance segmentation, addressing the detection and segmentation of camouflaged objects in outdoor

and underwater environments. This method leverages advanced diffusion models to identify and segment camouflaged instances from diverse vocabularies, improving the detection and analysis of such objects in challenging settings.

Lastly, the creation of a comprehensive dataset for camouflaged animals in videos aims to improve classification, detection, and segmentation algorithms. This dataset includes outdoor and underwater videos of camouflaged animals, facilitating the development of robust algorithms capable of accurately analyzing these animals in their natural habitats. This effort contributes to wildlife conservation and environmental monitoring by providing a valuable resource for further research.

In summary, this research pushes the boundaries of scene understanding by offering more accurate and robust solutions for interpreting complex scenes. By addressing specific challenges associated with different environments and object types, it aims to significantly advance the field of computer vision.

CHAPTER 1

INTRODUCTION

Understanding complex scenes has dramatically improved with progress in computer vision and 3D scene interpretation in recent years. However, scene understanding is still a challenging task, especially in complex scenarios. This exploration of scene understanding covers several critical dimensions, each with unique challenges that push current methodologies to their limits. These advancements aim to address the limitations of current scene understanding techniques and provide more accurate and robust results.

This research addresses several crucial areas in the field of scene understanding, focusing on specific tasks and environments to enhance the accuracy and robustness of current methodologies as shown in Figure 1.1. These topics include:

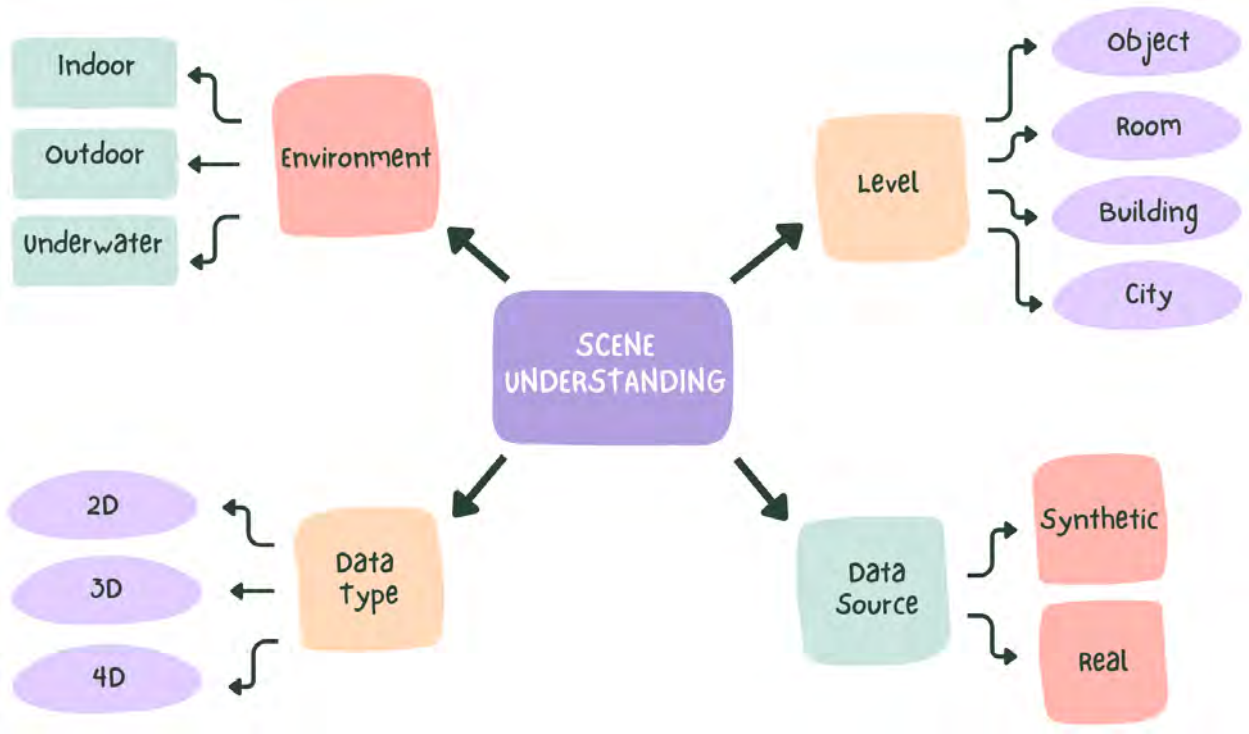


Figure 1.1: Research overview of this thesis.

- **3D Test Time Augmentation Method for Classification and Segmentation:** Enhancing 3D scene understanding involves robust classification and segmentation techniques that can adapt to both static and dynamic environments. This topic covers indoor scenes with static real and synthetic objects, as well as outdoor scenes with dynamic objects and changing backgrounds. By implementing test time augmentation methods, the research aims to improve the performance of 3D models, making them more resilient to variations in the data and enabling more accurate scene interpretation.

- **4D Joint Reconstruction and Flow Estimation for Dynamic Objects:** Understanding dynamic objects, such as humans and animals, in 4D (spatial and temporal dimensions) is crucial for applications that involve motion analysis and interaction. This research focuses on joint reconstruction and flow estimation in indoor environments with dynamic objects. By combining 3D reconstruction techniques with motion flow estimation, the aim is to achieve a comprehensive understanding of dynamic scenes, allowing for more accurate tracking and prediction of object movements.

- **2D Transparent Object Segmentation:** Transparent object segmentation remains one of the most challenging tasks in computer vision due to the nature of transparent materials that lack color and texture. This research focuses on indoor environments with difficult scenarios, such as cluttered backgrounds and varying lighting conditions. By developing advanced segmentation algorithms, the goal is to improve the detection and understanding of transparent objects in 2D images, enhancing the ability to interact with and manipulate these objects in practical applications.

- **2D Open Vocabulary Camouflage Instance Segmentation Using Text-to-Image Diffusion Models:** Camouflaged object detection is a significant challenge in outdoor and underwater environments due to the objects' ability to blend seamlessly with their surroundings. This research explores the use of text-to-image diffusion models to perform open vocabulary instance segmentation of camouflaged objects, including animals and some humans. By leveraging advanced diffusion models, the goal is to develop a system that can accurately identify and segment camouflaged instances from diverse vocabularies, improving the detection and analysis of such objects in challenging environments.

- **Video Camouflaged Animal Understanding Dataset:** The creation of a specialized dataset

for camouflaged animal understanding in videos is essential for advancing the field of video analysis and scene understanding. This dataset will include outdoor and underwater videos of camouflaged animals, focusing on tasks such as classification, detection, and segmentation. By providing a comprehensive dataset, the research aims to facilitate the development of more robust algorithms capable of accurately analyzing camouflaged animals in their natural habitats, contributing to wildlife conservation and environmental monitoring efforts.

Each of these topics represents a frontier in scene understanding research, addressing specific challenges associated with different environments and object types as shown in Table 1.1. By advancing methodologies in these areas, this research aims to push the boundaries of what is possible in computer vision, providing more accurate and reliable tools for interpreting complex scenes. Firstly, we work on 3D classification and segmentation, but we found that human is dynamic, deformable, which is challenging. So, we study human and animal shape and motion. Next, we observed that human can interact well with special objects like transparent, mirror, but robot or perception system mostly failed. Therefore, we work on 2D transparent and reflective object segmentation. After successfully work with transparent objects, we realize that transparent is one of the camouflage techniques. So, we move to recognize and segment camouflage objects (including animals and human). Lastly, instead of focus on visual appearance of camouflaged object, we actually can base on their motion (need multiple frames or video). However, there is only some small-scale datasets for video camouflage understanding. Therefore, we propose our video dataset to fill the gap. The remainder of this thesis is organized as follows:

Chapter 2 of this thesis investigates data augmentation, a powerful technique extensively used in 2D deep learning but relatively overlooked in 3D deep learning. Sparse representation and low point density of 3D shapes pose significant challenges, leading to diminished performance in downstream tasks. To address this, we explore the potential of test-time augmentation (TTA) for 3D point clouds. Drawing inspiration from recent advances in learning implicit representations and point cloud upsampling, this chapter proposes a systematic approach to augment point cloud data. By leveraging implicit field reconstruction and point cloud upsampling techniques, test-time augmented data is generated by sampling points from the reconstructed results. Extensive experiments on the ModelNet40, ShapeNet, ScanObjectNN, S3DIS, and SemanticKITTI datasets reveal that both test-time augmentation strategies improve accuracy. Notably, point cloud upsampling demonstrates significant performance gains for object classification and segmentation tasks, particularly on sparse

Table 1.1: The tasks of scene understanding and its corresponding challenging scenarios.

Scene Understanding	**Challenging Scenarios**
3D Classification and Segmentation	Dynamic and Moving Objects
4D (3D + Time) Reconstruction	Dynamic, Deformable Objects
2D Semantic Segmentation	Transparent and Reflective Objects
2D Instance Segmentation	Camouflage Human and Animals
3D / Video (2D + Time) Classification, Detection, Segmentation	Camouflage Animals, Category Diversity

point clouds.

Chapter 3 delves into object reconstruction from 4D point clouds, a domain often overlooked despite the impressive progress in 3D point cloud reconstruction. The novelty of this chapter lies in the proposal of a novel network architecture called RFNet-4D++. This architecture simultaneously reconstructs objects, and their motion flows from 4D point clouds, leveraging spatial and temporal features to boost overall performance. A key element of RFNet-4D++ is the introduction of a temporal vector field learning module that employs an unsupervised learning approach for flow estimation combined with supervised learning of spatial structures for object reconstruction. The presented experiments on benchmark datasets demonstrate the efficacy of RFNet-4D++, showcasing state-of-the-art performance for flow estimation and object reconstruction, all while significantly reducing training and inference times.

Chapter 4 shifts the focus towards the challenges in semantic scene understanding for transparent objects in computer vision. Glass is a ubiquitous material in modern household and industrial applications. Yet, scene-understanding tasks often treat it as an opaque entity. This chapter introduces a pioneering approach for transparent object segmentation from a single color image using a pyramidal transformer encoder-decoder architecture. Two novel object cues are presented to enhance the segmentation process's accuracy. First, the Boundary Feature Aware (BFA) module incorporates a novel boundary loss to learn and integrate glass boundary features, enabling precise localization and segmentation of glass-like regions. Second, the Reflection Region Aware (RRA) module decomposes reflection into foreground and background layers, thereby providing the network with additional features to distinguish between glass-like and non-glass areas. Thorough experimental

evaluations on the Trans10K-v2 and Stanford2D3D datasets validate the effectiveness and efficiency of the proposed method, showcasing its superiority over state-of-the-art techniques with a remarkable +6.23% mIoU improvement on Trans10K-v2 dataset and +10.1% mIoU on Stanford2D3D dataset.

Chapter 5 leverages text-to-image diffusion techniques for open-vocabulary camouflaged instance segmentation. The text-to-image diffusion techniques have shown exceptional capabilities of producing high-quality, dense visual predictions from open-vocabulary text. This indicates a strong correlation between visual and textual domains in open concepts and that diffusion-based text-image discriminative models can capture richly diverse information for effective segmentation in the wild. However, we found that those advantages are more difficult to hold true for camouflaged individuals because of the significant blending between their visual boundaries and their surroundings. In this work, while leveraging the benefits of diffusion-based text-to-image models for open-vocabulary performance, we aim to address a challenging problem in computer vision: camouflaged instance segmentation. Specifically, we propose a method built upon a state-of-the-art diffusion model empowered by open-vocabulary to learn multi-scale textual-visual features for camouflaged object representations. Such cross-domain representations are desirable in segmenting camouflaged objects where visual cues are subtly to distinguish the objects from the background, especially in segmenting novel objects not seen in training. We also develop technically supportive components to fuse cross-domain features effectively and engage relevant features towards respective foreground objects. We validate our method and compare it with existing ones on several benchmark datasets of camouflaged instance segmentation and generic open-vocabulary instance segmentation. Experimental results confirm the advances of our method over existing ones.

Chapter 6 introduce a new video camouflaged animal understanding dataset. We have been witnessing remarkable success led by the power of neural networks driven by a significant scale of training data in handling various computer vision tasks. However, less attention has been paid to monitoring the camouflaged animals, the masters of hiding themselves in the background. Performing robust and precise camouflaged animal segmentation is not trivial even for domain experts because of their consistent appearance with backgrounds. Even though several efforts were made to perform camouflaged animal image segmentation, there is only some work on camouflaged animal video segmentation to the best of the author's knowledge. Biologists usually favor videos with redundant information and temporal consistencies to perform biological monitoring and understanding of the behavior and events of animals. The scarcity of such labeled video data

is the most hindering issue. To address these challenges, we present **CamoVid60K**, a diverse, large-scale, and accurately annotated video dataset of camouflaged animals. This dataset comprises **218** videos with **62,774** finely annotated frames, covering **70** animal categories, which *surpasses* all previous datasets in terms of the number of videos/frames and species included. CamoVid60K also offers more diverse downstream tasks in CV, such as camouflaged animal classification, detection, and task-specific segmentation (semantic, referring, motion), *etc*. We have benchmarked several state-of-the-art algorithms on the proposed CamoVid60K dataset, and the experimental results provide valuable insights into future research directions. Our dataset stands as a novel and challenging testing set to stimulate more powerful camouflaged animal video segmentation algorithms, and there is still a large room for further improvement.

This thesis includes the following 6 publications [272, 268, 269, 271, 270, 273]

- ***Test-Time Augmentation for 3D Point Cloud Classification and Segmentation***
  Tuan-Anh Vu*, Srinjay Sarkar*, Zhiyuan Zhang, Binh-Son Hua, Sai-Kit Yeung
  International Conference on 3D Vision (3DV), 2024
- ***RFNet-4D: Joint Object Reconstruction and Flow Estimation from 4D Point Clouds***
  Tuan-Anh Vu, Duc-Thanh Nguyen, Binh-Son Hua, Quang-Hieu Pham, Sai-Kit Yeung
  European Conference on Computer Vision (ECCV), 2022 (**Oral**)
- ***RFNet-4D++: Joint Object Reconstruction and Flow Estimation from 4D Point Clouds with Cross-Attention Spatio-Temporal Features***
  Tuan-Anh Vu, Duc-Thanh Nguyen, Binh-Son Hua, Quang-Hieu Pham, Sai-Kit Yeung
  Under Review
- ***Power of Boundary and Reflection: Semantic Transparent Object Segmentation using Pyramid Vision Transformer with Transparent Cues***
  Tuan-Anh Vu, Hai Nguyen-Truong, Ziqiang Zheng, Binh-Son Hua, Qing Guo, Ivor W. Tsang, Sai-Kit Yeung
  Under Review
- ***Catch Me If You Can Describe Me: Open-Vocabulary Camouflaged Instance Segmentation with Diffusion***
  Tuan-Anh Vu, Duc-Thanh Nguyen, Nhat Minh Chung, Qing Guo, Binh-Son Hua, Ivor W. Tsang, Sai-Kit Yeung
  Under Review

- ***CamoVid60K: A Large-Scale Video Dataset for Moving Camouflaged Animals Understanding***
  Tuan-Anh Vu, Ziqiang Zheng, Chengyang Song, Qing Guo, Ivor W. Tsang, Sai-Kit Yeung
  Under Review

In addition, I contributed to 9 papers [351, 26, 259, 267, 325, 86, 200, 361, 353] during PhD but are not included in this thesis:

- ***MarineInst: A Foundation Model for Marine Image Analysis with Instance Visual Description***
  Ziqiang Zheng, Yiwei Chen, Huimin Zeng, **Tuan-Anh Vu**, Binh-Son Hua, Sai-Kit Yeung
  European Conference on Computer Vision (ECCV) 2024. (**Oral**)
- ***StyleCity: Large-Scale 3D Urban Scenes Stylization***
  Yingshu Chen, Huajian Huang#, **Tuan-Anh Vu**, Ka-Chun Shum, Sai-Kit Yeung
  European Conference on Computer Vision (ECCV) 2024.
- ***MarineVRS: Marine Video Retrieval System with Explainability via Semantic Understanding***
  Tan-Sang Ha*, Hai Nguyen-Truong*, **Tuan-Anh Vu**#, Sai-Kit Yeung
  OCEANS 2023, Limerick (**Oral**)
- ***Marine Video Kit: A New Marine Video Dataset for Content-based Analysis and Retrieval***
  Quang-Trung Truong, **Tuan-Anh Vu**, Tan-Sang Ha, Jakub Lokoc, Ajay Joneja, Sai-Kit Yeung
  29th International Conference on Multimedia Modeling (MMM), 2023. **Oral**
- ***Time-of-Day Neural Style Transfer for Architectural Photographs***
  Yingshu Chen, **Tuan-Anh Vu**, Binh-Son Hua, Sai-Kit Yeung
  IEEE International Conference on Computational Photography (ICCP), 2022. (**Oral**)
- ***Improving Referring Image Segmentation using Vision-Aware Text Features***
  Hai Nguyen-Truong*, E-Ro Nguyen*, **Tuan-Anh Vu**#, Binh-Son Hua, Minh-Triet Tran, Sai-Kit Yeung
  Under review
- ***Toward Transferable Attack Against Vision-LLM in Autonomous Driving with Typography***
  Nhat Minh Chung, Sensen Gao, **Tuan-Anh Vu**, Jie Zhang, Aishan Liu, Yun Lin, Jin Song Dong, Qing Guo
  In submission

- ***MarineGPT: Unlocking Secrets of "Ocean" to the Public***
  Ziqiang Zheng, Jipeng Zhang, **Tuan-Anh Vu**, Shizhe Diao, Sai-Kit Yeung
  In submission
- ***Exploring Boundary of GPT-4V on Marine Analysis: A Preliminary Case Study***
  Ziqiang Zheng, Yiwei Chen, Jipeng Zhang, **Tuan-Anh Vu**, Huimin Zeng, Yue W. Tim, Sai-Kit Yeung
  In submission

CHAPTER 2

ROBUST INFERENCE USING TEST TIME AUGMENTATION

2.1 Introduction

Point-based representation is of great importance to computer graphics and computer vision. In the modern era of deep learning, neural networks can be designed to learn features from point clouds, facilitating 3D perception tasks such as object classification, object detection, and semantic segmentation in many downstream applications. Nevertheless, such evolutions still leave 3D perception a challenging and unsolved problem. A typical disadvantage of point-based representation is that surface information is implied by point density and orientation, if any. Due to such ambiguity, techniques for data augmentation on point clouds are relatively scarce and challenging to design.

Recent advances in using neural networks to represent 3D data have opened new opportunities to revise and explore this problem from a new perspective [191, 212, 298]. One type of method is the so-called neural implicit representation based on the idea of training a neural network that can return queries of the 3D space from input coordinates [207, 191, 246, 212]. Particularly, one can train a neural network to encode a 3D point to various attributes such as occupancy, color, or a general feature vector. The power of a neural implicit representation is that the queries can be performed at arbitrary points, and no special mechanism is required for value interpolation. Another type of methods [329, 154, 298] employs upsampling to achieve both distribution uniformity and proximity-to-surface. The advantages of the upsampling-based method lie in self-supervision and more uniformly distributed dense representation without the need of surface ground truth.

In this work, we investigate both types of strategies and leverage them as a systematic way for data augmentation at test time. Particularly, for implicit representation, we leverage the convolutional occupancy network [212] to encode the 3D point clouds to a regular grid representation that allows the interpolation of features at an arbitrary location. For the upsampling-based method, we employ the self-upsampling method [298] to obtain a dense and uniformly distributed proximity-to-surface point cloud. We propose an effective technique to aggregate features of the original and augmented

point clouds to generate the final prediction. We select the task of object classification and semantic segmentation as the downstream task to validate our augmentation technique, as they play a key role in many practical applications, including perception in robotics and autonomous driving. We experiment with point cloud data from ModelNet40 [303], ShapeNet [19], ScanObjectNN [275], S3DIS [4] and SemanticKITTI [8] dataset, which demonstrates significant performance improvement.

In summary, our key contributions are:

- We analyze and compare existing reconstruction approaches, including surface-based sampling and point cloud upsampling for test-time augmentation.
- We propose a test-time augmentation method for 3D point cloud deep learning, which is suitable for both approaches;
- We identified a self-supervised point cloud upsampling method or surface-based sampling as a robust method for our test-time augmentation. It uses the proximity-to-surface cues to sample augmented point clouds.
- Extensive experiments and analysis prove the effectiveness of our augmentation method on two downstream tasks, including object classification and semantic segmentation on synthetic and real-world datasets.

2.2 Related Works

2.2.1 3D Deep Learning

Early methods usually convert the irregular and sparse 3D points into multiple regular 2D views [311, 120, 43] or 3D voxels [303, 295, 104, 42, 263]. Despite the improvements gained by performing CNN on these regular structures, these methods usually suffer information loss and high computational costs.

Point cloud is a universal representation for 3D data. PointNet [217] is the pioneering work that can process 3D points directly by symmetric functions and max-pooling to extract global features. To capture local features, PointNet++ [219] performs hierarchical PointNets on different scales. In recent years, various convolution operators and networks have been proposed for point clouds, such as PointCNN [157], SpiderCNN [315], DGCNN [293], and ShellNet [341] with the supreme performance achieved on classification, retrieval, and segmentation tasks. There are also approaches [105, 287, 80, 101, 312] specially designed for semantic segmentation. Huang *et al*. [105]

learn the local structure, particularly for semantic segmentation, by applying learning algorithms from recurrent neural networks. SPG [144] constructs graphs between coarsely segmented super-points for large-scale point cloud semantic segmentation.

To balance the efficiency and accuracy, hybrid works [145, 173, 334] utilize the characteristics of multiple representations. PointGrid [145] assigns fix number of points in each grid cell, making the conventional CNN feasible. While it runs fast, the accuracy is still not high. PVCNN [173] represents the input in points and performs the convolutions in voxels with a superior performance achieved than sole point or voxel representations. To handle large-scale lidar point clouds, FusionNet [334] divides the input into voxels and extracts features from both voxels and inner points.

2.2.2 Neural 3D Reconstruction

3D reconstruction works can be classified into four categories in terms of the representation of the output: voxel-based, point-based, mesh-based, and implicit function-based methods.

Similar to semantic segmentation, voxel is also a popular representation for 3D reconstruction [38, 299, 303]. In the category, voxel grids are used to store either occupancy that encodes whether the voxel is occupied or not [303, 38] or SDF information that holds signed projective distances from voxels to the closest surfaces [44, 158, 252]. However, as mentioned in the segmentation works, such methods inherit the limitations of high memory costs.

Another line of works output point clouds directly for 3D reconstruction [69, 160, 215, 318]. These methods design generative models to produce dense points for scene representation. Despite the efficiency, the generated points cannot sufficiently represent complicated surfaces as there is no topology between the points.

Mesh is another popular output representation for 3D reconstruction. In this category, some work deform shapes with simple topology to more complicated shapes, which usually constrain to certain fixed templates [122, 225] or topologies [244, 9]. To reconstruct a shape of arbitrary topology, AtlasNet [82] warps multiple 2D planes into 3D shapes. Despite the superior results, this method can result in self-intersecting mesh faces.

To overcome the limitations of the above explicit representations (voxel, point, mesh), more recent works focus on implicit representations that employ occupancy [191, 212] and distance field [207, 17] with a neural network to infer an occupancy probability or distance value for the

input 3D points. As implicit representation models shape continuously, more detail is preserved, and more complicated shape topologies can be obtained. In this work, we also employ implicit representation to aim to augment a point cloud for various downstream tasks.

2.2.3 Point Cloud Upsampling.

Point cloud upsampling can produce a dense, uniform, and complete point cloud from a sparse and noisy complete with or without missing parts.

Traditional Point Cloud Upsampling. A seminal point cloud upsampling algorithm is to interpolate points as vertices of a Voronoi diagram [1]. [167] later proposed an algorithm by introducing the locally optimal linear projector for surface reconstruction and using it to project a set of points onto the input point cloud. This work was followed by [102], who proposed a weighted locally linear operator in order to make the point cloud distribution more even. [103] introduces an edge-aware resampling method by sampling points on the edge and calculating the normals at those points. All of the above-mentioned methods are not data-driven and thus they heavily rely on priors such as normal estimation.

Deep-Learning Based Point Cloud Upsampling. PU-Net [329] was the first deep learning-based point cloud upsampling method that used a multi-branch feature expansion module to extract multi-scale features and expand a point cloud in the feature space. This was followed by EC-Net [328], which achieves edge-aware point cloud upsampling by learning distance features obtained by the perturbation of the generated point cloud relative to the input point cloud. In this work, we propose to use point cloud upsampling as a test-time augmentation technique.

2.2.4 Data Augmentation and Test-Time Augmentation

In modern deep learning, large-scale data is often required for training deep neural networks; however, acquiring a large amount of data is a thorough and prohibitively expensive process. Data augmentation is a common but useful technique to scale up the data artificially. In image classification, popular data augmentation includes simple transformations of the images, including rotations, flipping, cropping, etc. [137, 100, 257, 243, 92, 100]. In 3D deep learning, traditional methods (e.g., PointNet [217] and PointNet++ [219]) utilize similarity transformations such as random rotations, scaling, and jittering for data augmentation during training. Similarity transformations are also

used to augment real-world data to build ScanObjectNN [275], a real-world dataset for object classification. Research efforts for more sophisticated augmentation techniques for 3D point clouds are relatively scarce. Recently, PointMixup [27] is proposed to mix two point clouds based on shortest path linear interpolation; PointAugment [153] uses adversarial learning to seek augmented point clouds satisfying a given classifier. PPBA [34] automates the design of augmentation policies for the specific task of 3D object detection. Apart from train-time data augmentation, in this work, we assume that pre-trained models for specific downstream tasks are already given and investigate test-time augmentation techniques [6] that can boost overall performance.

Particularly, Test-time augmentation (TTA) first transforms the input, then perform predictions on the augmented versions of the input, and finally combines the prediction results of both the input and augmented version ones. This strategy is common to image classification with simple augmentation policies such as flipping, cropping, and scaling [100]. More sophisticated methods involve learning and optimizing augmentation policies [235, 130, 182, 240], or learning to combine predictions [240]. Beyond images, TTA has also been applied to medical image segmentation [197, 280] and text recognition [152]. For point clouds, however, we are unaware of any recent method tailored to test-time augmentation and augmentation policies.

2.3 Our Method

2.3.1 Overview

Given a point set $\{\mathbf{p}_i\}_{i=1}^n$ with $\mathbf{p}_i \in \mathbb{R}^3$ represented by a matrix $\mathbf{x}_0 \in \mathbb{R}^{n\times 3}$. Without loss of generality, we assume at inference, $\mathbf{x}_0$ is passed to pre-trained network f for feature extraction, and the features are passed to a network g for final label prediction $f(\mathbf{x}_0)$. Our goal is to achieve performance improvement in the downstream task via test-time augmentation, where the final prediction can be defined as:

$$g(\phi(f(\mathbf{x}_0), f(\mathbf{x}_1), f(\mathbf{x}_2), ...)) \tag{2.1}$$

where ϕ is an aggregation function to combine multiple features resulting from the original point set $\mathbf{x}_0$ and the augmented point sets $\mathbf{x}_1$, $\mathbf{x}_2$, etc. Note that the network f and g are pre-trained and left untouched in test-time augmentation; only the input is augmented.

Traditionally, a simple method for test-time augmentation is jittering, which adds Gaussian noise

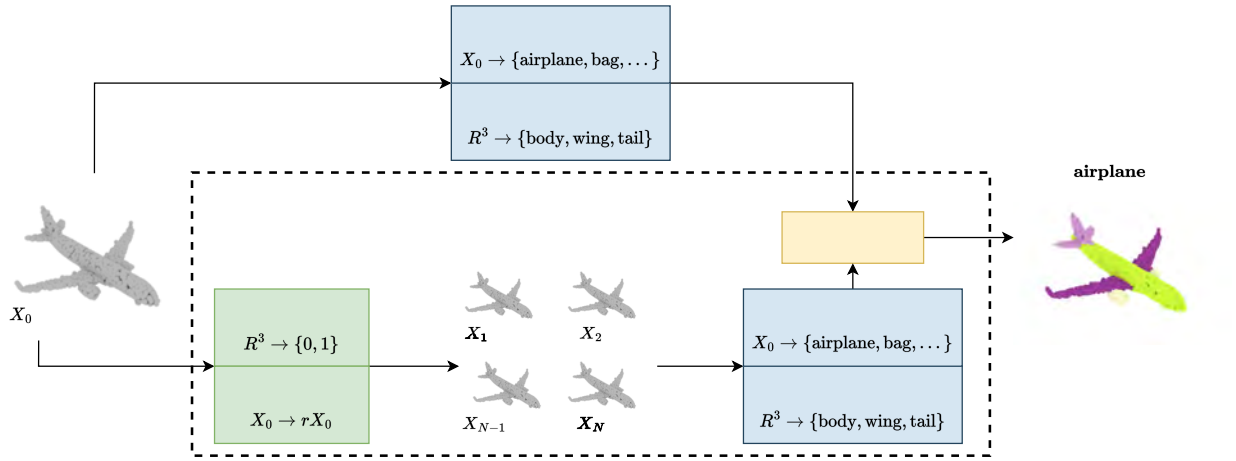


Figure 2.1: Illustration of our test-time augmentation method for point clouds downstream tasks such as classification and segmentation. We view the input point cloud as a noisy estimate of a latent surface and propose using an implicit field represented by an occupancy network or a point cloud upsampling network to sample augmented point clouds so that the point clouds share the same underlying surfaces. We then perform the downstream task on each point cloud and aggregate the point features to produce the final result.

to perturb the point cloud $\mathbf{x}_0$ to generate an augmented point cloud $\mathbf{x}_k$:

$$\mathbf{x}_k = \mathbf{x}_0 + \lambda \mathbf{z}_k \tag{2.2}$$

where $\mathbf{z}_k \sim \mathcal{N}(0, I)$ is a random noise vector from a normal distribution, and λ is a scalar value to control the noise magnitude. This simple augmentation has been widely adopted since the seminal PointNet [217]. An issue of such augmentation is that it does not consider the underlying surface or point distribution because the noise $\mathbf{z}$ is independent of $\mathbf{x}_0$, resulting in marginal performance improvement in many cases. In this work, we viewpoint set $\mathbf{x}_0$ as a noisy estimate of a latent surface representation $\mathcal{S}$, and therefore, we define point cloud augmentation as the process of sampling additional point clouds $\mathbf{x}_k$ that explain the same surface. We propose to sample augmented point clouds $\mathbf{x}_k$ ($k \geqslant 1$) in two ways: surface sampling and point cloud up-sampling. The sampled point clouds can then be leveraged for downstream tasks such as classification and segmentation. Our method is visualized in Figure 2.1.

In the following sections, we explain the technique for sampling augmented point clouds using an implicit representation network (Section 2.3.2) and a self-supervised point upsampling network (Section 2.3.3). We then present downstream tasks that leverage the proposed test-time augmentation and discuss feature aggregation and final label prediction for point cloud classification

and segmentation.

2.3.2 Augmentation by Implicit Field Reconstruction

We are motivated by the recent advances in geometry reconstruction using neural implicit representation. The basic idea is to learn a mapping $f_\theta : \mathbb{R}^3 \longrightarrow \{0, 1\}$ using a neural network parameterized by θ. This function implicitly encodes the geometry in the 3D space to allow the query of the occupancy at any point in the 3D space. To obtain the geometry explicitly, the Marching Cubes algorithm [175] can generate a triangle mesh containing surfaces at zero crossings in the implicit field. Our neural implicit field is built upon the convolutional occupancy network [191, 212]. The convolutional occupancy network uses a combination of convolutional and linear layers, thus endowing its features with equivariance and scalability. This enables the network to produce implicit representations for both single objects and large-scale scenes. Our implementation uses the network variant that stores features on a 3D regular grid. Figure 2.2 shows the comparison of different reconstruction methods, *e.g*. unsupervised [81], screened poisson [128] and supervised [212] reconstruction.

Encoder. The encoder is a shallow PointNet [217] but with local pooling layers. By using these input features generated by the local PointNet encoder, we obtain a 32^3 volumetric feature grid that captures the local information in the neighborhood of the points, which is necessary to capture local geometric information about the shape of the input point cloud. Due to memory constraints, the volumetric feature can represent rich 3D information but is restricted to small resolutions and sizes.

Decoder. To endow the encoder features with inductive bias, the occupancy network uses a 3D UNet encoder [364] to process the volumetric feature grid. Since U-Net contains convolutional operations, this also introduces translational equivariance in the encoder features, which makes it able to predict the occupancy of different shapes but from the same categories. These aggregated feature maps from the U-Net [231] are then fed into a decoder for predicting occupancy labels. To predict the occupancy value at any arbitrary position, we use tri-linear interpolation to find the features at that point by using the features of all points belonging to the same voxel in the volumetric grid. This point's location and features are passed through a decoder that outputs an occupancy value for each 3D grid location.

Surface Sampling. We render an output mesh of the given input point cloud from the predicted occupancy of the grid points of the convolutional occupancy network using the MISE algorithm [191].

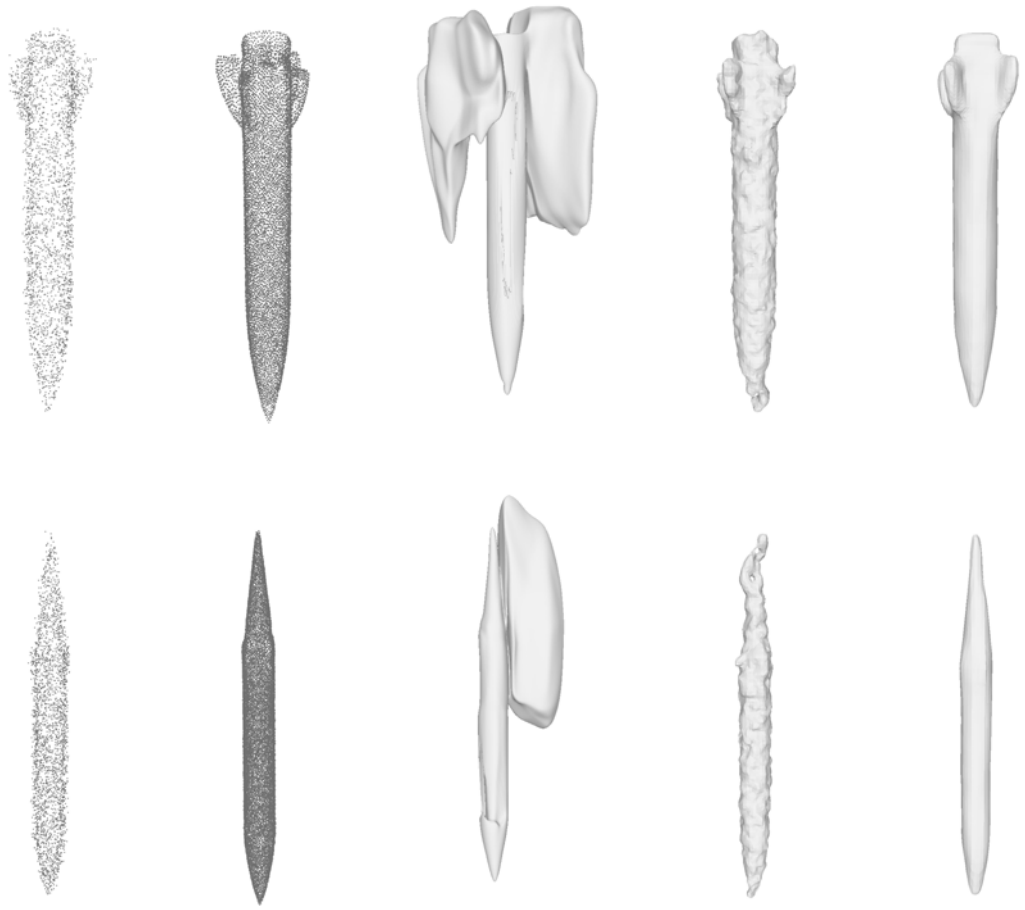


Figure 2.2: **Visual comparison of different reconstruction and upsampling methods.** From left to right: Input, Up-sampling point clouds, Unsupervised reconstruction, Poisson reconstruction, and Supervised reconstruction. As can be seen, the shape quality of supervised reconstruction [212] using neural implicit representation is finer and smoother compared to unsupervised method [81] and Screened Poisson method [128]. In addition, we can obtain a dense and uniformly distributed proximity-to-surface point cloud using Self-supervised Upsampling [298], which contributes to the success of our method. Best viewed with zoom.

We then produce an augmented version $\mathbf{x}_k$ of the original point cloud $\mathbf{x}_0$ by randomly sampling a point cloud from the vertices of the rendered mesh, where k indicates the k-th augmentation.

2.3.3 Augmentation by Point Cloud Upsampling

Inspired by [298], we upsample an input sparse point cloud $\mathbf{x} = \{\mathbf{p}_i\}_{i=1}^{n} \in \mathbb{R}^{n\times 3}$ to a dense point cloud $\mathbf{y} = \{\mathbf{p}_i\}_{i=1}^{N} \in \mathbb{R}^{N\times 3}$ including $N = \lfloor r \times n \rfloor$ points, where r is a desirable scaling factor (set

default to 4). The high-resolution point cloud $\mathbf{y}$ must be dense, uniform, complete, and noise-tolerant. The self-supervised point cloud upsampling strategy includes four steps: seeds sampling, surface projection, outliers removal, and arbitrary-scale point cloud generation.

Seeds Sampling. To obtain uniformly sampled seed points, given a point cloud, we divide the 3D space into equally spaced voxels and estimate the distance from centers to the surface by computing the distance to the triangles formed by the nearest points. Then we choose the centers in a preset range as the seed points.

Surface Projection. Given a seed point c, we obtain the coordinate of the projection point of the seed point c as: $c_p = c + n \times d$, where $\mathbf{n} \in [-1, 1]^3$ and $d \in \mathbb{R}$ are projection direction and projection distance, respectively. The $\mathbf{n}$ and d can be obtained by two multi-layer fully-connected neural networks f_n, and f_d, which borrows from Occupancy Network [191] and DGCNN [294]. The detail of architectures and training procedures can be found in [298].

Outliers Removal. For a projection point c_p, we determine a point as an outlier if $b_p > 1.5\bar{b}$, where b_p is the average bias between c_p and its nearest points and $\bar{b}$ is the average bias of all projection points.

In practice, outlier removal can be regarded as optional, but we empirically found that outlier removal can yield some minor performance improvement of downstream tasks such as classification and part segmentation, and therefore use this step by default in the augmentation.

Point Cloud Generation. We upsample the input point cloud $\mathbf{x}_0$ to a dense point cloud $\mathbf{y}$ using the upsampling network. Then, we sample a fixed number of points from the upsampled point cloud $\mathbf{y}$ by using the farthest-point sampling algorithm to obtain an augmented point cloud $\mathbf{x}_k$ with the desired number of points, where k indicates the k-th augmented point cloud. The examples are shown in Figure 2.2.

2.3.4 Downstream Tasks.

Object Classification. To leverage the augmented point clouds for classification, for both PointNet [217], DGCNN [293], and PointNeXt [220], we extract the global features of each point cloud $\mathbf{x}_k$ including the original point cloud $\mathbf{x}_0$, and then take an average of the features before passing

them to the classifier. Without changing of notation, assume that f is the global feature extractor, and g is the classifier, we can write the label prediction as:

$$g(\text{avgpool}(f(\mathbf{x}_0), f(\mathbf{x}_1), f(\mathbf{x}_2), ...)) \tag{2.3}$$

Algorithm 1: Pseudo-code for our test-time augmentation for the segmentation task.

```
# get_log(p):  return logit at point p.
# get_feat(p): return 3D coords wo/ or w/ logit
# knn(p, X):   return the nearest neighbor of p in X.
# agg(a, b):   combine tensors a and b.

for each point p in X_0:
  logit = get_log(p)
  feat = get_feat(p)
  for i = 1 to N
    neighbors = knn(feat, X_i)
    for each point q in neighbors:
      logit = agg(logit, get_log(q))
  label = argmax(logit)
```

Semantic and Part Segmentation. For semantic segmentation and part segmentation, the aggregation function is more evolved. The basic idea is first to perform segmentation on each point cloud, and then aggregate the results to produce the final segmentation for the original point cloud $\mathbf{x}_0$, but now the aggregation occurs at a per-point level instead of the global features. Let $f_i(\mathbf{x}_k)$ be the features of point i in point cloud $\mathbf{x}_k$, the label prediction of point i in the original point cloud $\mathbf{x}_0$ can be written as:

$$g(\phi(f_i(\mathbf{x}_0), \{f_{\pi_{1,i}}(\mathbf{x}_1)\}, \{f_{\pi_{2,i}}(\mathbf{x}_2)\}, ...)) \tag{2.4}$$

where $\pi_{k,i}$ indicates the corresponding points of point i in $\mathbf{x}_0$ to point cloud $\mathbf{x}_k$, and g as the classifier or any post-processing network. Here we propose a simple algorithm to establish such correspondences via nearest neighbors on the logit vectors, which are detailed in Algorithm 1.

Table 2.1: 3D object classification in ModelNet40 [303] and ScanObjectNN (PB_T50_RS) [275] using self-supervised upsampling point clouds [298].

Method	ModelNet40		ScanObjectNN	
	oAcc	mAcc	oAcc	mAcc
PointNet [217]	89.20	86.20	68.20	63.40
Ours	**92.07**	**88.78**	**76.69**	**72.93**
DGCNN [294]	92.90	90.20	78.10	73.60
Ours	**94.23**	**91.79**	**87.71**	**85.84**
PointNeXt [220]	93.96	91.14	88.18	86.83
Ours	**95.48**	**92.96**	**90.38**	**88.99**
PointMixer [37]	91.41	87.89	82.51	80.03
Ours	**92.71**	**90.42**	**84.18**	**81.25**
PointTransformer [342]	90.64	87.84	82.31	80.77
Ours	**92.55**	**89.73**	**83.66**	**81.37**

2.4 Experimental Results

2.4.1 Implementation Details

We implement our method in Pytorch. We use the convolutional occupancy network [212], and self-supervised point upsampling network [298] for test-time augmentation. For downstream tasks, we experiment with pre-trained models for classification and part segmentation such as PointNet [217], DGCNN [294], PointMixer [37], PointNeXt [220], PointTransformer [342] as well as for large-scale semantic scene segmentation such as RandLANet [101].

Dataset and metric. Our experiments are conducted on different datasets such as ShapeNet [19], ScanObjectNN [275], ModelNet40 [303], SemanticKITTI [8], and S3DIS [4] datasets, including indoor and outdoor environments with both synthetic and real data.

We employ several popular metrics for evaluation, such as the overall and mean percentage accuracy are computed for the classification task, the Instance and Category Intersection-Over-Union (mInsIoU, mCatIoU) are utilized for the part segmentation task, and the mean Accuracy (mACC)

and mean IoU (mIoU) are used for semantic segmentation task.

Data processing. For ShapeNet [19] dataset, we use the pre-processed data produced by PointNet [217], which is an early version of ShapeNet (version 0) to train the segmentation network. Nonetheless, the ShapeNet data used to train the convolutional occupancy network [212] is a different version (version 1). Since the number of objects differs in these variants of ShapeNet, we only use the objects that appear in both datasets. For ScanObjectNN [275], and ModelNet40 [303] datasets, we follow the instruction in the official implementation of PointNeXt [220]. For SemanticKITTI [8] dataset, we follow the instruction in the official implementation of RandLANet [101]. We also follow Self-UP [298] to prepare the data for point cloud upsampling.

2.4.2 Classification Results

The object classification results are shown in Table 2.1 and are conducted on two challenging datasets (ScanObjectNN [275], and ModelNet40 [303]). ScanObjectNN presents considerable problems to the various point cloud analysis algorithms already in use due to occlusions and noise. Based on PointNeXt [220], we conduct experiments on PB_T50_RS, the most challenging and widely deployed version of ScanObjectNN. Note that the reported performance of PointNet and DGCNN in our paper is higher than the original PointNet and DGCNN paper because we adopt the re-implementation of PointNet and DGCNN from the PointNeXt paper, which includes optimized training strategies.

As can be seen, the optimized baseline model by PointNet [217] performed very well on both ModelNet40 and ScanObjectNN classification. Despite such, applying augmentation with our method leads to a performance boost of $1-2\%$, which is a significant gain given the saturating accuracy of this dataset. We also empirically found that augmenting with more than one sampled point cloud does not significantly improve this task.

Note that as convolutional occupancy network [212] requires ground truth signed distance functions to train surface reconstruction, we only perform the classification task using self-supervised point upsampling [298].

Table 2.2: Part segmentation on ShapeNet [19] using self-supervised upsampling point clouds as input.

2048 points	mInsIoU	mCatIoU
PointNet [217]	80.74	83.73
Ours	**82.88**	**86.25**
DGCNN [294]	81.08	84.18
Ours	**83.38**	**86.70**
PointNeXt [220]	84.23	86.73
Ours	**85.07**	**87.60**

Table 2.3: Part segmentation on ShapeNet [19] using surface sampling with different numbers of points. We can obtain competitive segmentation results by aggregating predictions from augmented point clouds to original point clouds.

Method	**128 points**		**256 points**	
	mInsIoU	mCatIoU	mInsIoU	mCatIoU
PointNet [217]	79.06	81.72	83.12	85.12
Ours	**79.55**	**82.66**	**83.25**	**85.82**
DGCNN [294]	59.75	66.34	69.88	74.57
Ours	**71.63**	**81.95**	**79.98**	**85.65**

2.4.3 Segmentation Results

Part Segmentation. The part segmentation results are shown in Table 2.2 and Table 2.3. It can be seen that by applying our method, the mInsIoU, and mCatIoU are improved compared to the baseline approach. The results also demonstrate the robustness of our method as it works well with different network backbones, e.g., PointNet [217] that involves only per-point and global point cloud features, DGCNN [293] which establishes and learns dynamic graphs in point neighborhoods, and the SOTA PointNeXt [220]. It is worth noting that the improvement is mainly gained from the refinement of the segmentation boundaries (Figure 2.6 and 2.7 (d)).

Semantic Segmentation. To assess the generalizability of our strategy, we also tested on real-world data from S3DIS [4] and SemanticKITTI [8] datasets.

As SemanticKITTI data is captured by LiDAR sensors, it is favorable to use point upsampling as the augmentation technique. It can be seen in Table 2.4, by applying our method, the mAcc, and mIoU are improved compared to the baseline approach. We provide qualitative results in Figure 2.3 for a better comparison.

We provide quantitative and qualitative results on the S3DIS dataset in Table 2.5 and Figure 2.4. As can be seen, our TTA is effective and improves upon the baseline PointNeXt.

2.4.4 Additional Analysis

We perform additional experiments to validate the performance of our test-time augmentation. We select the segmentation task for these experiments as it produces dense prediction, which can be

Table 2.4: Semantic segmentation on SemanticKITTI [8] using self-supervised upsampling point clouds.

Method	mAcc	mIoU
RandLANet [101]	97.23	68.84
Ours	**99.17**	**70.55**

Table 2.5: Semantic segmentation on S3DIS [4] dataset using self-supervised upsampling point clouds.

Method	mAcc	mIoU
PointNeXt [220]	70.69	64.26
Ours	**71.75**	**65.23**

seen as generalized classifications.

Point density. In Figure 2.5, we plot the segmentation accuracies (mIoU) across different numbers of input points. Specifically, we randomly sample 128, 256, 512, 1024, and 2048 points as input to perform the segmentation. Compared to PointNet, it can be seen that our augmentation offers significant performance improvement on sparse point clouds (128 and 256 points) and performs similarly to PointNet when the input points get denser.

We also found that by varying the number of input points (Figure 2.5), DGCNN cannot perform well on sparse point clouds with a very large performance gap between the sparse and dense point clouds (more than 20% between 128 and 2048 points). This is because for sparse point clouds, the neighbor graphs by DGCNN degenerate [294]. Despite such, our test-time augmentation can still improve the performance and significantly reduce the performance gap to around 6%. This shows that our test-time augmentation is robust to the number of input points.

Ablation study. We conduct an ablation study on the part segmentation task on ShapeNet and provide the results in Table 2.6. We select the segmentation task as it is a generalized form of classification at the per-point level, and also aim to justify the more complex design choices in the aggregation function for this task. We use inputs with 2048 points. Our baseline is an implementation that k-nearest neighbors are performed with just 3D coordinates as features. By adding logits as features, we can have 2% gain in mIoU (model A). We also test different aggregate functions like max pooling and average pooling and find that average pooling performs better (model B vs. C). Additionally, we repeat the sampling to obtain multiple augmented point clouds. By fusing the segmentation of these augmented point clouds to the original point cloud, further improvement can be achieved (model A&C and B&C). This shows that it is critical to compute accurate correspondences between the augmented point cloud and the original point cloud to achieve higher accuracies. From

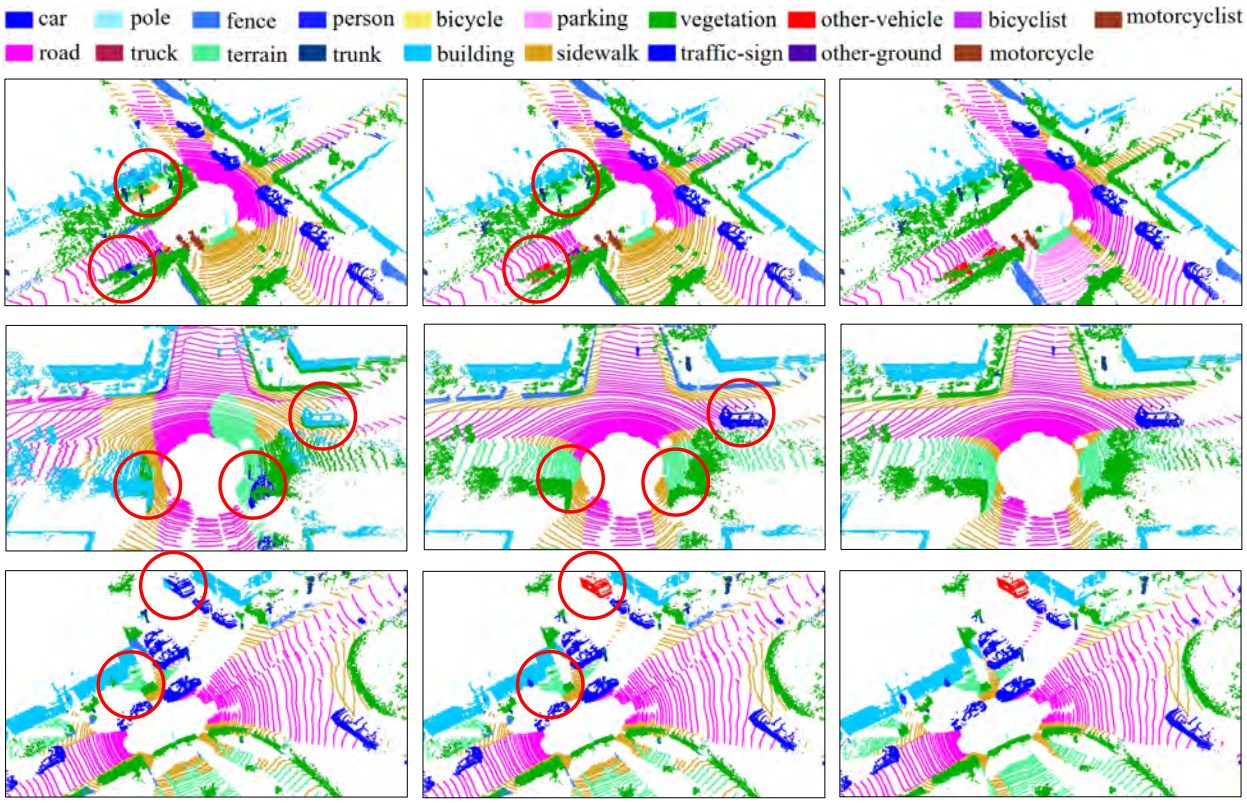
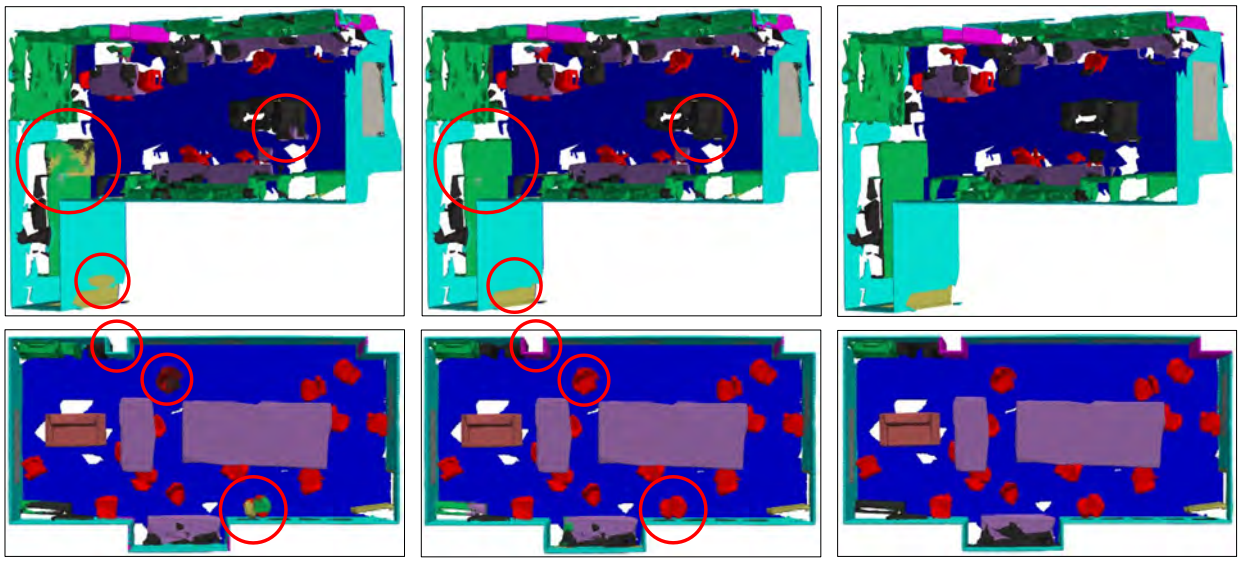


Figure 2.3: Semantic segmentation results of RandLANet [101], ours and ground truth on SemanticKITTI dataset [8].

the above analysis, we can see that multi-sampling and changing aggregate functions can yield further improvement.

Comparison among augmentation techniques. We provide a comparison to study which augmentation technique should be used in practice. We compare the popular Screened Poisson reconstruction [128] to convolutional neural network [212] and point cloud upsampling [298]. The results in Table 2.7 show that our augmentation techniques are more favorable in performance than Screened Poisson reconstruction. The performance between the convolutional occupancy network and self-supervised upsampling is rather similar, with the convolutional occupancy network is slightly better in the instance IoU metric. We hypothesize that when (ground truth) surface information is available, it could be used to supervise the augmentation, else point cloud upsampling could be an effective and robust augmentation in several scenarios. We also explore a recent unsupervised reconstruction [81] but find that the shape quality is poor compared to Screened Poisson



PointNeXt Ours Ground Truth

Figure 2.4: Semantic segmentation results of PointNeXt [220], ours and ground truth on S3DIS dataset [4]. Our TTA can group and segment well objects.

Table 2.6: Ablation studies of our test-time augmentation on ShapeNet [19] using surface sampling. Performing k-nearest neighbor search on high-dimensional feature space (model A, B) and using the average function (model B) for aggregating predictions result in improved accuracies. The performance can be further boosted by using extra augmented point clouds (model A&C and B&C). The reported metric is mCatIoU.

2048 points	PointNet [217]	DGCNN [294]
xyz (max)	86.45	84.26
A: with logit (max)	88.30	85.90
B: with logit (avg)	88.43	86.05
C: with 10x samples	86.39	84.13
A & C (max)	88.26	85.96
B & C (avg)	**88.58**	**86.16**

reconstruction, and thus unsuitable for augmentation. Exploring more robust reconstruction could lead to interesting augmentation techniques for future work.

Augmentation without normals. We experiment with implicit surface representation for data augmentation in a practical setting where normals are not available. In this case, existing methods have to rely on the pure 3D coordinates (xyz), causing a performance decrease, while our method

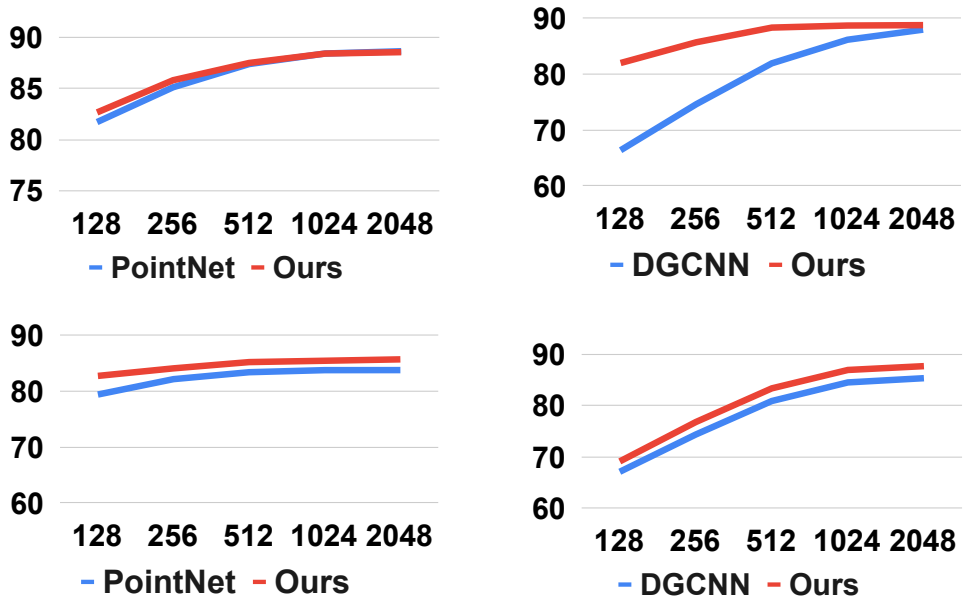


Figure 2.5: TTA using surface sampling (top row) and self-supervised upsampling (bottom row) on part segmentation on ShapeNet with different numbers of points. We found that applying TTA for sparse point clouds of the surface sampling method yields significant improvement. In contrast, the improvement of TTA on upsampling point clouds is more stable, thanks to dense and uniformly distributed proximity-to-surface point clouds. The horizontal axis is in log scale.

Table 2.7: Comparison of different augmentation methods on part segmentation with PointNet [217] as backbone on ShapeNet [19] dataset.

128 points	mCatIoU	mInsIoU
Poisson [128]	81.79	77.86
Implicit [212]	82.66	79.55
Self-UP [298]	82.70	78.99

Table 2.8: Augmentation with normals using surface sampling on ShapeNet [19]. The backbone is PointNet [217].

2048 points	mAcc
Org. xyz	97.73
Aug. xyz	98.53
Aug. xyz & normals	98.38

can easily solve this problem by sampling the 3D points as well as the normal vectors directly from the implicit surface. In this way, our method can maintain the performance to the same level regardless of the normal existence. This is verified in Table 2.8 that our augmentation outperforms the baseline when only the 3D coordinates (xyz) are available.

Comparison to traditional augmentation. We also conduct comparison experiments with traditional augmentation. Adding Gaussian noise is a commonly used traditional data augmentation scheme that perturbs the points by sampling from a Gaussian distribution. Combining results from

this perturbation in test-time augmentation is known as voting [134, 171]. In our implementation, we sample offsets from a zero mean Gaussian with different standard deviation σ and add the offsets back to the original point clouds to form augmented point clouds. For our TTA, we sample one more point cloud and then average the global features of the additional point cloud and the original point cloud before passing them to the classifier. As can be seen in Table 2.9, our method outperforms the traditional augmentation scheme. This shows the advantage of our method that sampling points from the surface reconstructed by the implicit representation is unnecessary to tune the standard deviation σ, which is sensitive to point density.

In addition, the results from using flip and scale are shown in Table 2.10 on ModelNet40, which confirms the better performance of our method. The crop is not effective in our case because cropped point clouds are very different from the original point clouds.

Table 2.9: Comparison with traditional augmentation Gaussian Noise using surface sampling on ShapeNet [19]. The backbone is PointNet [217].

2048 points	mAcc
$\sigma = 0.05$	98.21
$\sigma = 0.07$	97.69
$\sigma = 0.1$	96.54
Ours	**98.53**

Table 2.10: Comparison with traditional augmentations, *e.g.* scale (S) and flip (F) on ModelNet40 [303].

	PointNet	S(0.6)	S(0.8)	S(2.0)	Ours
mAcc	83.88	82.35	82.58	83.14	**85.34**
oAcc	86.91	86.56	86.88	86.92	**88.65**
	PointNet	F(xyz)	F(yz)	F(xy)	Ours
mAcc	83.88	81.51	81.69	81.63	**85.34**
oAcc	86.91	86.16	86.44	86.53	**88.65**

Computation overhead. While having better performance, modern TTA, including our method, relies on a neural network to predict augmented samples from each input and thus has more overhead compared to traditional methods. For example, if we use M augmented point clouds, the overhead is approximately M times the original time cost. To circumvent this problem, we propose to exploit parallelism and use batched inference instead. First, our computation overhead is sub-linear, which means that even if we have 10 times as many augmented samples (the same number of augmented samples that we used for all of our experiments), the overhead will only increase by a factor of two (see Table 2.11). This overhead is manageable and can be reduced even further through the utilization of batched prediction, as demonstrated in Table 2.11 below. Additionally, additional engineering like deploying the network to an inference-only framework (TensorFlow Lite) would further optimize

inference. Second, we can reduce the number of augmented samples ($M \in \{2, 4, 8, 10\}$), which would result in milder improvement in comparison to the baseline but would incur significantly less overhead (see Table 2.11). Finally, the overhead of TTA can be offset by its ease of use compared to other methods for performance improvement, e.g., when only pre-trained models are given or when retraining the entire model is not possible.

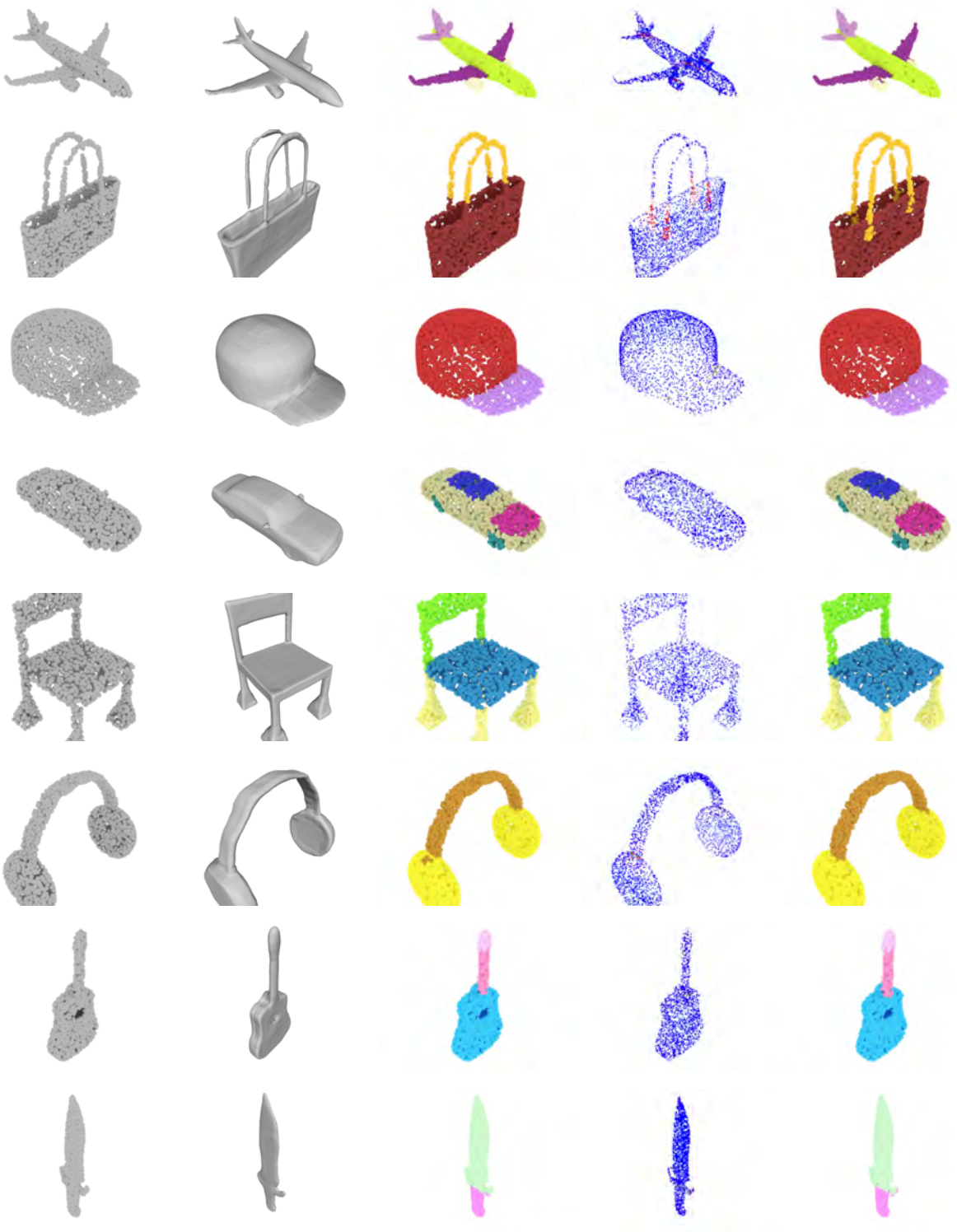
Table 2.11: Running time breakdown of different stages of our TTA with different augmentation samples M for the classification task on ModelNet40.

	M=10	M=8	M=4	M=2	PointNeXt
Batched forward 10x	0.6573	0.1025	0.1021	0.1016	0.2147
Batched GPU FPS 10x	0.1036	0.5258	0.2892	0.1446	0.0
Aggregation	0.0968	0.0959	0.0948	0.0934	0.0
Others	0.4382	0.4334	0.4295	0.4252	0.4382
Total time	1.2959	1.1576	0.9156	0.7648	0.6529
mAcc	92.96	92.51	91.87	91.38	91.14

2.5 Discussion and Conclusions

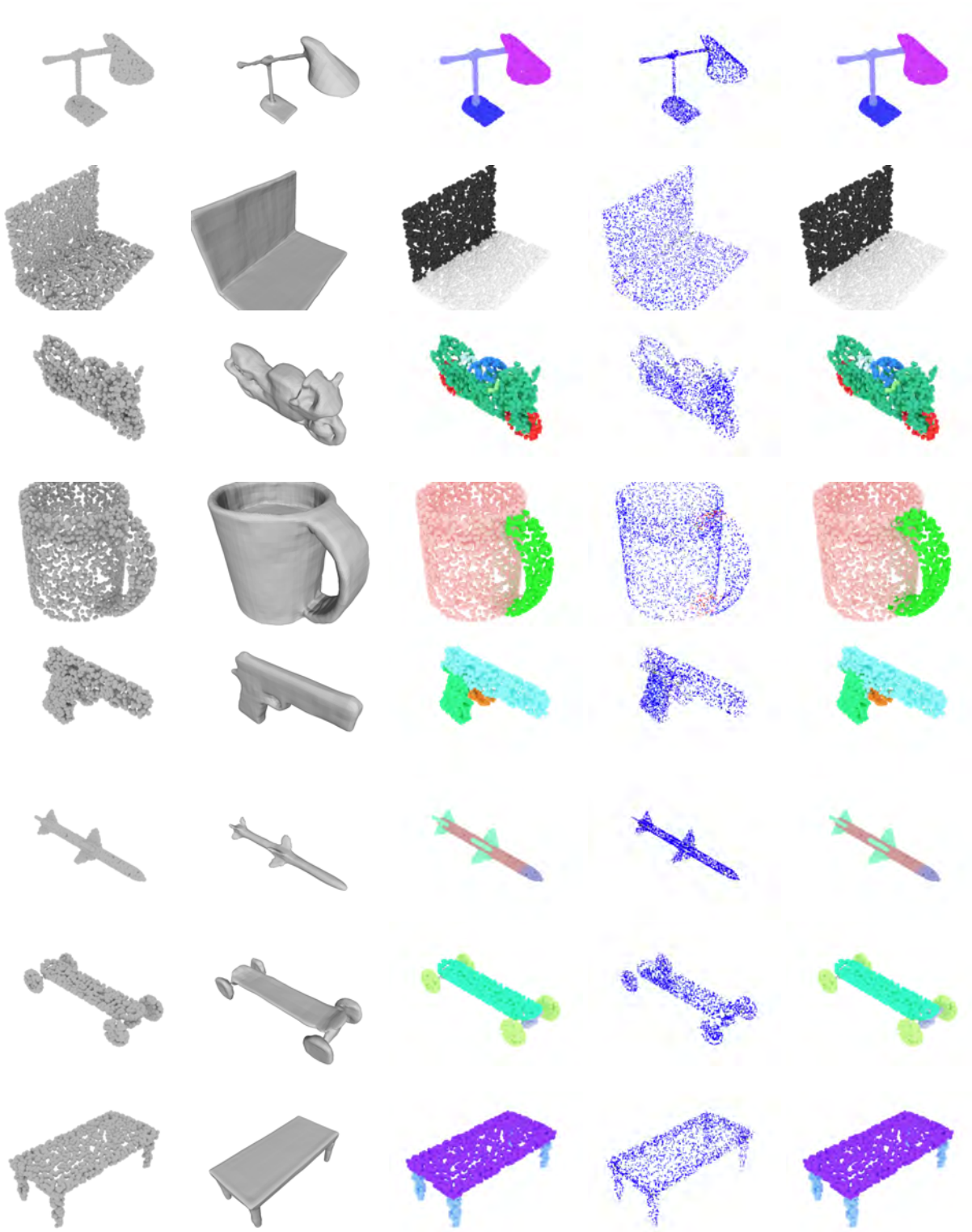
We presented a new method for augmenting point clouds at test time by leveraging a neural implicit network and a point upsampling network to sample augmented point clouds and showed that such augmentation works effectively for the classification and semantic segmentation task. Our results are encouraging since this is one of the first attempts to design a test-time augmentation technique for 3D point cloud deep learning.

A main difference between our TTA and traditional methods is that traditional methods only use simple transformations and are thus lightweight, but not input-aware and less robust. While our TTA requires more resources, the extra computation remains affordable and our method shows good results across tasks and datasets. We believe further explorations to reduce such performance trade-offs would be valuable contributions to this less-explored area of test-time augmentation for 3D point clouds.



(a) Input (b) Reconstruction (c) Our result (d) Difference map (e) Ground truth

Figure 2.6: Visualization of part segmentation results. As can be seen in the difference map, where blue and red points indicate correct and wrong labels, respectively, our test-time augmentation mainly deals with the labels along the boundaries, improving their accuracies through aggregating predictions from augmented point clouds.



(a) Input (b) Reconstruction (c) Our result (d) Difference map (e) Ground truth

Figure 2.7: Visualization of part segmentation results. As can be seen in the difference map, where blue and red points indicate correct and wrong labels, respectively, our test-time augmentation mainly deals with the labels along the boundaries, improving their accuracies through aggregating predictions from augmented point clouds.

CHAPTER 3

4D DYNAMIC POINT CLOUDS FLOW ESTIMATION AND SHAPE RECONSTRUCTION

3.1 Introduction

Literature has shown several breakthroughs in deep learning for reconstruction of 3D models from point clouds. Recently, the research community has seen great successes in neural representations using implicit fields [192, 208, 28, 194], which pave an effective way on how 3D data can be represented by neural networks. Unlike traditional representations that are often realized in discrete forms (e.g., discrete grids of pixels in image representation, discrete grids of voxels in 3D object representation), the neural implicit representation parameterizes a signal as a continuous function via a neural network. This function maps a signal from its original domain, which can be queried at any resolution, to an output domain that captures some properties of the query. Most existing methods focus on the neural representation of 3D data in static conditions. However, in reality, real-world objects exist in dynamic environments that change over time and space, and thus cannot be well modeled using implicit representations applied to static shapes. Approaches for 4D reconstruction (i.e., reconstruction of a 3D object over time) have been explored but they often need expensive multi-view settings [148, 41, 198, 5]. These settings rely on a template model (of the target object) with fixed topology [2, 124, 274, 347], or require smooth spatio-temporal input [211, 279], and thus limiting their applicability in practice.

To enable object reconstruction directly from 4D data without predefined templates, OFlow [202], a pioneering method for 4D reconstruction, was developed to calculate motion fields of 3D points in a 3D point cloud in space and time to implicitly represent trajectories of all the points in dense correspondences between occupancy fields. To learn the motion fields in both space and time domains, OFlow made use of a spatial encoder to learn the spatial structure of the input point cloud and a temporal encoder to learn the temporal changes of the point cloud in time. Despite impressive reconstruction results, this paradigm has a number of drawbacks. First, its spatial encoder does not take geometric attributes from numerous frames into consideration, impairing the capacity

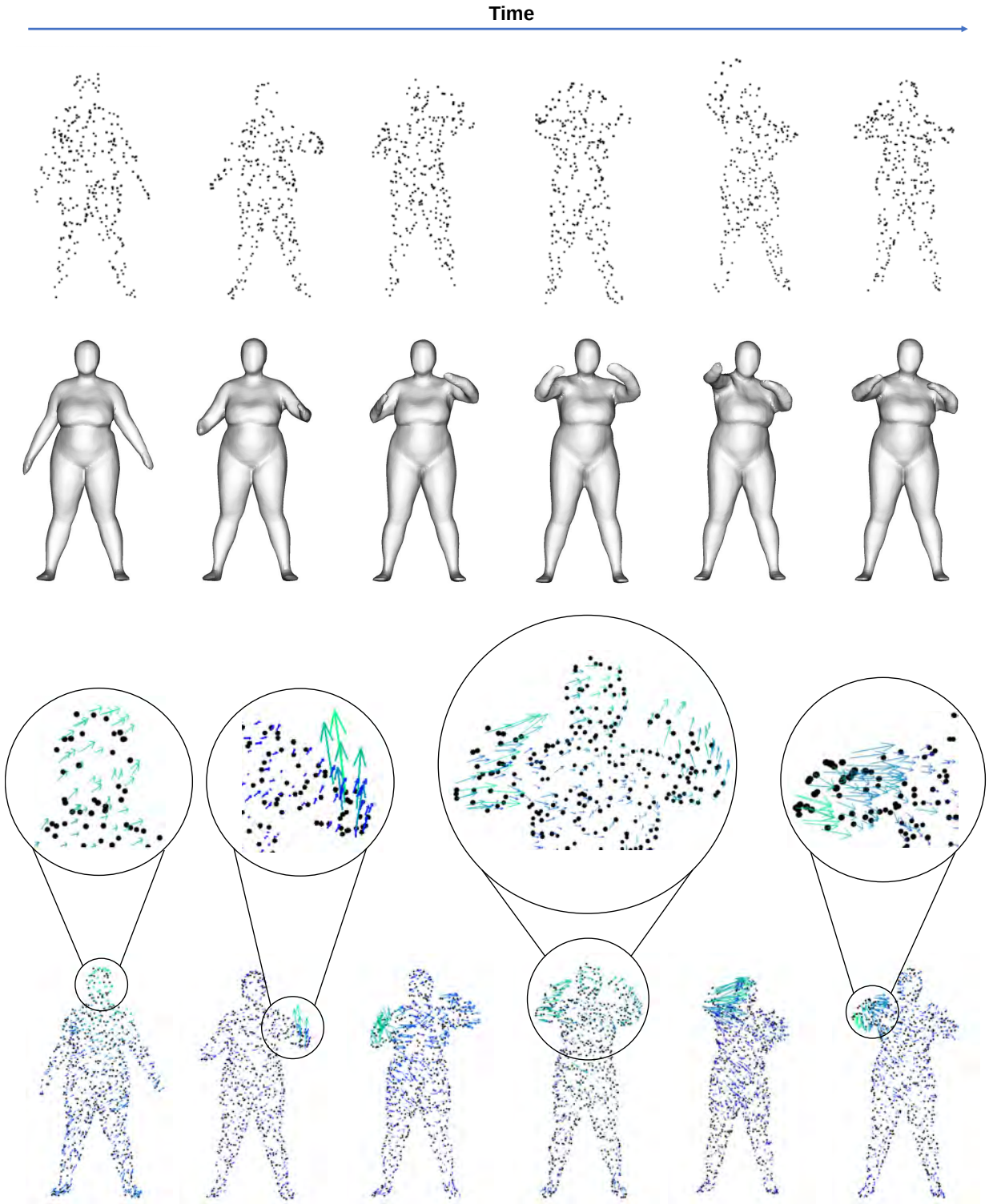


Figure 3.1: **Summary of our method.** Given a sequence of time-varying 3D point clouds (first row), we jointly reconstruct corresponding 3D geometric shapes (second row) and estimate the motion field for every point, including the original points and reconstructed points (third row).

to precisely reconstruct geometric surfaces. Neither does its temporal encoder take into account temporal correspondences, which are critical for accurately capturing temporal dynamics. Second, errors in the prediction of temporal continuity and reconstructed geometries are accumulated by an integral of estimated instantaneous findings. Third, OFlow is trained using supervised learning. This requires correspondence labeling for all 3D points across frames in training data, leading to high labeling costs and low scalability. Fourth, the method exhibits low computational efficiency in both the training and inference phases. This is due to the expensive computations required to sequentially determine trajectories of 3D points throughout time by solving complex ordinary differential equations.

To address the aforementioned challenges, we propose a network architecture, namely RFNet-4D++, for 4D reconstruction and flow estimation of dynamic point clouds. Our key idea is to jointly perform two tasks: 4D reconstruction and flow estimation with an intention that each task can leverage the other one to improve the overall performance. Specifically, our network takes as input a sequence of 3D point clouds of an object over time. The point clouds are encoded using a dual cross-attention-based compositional encoder, resulting in spatio-temporal representations. The spatio-temporal representation of a point cloud at a time step is calculated from the spatial layout of points in that point cloud and the temporal changes of the points in the point cloud throughout time. Spatio-temporal representations are then decoded by a joint decoder which jointly reconstructs the object and predicts a motion vector for each point in the reconstructed object throughout time. The entire network can be trained end-to-end, where the reconstruction and flow estimation tasks are trained with supervised and unsupervised learning, respectively.

Our method offers numerous advantages. First, our method relaxes the requirement of point-to-point correspondence labelling by making flow learning unsupervised. Second, our method allows fast inference by enabling parallel computations of spatial and temporal features. This ability makes our method much advantageous in comparison with OFlow which estimates the motion flows sequentially and thus often experiences time lags. We illustrate several results of our method in Figure 3.1. In summary, the contributions of our work are as follows:

- RFNet-4D: a network architecture for joint object reconstruction and flow estimation from a sequence of time-varying 3D point clouds.
- A joint learning method for training the proposed RFNet-4D using both supervised and unsupervised learning, and in both forward and backward time direction. To the best of our

knowledge, this learning mechanism is novel, and its benefit is verified throughout extensive experiments and datasets.

- We propose a new spatio-temporal representation that utilizes a dual cross-attention mechanism to improve the overall accuracy of our method (namely RFNet-4D++).
- Extensive experiments and analyses on D-FAUST (human) and DeformingThings4D (humanoids, and various animal species) that prove the effectiveness and efficiency of our proposed method on two tasks: 4D reconstruction and flow estimation.

3.2 Related Work

3.2.1 3D Reconstruction

Numerous studies have been conducted with the goal of reconstructing a continuous surface from a variety of inputs, including RGB images [262, 297, 126], point clouds [128]. Thanks to advances in deep learning, recent 3D object reconstruction approaches have resulted in significant progress. Early attempts represent reconstructed objects in a regular grid of 3D voxels [285, 78] or point clouds [216, 70]. However, those representations cannot well capture surface details and suffer from low resolutions. Alternatively, there are methods, e.g., [284, 159, 123] reconstructing triangular meshes (including vertices and faces) of 3D objects. In these methods, an initial template with fixed topology is employed and the reconstruction is performed using regression. For surface representation, several methods focus on learning an implicit field function that allows more variable topology in reconstructed objects [36, 18, 59, 117].

To extend the ability of implicit functions on representations other than traditional forms (i.e., voxels, points, meshes), occupancy maps [192, 213] and distance fields [208, 18] are proposed. An occupancy map of a 3D point cloud contains indicators that indicate being foreground of points in the 3D space. A distance field provides the distance from every point to its nearest surface. Since the implicit function models objects in a continuous manner, more information is preserved and more complicated shapes can be well described. For instance, Occupancy Network in [192] described a 3D object using continuous indicator functions that indicate which sub-sets of 3D space the object occupies, and an iso-surface retrieved by employing the Marching Cube algorithm [176].

3.2.2 4D Reconstruction

Despite being less studied compared with 3D reconstruction, literature has also shown recent attention of the research community to 4D reconstruction, i.e., reconstruction of a sequence of 3D objects from time-varying point clouds [148, 198, 5]. In this section, we limit our review to only learning-based 4D reconstruction methods.

A crucial component in 4D reconstruction is motion capture and modelling. Niemeyer *et al*. [202] introduced a learning-based framework that calculates the integral of a motion field specified in space and time to implicitly represent the trajectory of a 3D point cloud to generate dense correspondences between occupancy fields. Jiang *et al*. [116] proposed a deformable representation for 4D capture that encloses a 3D shape and the velocity of its 3D points over time. Specifically, to simulate the motion of time-varying 3D data, a neural Ordinary Differential Equation was trained to update the starting state of the motion based on a learnt motion representation. A 3D model at each time step was then reconstructed using a shape representation and a respective motion state. Tang *et al*. [261] proposed a pipeline for determining the temporal evolution of the 3D shape of the human body using spatially continuous transformation functions between cross-frame occupancy fields. By explicitly learning of the continuous displacement motion fields, the pipeline aims to construct dense correspondences between projected occupancy fields at different time steps. Li *et al*. [156] presented 4DComplete, a model to combine geometry completion with non-rigid motion tracking. They argued that the shape and motion of non-rigid objects are intricate data modalities, i.e., the geometry provides valuable information for motion estimation and the motion is seen as the evolution the geometry in time.

3.2.3 Motion Transfer

Traditional techniques for 3D pose transfer problems often use discrete deformation transfers. An example of mesh deformation is described in [282], where spatially adaptable instance normalisation [107] was used to modify 3D meshes. However, this method requires a dense triangular mesh of an object to be given in advance, while there is a specific mechanism to depict continuous flows in both spatial and temporal domains.

3D motion transfer is another technique for creating 3D objects from a pair of source and target object sequences. It operates by causing the target object sequence to undergo the same temporal

deformation in the source object sequence. This technique can be applied to model the continuous transformation of an object's pose over time. For instance, OFlow [202] transmitted motion across sequences of source and target human models by applying motion field-based representations to the targets in a predetermined manner. However, since OFlow does not explicitly differentiate the representations of pose and shape, we found that it only produces reasonable motion transfer results when the identities and initial poses of both the source and target objects are similar.

3.2.4 Shape Correspondence Modelling

Modeling of point-to-point correspondences between two 3D shapes is a well-studied topic in computer vision and computer graphics [10]. The goal of modeling time-varying occupancy fields is strongly related to the goal of field-based deformation [183], which we have previously discussed. However, most of these works describe the motion fields only on object surfaces. To better describe the motion flow, we argue to model the correspondences between 3D shapes in the entire 3D space.

When modelling the growth of a signed distance field, Miroslava *et al*. [247] chose to implicitly provide the correspondences between points in two 3D shapes rather than yielding them explicitly. The point-to-point correspondence estimation was formulated as the optimisation of an energy function capturing the similarity between the Laplacian eigen function representations of the input and the target shapes. However, we found that this method is sensitive to noise, probably due to lack of the capability of providing correspondences accurately from signed distance fields. In contrast, we learn the rich correspondences between time-varying occupancy fields based on a intuitive insight, that the occupancy values of points are always invariant under temporal evolution of the occupation fields (please check the method and experiment sections for more details).

Recently, there are methods modelling deformable shapes overtime using implicit functions (continuous signed distance fields), e.g., DIT [350], NDF [255], ImplicitAtlas [319]. DIT [350] employed LSTM [99] to model smooth deformations, while NDF [255] utilised NODE [25] to achieve diffeomorphic deformations. On the other hand, ImplicitAtlas [319] leveraged the integration of multiple templates to enhance the capability of shape modelling. Remarkably, this improvement in shape modelling comes at a minimal computational cost, making ImplicitAtlas [319] an efficient choice for practical applications. By combining multiple templates, ImplicitAtlas [319] is shown to effectively capture and represent complex shape variations while maintaining computational efficiency.

3.3 Our Method

3.3.1 Overview

Our network takes as input a sequence of sparse, incomplete, and noisy 3D point clouds $\{P_t|t = 1, ..., T\}$ where T is the length of the sequence, and each point cloud P_t is a set of 3D locations. Our aim is to simultaneously perform the following tasks:

- Reconstruct a sequence of occupancy maps $\{O_t|t = 1, ..., T\}$ where each O_t is an occupancy map of a point cloud P_t, i.e., $O_t(\mathbf{p}) = 1$ if $\mathbf{p}$ is a 3D point on the reconstructed surface of P_t, and $O_t(\mathbf{p}) = 0$, otherwise;
- Estimate a sequence of vector fields $\{V_t|t = 1, ..., T\}$ where each V_t is a 3D vector field capturing motion flows of reconstructed points of P_t, i.e., $V_t(\mathbf{p}) \in \mathbb{R}^3$ represents the motion flow of a reconstructed point $\mathbf{p}$ at time step t given a point cloud P_t.

Both tasks benefit from a compositional encoder that learns spatio-temporal representations from time-varying point clouds. The temporal features contained in these spatio-temporal representations capture holistic motion information and are computed once on the entire input point cloud sequence. This allows fast computations in following operations as spatio-temporal data can be processed at any arbitrary frame. The spatio-temporal representations are processed by a joint decoder including two decoders, each of which extracts relevant information for its downstream task (i.e., reconstruction and flow estimation). These decoders do not operate independently but cooperate closely to fulfill their tasks. To further exploit the benefit of temporal information, we couple the reconstruction and flow estimation tasks in both forward and backward time directions. We present an overview of our method inFigure 3.2. We describe the main components of our method in the following sections.

3.3.2 Compositional Encoder

The compositional encoder includes a temporal encoder and a spatial encoder. There exist several manners to encode 4D point clouds. For instance, Liu *et al.* [170] applied spatio-temporal neighborhood queries in representing 4D point clouds. However, this method requires high computational complexity. Inspired by the success and efficiency of the point cloud representation used in OFlow [202] and LPDC [261], we adopt the parallel ResNet [93] variant of PointNet [218] for both the spatial and temporal encoder (see Figure 3.3). These encoders are basically similar in their architectures. The difference between them is that while the spatial encoder processes each point

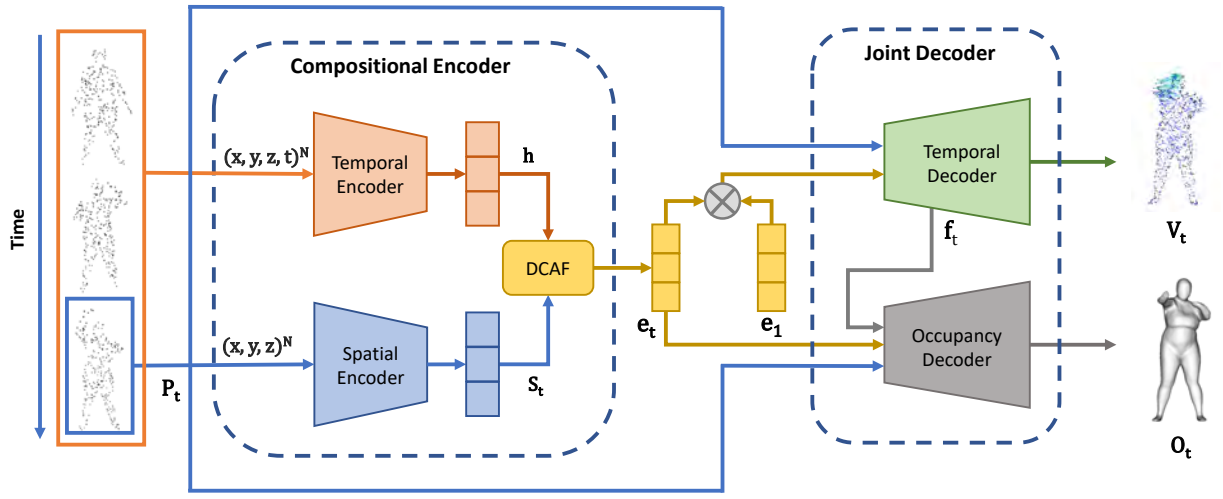


Figure 3.2: **Overview of our method RFNet-4D++.** An input 3D point cloud sequence is fed into a spatio-temporal encoder to extract spatio-temporal representations. The representations are then passed via two distinct decoders, occupancy and motion decoders. In each data frame, the occupancy decoder aims to predict an occupancy field of the point cloud in the frame. Simultaneously, the motion decoder predicts the correspondences between points in the current frame and its preceding frame. $\otimes$ indicates a concatenation operation.

cloud P_t individually at a time t to generate a representation $\mathbf{s}_t$, the temporal encoder acquires the whole point cloud sequence to calculate a holistic temporal representation $\mathbf{h}$ once.

The spatial and temporal representations are finally fused to form a spatio-temporal representation $\mathbf{e}_t$ that encodes the geometric information of a point cloud P_t in space with regard to its temporal changes (see Figure 3.2). Our encoders share similar structures with the encoders in LPDC [261]. In our original work (RFNet-4D) [268], we created the spatio-temporal representation by simply concatenating the spatial and temporal features. However, such simple concatenation does not effectively capture the topology the point clouds' structures over time (e.g., different parts of a human body in motion can have different velocities while still obeying topological rules of deformation). To overcome this issue, we introduce here a dual cross-attention fusion module (DCAF) that effectively learns the correlation between spatial and temporal features. Our DCAF is inspired by the works in [112, 49], which learn attention scores across different modalities.

We describe the architecture of the DCAF module in Figure 3.4. The DCAF module includes two sub-modules: a Spatial-guided Cross Attention (SCA) module and a Temporal-guided Cross Attention (TCA) module. In the SCA module, we utilise the spatial representation $\mathbf{S}_t$ for a query and the temporal representation $\mathbf{h}$ for a key and a value. This configuration enables us to effectively

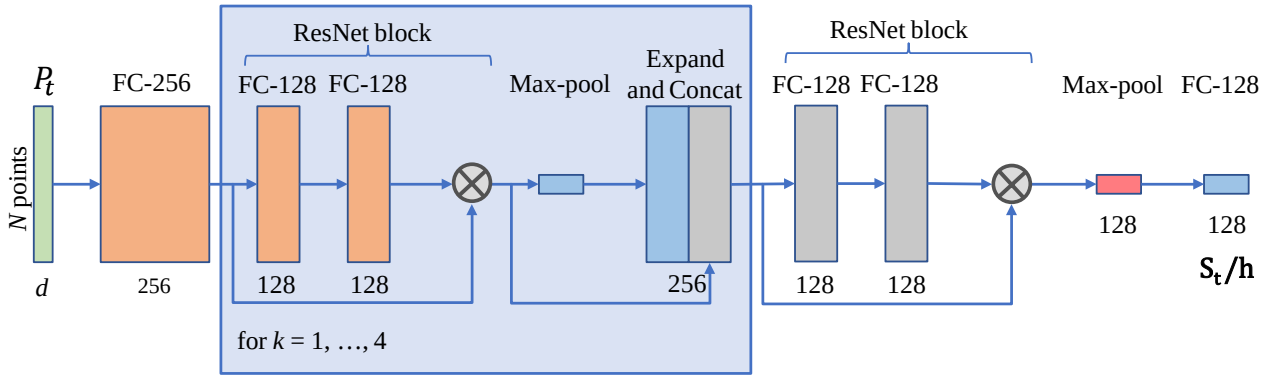


Figure 3.3: **Spatial/temporal encoder.** The input dimension d is set accordingly to a corresponding encoder. In particular, $d = 3$ (i.e., (x, y, z)-coordinates) for the spatial encoder and $d = 4$ (i.e., (x, y, z)-coordinates and time variable) for the temporal encoder. $\otimes$ indicates a concatenation operation. Outputs of the spatial and temporal encoder are $\mathbf{S}_t$ and $\mathbf{h}$, respectively.

capture inter-point spatial dependencies in a point cloud, allowing for extraction of a global spatial context $\mathbf{e}_t^{SCA}$. The TCA module takes two inputs: a concatenated representation (from $\mathbf{S}_t$ and $\mathbf{h}$) which serves as a query, and the output of the SCA module $\mathbf{e}_t^{SCA}$ which forms a key and a value. Unlike the SCA module, the TCA module aims to extract a global temporal context $\mathbf{e}_t^{TCA}$ by integrating all spatial positions across different time steps. Each the SCA/TCA module consists of several layer normalisation and projection transformations implemented by fully connected layers. The spatio-temporal representation $\mathbf{e}_t$ is finally formed by element-wise addition of $\mathbf{e}_t^{SCA}$ and $\mathbf{e}_t^{TCA}$.

Following the conventional definition of cross attention [112, 49], the output of a cross attention module (SCA/TCA) can be expressed as:

$$\text{Cross-Attention}(Q, K, V) = \text{Softmax}\left(\frac{QK^\top}{\sqrt{C}}\right)V \tag{3.1}$$

where Q, K, V represent the projected queries, keys and values, respectively, $\frac{1}{\sqrt{C}}$ is the scaling factor where C is the number of total channels.

Depending on the particular cross attention module (SCA/TCA), Q, K, V are defined accordingly. Specifically, as shown in Figure 3.4, for the SCA, Q is calculated from $\mathbf{S}_t$, while K and V are derived from $\mathbf{h}$. On the other hand, for the TCA, Q is obtained from the concatenation of $\mathbf{S}_t$ and $\mathbf{h}$, while K and V are determined from the output of the SCA module.

Since $\mathbf{h}$ is computed once on the entire input point cloud sequence, $\mathbf{e}_t$ can be extracted at any

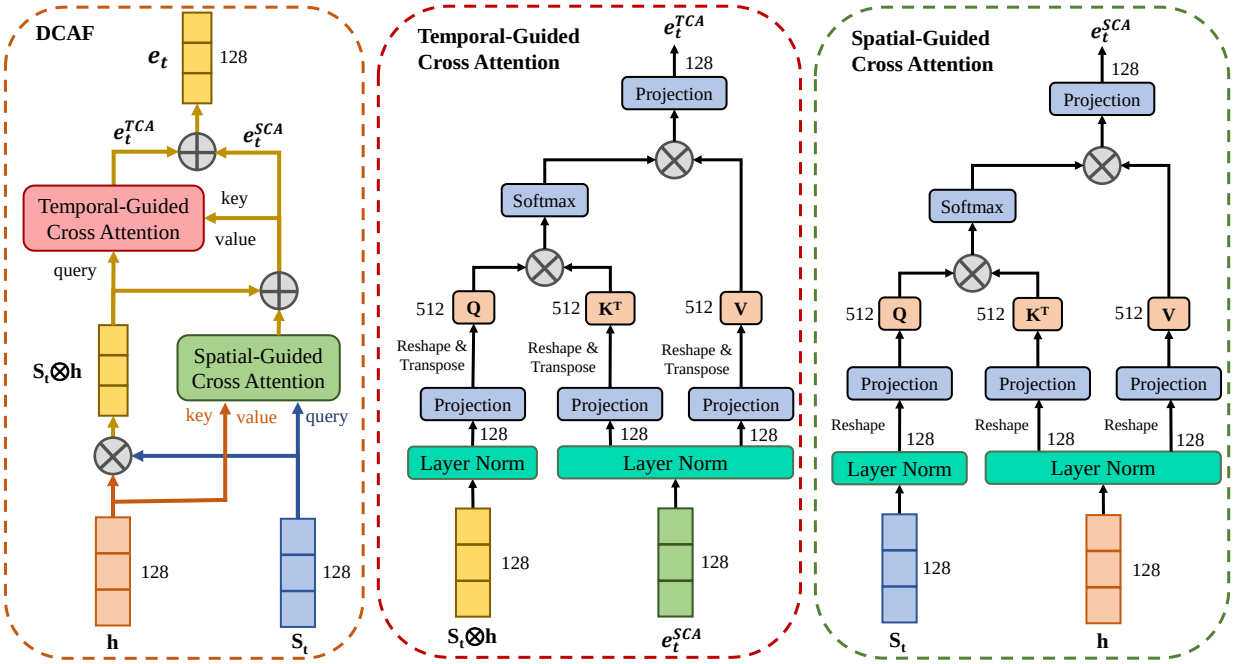


Figure 3.4: **Dual cross-attention fusion (DCAF) module and its sub-modules: Spatial-guided Cross Attention (SCA) and Temporal-guided Cross Attention (TCA).** Inputs are the spatial and temporal representations $\mathbf{S}_t$ and $\mathbf{h}$, and output is a spatio-temporal representation $\mathbf{e}_t$ which captures long-range contextual information in both temporal and spatial dimensions. $\oplus$ denotes an element-wise addition operation and $\otimes$ denotes a concatenation operation.

arbitrary time step t without time lags, as opposed to methods processing point clouds sequentially, e.g., OFlow [202]. Thanks to this advantage, the processing time of RFNet-4D++ can be optimised by calculating the spatio-temporal representations $\mathbf{e}_t$ for all the time steps t in parallel.

3.3.3 Joint Decoder

The joint decoder takes a spatio-temporal representation $\mathbf{e}_t$ and the original point cloud sequence as input then passes this input into two decoders (temporal decoder and occupancy decoder) to perform flow estimation and object reconstruction. Our temporal decoder and occupancy decoder are built upon the architecture from LPDC [261]. However, instead of decoupling the decoders as in [261], we hypothesized that jointly addressing two tasks by sharing information between corresponding decoders can leverage individual tasks. As a consequence, the close collaboration of flow estimation and object reconstruction allows some relaxation in the supervision need.

The temporal decoder operates as follows (see Figure 3.5 (a)). We first extract a spatio-temporal representation $\mathbf{e}_1$ for the first point cloud P_1 from the input sequence, using the compositional

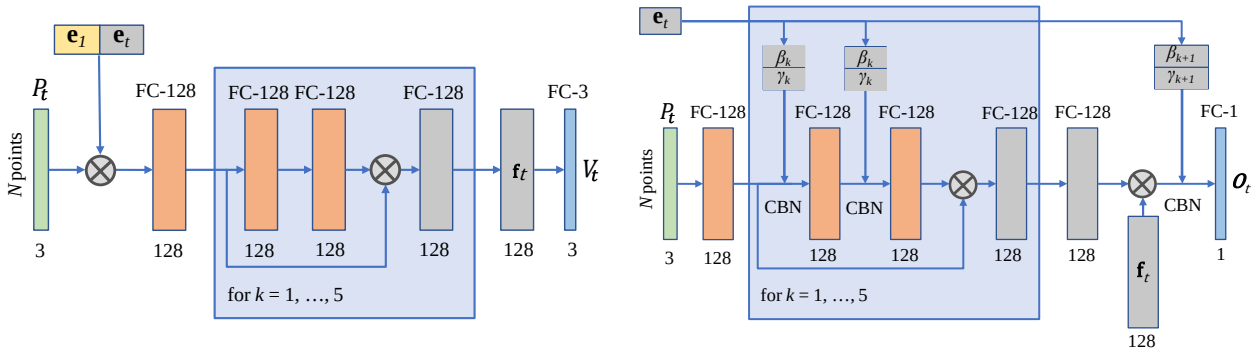
encoder. For each following point cloud P_t, we compute its spatio-temporal representation $\mathbf{e}_t$, then concatenate $\mathbf{e}_t$ with $\mathbf{e}_1$. This concatenated representation captures temporal changes of P_t in relative to P_1, and is again concatenated with all points in P_t to be processed by a series of five ResNet residual blocks [93]. Each block consists of two fully connected layers with skip connections and ReLU activation functions [79]. The outcome of these blocks is a feature map, namely $\mathbf{f}_t$. This feature map is finally passed to a fully connected layer, returning a motion field V_t describing the motion of P_t.

The occupancy decoder is slightly different from the temporal decoder (see Figure 3.5 (b)). Also different from all existing methods, our occupancy decoder works collaboratively with the temporal decoder. Particularly, input for the occupancy decoder to reconstruct the object at time step t includes a point cloud P_t, a spatio-temporal representation $\mathbf{e}_t$ (obtained from the compositional encoder), and a flow feature map $\mathbf{f}_t$ (returned by the temporal decoder). The point cloud P_t is first processed by a fully connected layer to extract a feature map. Similarly, the spatio-temporal representation $\mathbf{e}_t$ is fed to two different fully connected layers to obtain feature maps β and γ. These output feature maps (from P_t and $\mathbf{e}_t$) are passed to a series of five residual blocks, similar to those used in the temporal decoder. Following ONet [192], we apply Conditional Batch Normalization (CBN) introduced in [58, 45] to β and γ. Finally, the flow feature map $\mathbf{f}_t$ is injected into the occupancy decoder to produce an occupancy map $O_t(\mathbf{p})$, where $O_t(\mathbf{p}) = 1$ if the point $\mathbf{p}$ belongs to the object at time step t, and $O_t(\mathbf{p}) = 0$ otherwise.

Following [111], the CBN layers are implemented in the following way. First, we pass $\mathbf{e}_t$ through two fully-connected layers to obtain 128-dimensional learnable parameter vectors $\beta(\mathbf{e}_t)$ and $\gamma(\mathbf{e}_t)$. We next normalize a 128-dimensional input feature vector f_{in}^{BN} using first and second-order moments, then multiply the normalized output with $\gamma(\mathbf{e}_t)$ and finally add the bias term $\beta(\mathbf{e}_t)$ as,

$$f_{out}^{BN} = \gamma(\mathbf{e}_t)\frac{f_{in}^{BN} - \mu}{\sqrt{\sigma^2 + \epsilon}} + \beta(\mathbf{e}_t),$$

where μ and σ are the empirical mean and standard deviation (over the batch) of the input features f_{in}^{BN} and $\epsilon = 10^{-5}$ (the default value of PyTorch). Moreover, we compute a running mean over μ and σ^2 with momentum 0.1 during training. At inference time, we replace μ and σ^2 with the corresponding running mean.



(a) Temporal Decoder (b) Occupancy Decoder

Figure 3.5: **Architecture of the temporal and occupancy decoder**; $\oplus$ indicates a concatenation operation. The temporal decoder returns both a motion field V_t and a motion feature map $\mathbf{f}_t$, which is then inputted to the occupancy decoder.

3.3.4 Joint Learning

Our RFNet-4D++ is trained by jointly performing two optimization processes: unsupervision for flow estimation and supervision for object reconstruction. Existing works train flow estimation using supervised learning [70, 192, 202, 261, 116], requiring fully annotated point correspondences in training data. In this work, we propose to learn point correspondences in a point cloud sequence via an unsupervised manner, thus opening ways to new applications and more training data. Specifically, let V_t be a motion field (i.e., a set of 3D vectors) at P_t, V_t is estimated using the temporal decoder. We measure the correspondences between points in P_t and P_{t+1} via the Chamfer distance between P_{t+1} and a translated version of P_t made by V_t (i.e., $P_t + V_t$). We define our flow loss as follows,

$$\mathcal{L}_{flow} = \sum_{t} \max \left\{ \frac{1}{|P_t|} \sum_{\mathbf{p} \in P_t + V_t} \min_{\mathbf{p}' \in P_{t+1}} \|\mathbf{p} - \mathbf{p}'\|_2, \right. \\ \left. \frac{1}{|P_{t+1}|} \sum_{\mathbf{p}' \in P_{t+1}} \min_{\mathbf{p} \in P_t + V_t} \|\mathbf{p}' - \mathbf{p}\|_2 \right\} \tag{3.2}$$

The reconstruction task can be trained using a supervised approach. We use conventional binary cross entropy (BCE) loss to measure the difference between predicted occupancy maps and corresponding ground truth maps. Specifically, we define our reconstruction loss as follows,

$$\mathcal{L}_{reconstruction} = \sum_{t} \sum_{\mathbf{p} \in P_t} \mathcal{L}_{\text{BCE}} \left(O_i(\mathbf{p}), O_i^{gt}(\mathbf{p}) \right) \tag{3.3}$$

where O_i^{gt} represents the ground truth occupancy map of the point cloud P_t.

Finally, we use the following loss to train the entire RFNet-4D++,

$$\mathcal{L} = \mathcal{L}_{flow} + \lambda \mathcal{L}_{reconstruction} \tag{3.4}$$

where λ is a hyper-parameter.

To further exploit the benefit of temporal information, we train our RFNet-4D++ in both forward and backward directions in time. Particularly, we calculate the holistic temporal representation $\mathbf{h}$ for two sequences $\{P_1, ..., P_T\}$ (forward) and $\{P_T, ..., P_1\}$ (backward), and use $\mathbf{h}$ to encode the spatio-temporal representations $\mathbf{e}_t$ in both forward and backward time direction. As shown in our experiments, this training strategy improves the performance of our network in both object reconstruction and flow estimation tasks.

3.4 Experiments

3.4.1 Experimental Setup

Dataset. We trained and evaluated our method on the pre-processed data of D-FAUST dataset [11], a benchmark dataset commonly used in state-of-the-art. D-FAUST dataset contains raw-scanned and registered meshes for 129 sequences of 10 human subjects (5 females and 5 males) with various motions such as "shake hips", "running on spot", or "one leg jump". We followed the train/test split used in [202]. Specifically, we divided all the sequences in the D-FAUST dataset into three sets: training set (105 sequences), validation set (6 sequences), and test set (21 sequences). Since each sequence is relatively long (with more than 1,250-time steps) and in order to increase the size of the dataset, we sub-sampled each sequence into smaller sub-sequences of 17 to 50 time steps.

Our proposed method was also evaluated on the DeformingThing4D-Animals [156] dataset, which consists of 1494 non-rigidly deforming animations featuring 40 identities belonging to 24 categories. We follow [260] to establish a train/test split, we divided all animations into training set (1296) and test set (198). Similar to the D-FAUST [11] dataset, the train/test split is based on these identity and motion names of deforming sequences. To begin with, we initially split the dataset's animations into two categories: seen identities and unseen identities. Within the animations of seen identities, we further separate them into two subsets: seen motions of seen identities, which we

utilize as the training set, and unseen motions of seen identities, which serve as test set S1. The animations belonging to unseen identities are reserved exclusively for test set S2. As a result, the train, test set S1, and test set S2 consist of 1296, 143, and 55 deforming sequences, respectively.

Implementation Details. We implemented our method in Pytorch. We adopted Adam optimizer [131] where the learning rate γ was set to 10^{-4} and decay was set to 5,000 iterations. We empirically set λ to 0.1 in our experiments. Our RFNet-4D was trained with a batch size of 16, and on a single NVIDIA RTX 3090 GPU. We evaluated all the variants of our network (see Ablation study) on the validation set for every 2,000 iterations during the training process and used the best model of each variant on the validation set for evaluation of the variant on the test set. The training process was completed once there were no further improvements achieved. For calculating the losses during training, we randomly sampled a fixed number of 512 points in 3D space and time interval for reconstruction loss, and uniformly sampled trajectories of 100 points for flow estimation loss. More details can be found in our supplied code.

We also followed the evaluation settings used in [202, 268]. Specifically, for each evaluation, we carried out two case studies: S1 - seen individuals but unseen motions (test subjects were included in the training data but their motions were not given in the training set), and S2 - unseen individuals but seen motions (test subjects were found only in the test data but their motions were seen in the training set).

Evaluation Metrics. To measure the performance of 4D reconstruction, we applied the common volumetric IoU (Intersection over Union) and the Chamfer distance reflecting the coincidence of reconstructed data and ground-truth data. To evaluate flow estimation, we used ℓ_2-distance to measure the correspondences between estimated flows and ground-truth flows.

We use volumetric IoU and Chamfer distance for the evaluation of object reconstruction at each time step. Following [192], for each point cloud, we randomly sample 100,000 points inside or on the reconstructed result (i.e., predicted mesh) of the point cloud and determine whether these points lie inside or outside its corresponding ground truth mesh. Then the volumetric IoU is computed as the ratio of the volume of the intersection and union of the predicted mesh and ground truth mesh. The Chamfer distance is defined as the mean of accuracy and a completeness metric. The accuracy metric is defined as the mean distance of points on the predicted mesh to their nearest neighbors on the ground truth mesh. The completeness metric is defined similarly, but in the opposite direction,

i.e., given a ground truth point, its nearest neighbor on the predicted mesh is used. We estimate both accuracy and completeness metrics efficiently by randomly sampling 100,000 points from both predicted and ground truth mesh and using a KD-tree to estimate corresponding nearest neighbors from the meshes.

To evaluate flow estimation, we adopt the ℓ_2-distance used in [202]. Specifically, given a reconstructed mesh and a 3D vector field at a time step t of a point cloud P_t, we apply the vector field on vertices of the reconstructed mesh to estimate their locations at time step $t+1$. We then find the corresponding nearest points of these new locations on the ground truth mesh of P_{t+1} at time step $t+1$. These nearest points can be computed efficiently using a KD-tree. We finally measure the mean of ℓ_2-distances between estimated locations and their corresponding nearest points. Note that point correspondences are available in the D-FAUST dataset. However, since our RFNet-4D is trained to estimate motion flows in an unsupervised manner, these point correspondences are only used for evaluation. In contrast, they are used for both training and evaluation in existing methods, e.g., PSGN-4D, OFlow, and LPDC.

3.4.2 Results

We report the performance of our method in two case studies in Table 3.1. As shown in the experimental results, our method performed better in the seen individuals case study, for both object reconstruction and flow estimation. However, our method works consistently, and the differences in all performance metrics between the two case studies are marginal. For instance, the IoU difference between the two case studies is less than 4%, the differences in Chamfer distance and ℓ_2 correspondence between these case studies are about 0.005×10^{-3} and 0.015×10^{-2} respectively.

In addition to the evaluation of RFNet-4D++, we also compared it with its previous version RFNet-4D [268] and with other existing methods including PSGN-4D [70], ONet-4D [192], OFlow [202], LPDC [261], and 4DCR [116]. For the existing works, we re-trained using their released source code (for some methods, our reported results are higher than their original reported numbers). We show comparison results in Table 3.1. It can be seen that RFNet-4D++ outperforms its previous version RFNet-4D and all other baselines in 4D reconstruction using both IoU and Chamfer distance metrics. In flow estimation, RFNet-4D++ still outperforms RFNet-4D but achieves comparable performance with state-of-the-art on seen individuals, e.g., there is a slight difference (about 0.1×10^{-2}) in ℓ_2-distance from the first ranked method (LPDC). However, unlike existing

Table 3.1: **Quantitative evaluations and comparisons on the D-FAUST dataset** of RFNet-4D++ and existing methods in object reconstruction and flow estimation. The first five methods are trained with point-to-point correspondence labels (i.e., fully supervised learning), while the last two methods (RFNet-4D and RFNet-4D++) are trained without point-to-point correspondence supervision. We report the volumetric IoU (higher is better), Chamfer distance (lower is better), and correspondence ℓ_2 distance (lower is better). The notation '-' means no results, e.g., PSDN-4D does not perform reconstruction, and ONet-4D and 4DCR do not predict point correspondences across time. For each evaluation metric, the best and second best performances are bold and underlined, respectively.

Method	**Test set S1:** Seen Individuals, Unseen Motions			**Test set S2:** Unseen Individuals, Seen Motions		
	IoU↑	**Chamfer**↓ ($\times 10^{-3}$)	**Corres.**↓ ($\times 10^{-2}$)	**IoU**↑	**Chamfer**↓ ($\times 10^{-3}$)	**Corres.**↓ ($\times 10^{-2}$)
PSGN-4D [70]	-	0.619	1.108	-	0.688	1.329
ONet-4D [192]	0.771	0.592	-	0.683	0.701	-
OFlow [202]	0.817	0.177	<u>0.869</u>	0.736	0.274	1.084
LPDC [261]	0.851	0.153	**0.780**	0.762	0.219	0.987
4DCR [116]	0.817	0.167	-	0.697	0.222	-
RFNet-4D [268]	<u>0.855</u>	<u>0.151</u>	0.883	<u>0.816</u>	<u>0.159</u>	**0.864**
RFNet-4D++	**0.866**	**0.146**	0.871	**0.835**	**0.142**	<u>0.886</u>

works following supervised paradigm, RFNet-4D++ is trained in unsupervised fashion requiring no point-to-point correspondence labelling. Table 3.1 also show that, both RFNet-4D and RFNet-4D++ methods stand out in flow estimation on unseen individual sequences, proving the ability of our architecture and joint learning of spatio-temporal representations of novel object shapes.

We report the results of RFNet-4D++, RFNet-4D, and LPDC on the DeformingThing4D-Animals dataset in Table 3.2, which also shows consistent performance trend with the D-FAUST dataset (i.e., reconstruction of unseen individuals with seen motions is more challenging than its counterpart case study). The results clearly demonstrate the dominance of RFNet-4D++ over its previous version RFNet-4D and LPDC in 4D reconstruction, evident by its superior performance on both the IoU and Chamfer distance metrics. In flow estimation, RFNet-4D++ outperforms RFNet-4D on both the case studies. The supervised approach (LPDC) still shows its advantage over the unsupervised one (RFNet-4D, RFNet-4D++) on the ℓ_2-distance, though the difference is subtle (e.g., the averaged ℓ_2-distance difference between the results of LPDC and RFNet-4D++ is about 0.041×10^{-1} on the test set S1 and 0.022×10^{-1} on the test set S2). However, labelling of point correspondences in

Table 3.2: **Quantitative evaluation on the DeformingThing4D-Animals dataset** of RFNet-4D++ and existing methods object reconstruction and flow estimation. We report the volumetric IoU (higher is better), Chamfer distance (lower is better), and correspondence ℓ_2 distance (lower is better). For each evaluation metric, the best and second best performances are bold and underlined, respectively. Note that, for fully supervised learning-based methods, e.g., PSGN-4D, ONet-4D, OFlow, 4DCR, we showcase only LPDC as we found it performs best among them.

Method	**Test set S1:** Seen Individuals, Unseen Motions			**Test set S2:** Unseen Individuals, Seen Motions		
	IoU↑	**Chamfer**↓ ($\times 10^{-3}$)	**Corres.**↓ ($\times 10^{-1}$)	**IoU**↑	**Chamfer**↓ ($\times 10^{-3}$)	**Corres.**↓ ($\times 10^{-1}$)
LPDC [261]	0.759	0.153	**0.209**	0.569	0.132	**0.361**
RFNet-4D [268]	<u>0.774</u>	<u>0.149</u>	0.305	<u>0.569</u>	<u>0.127</u>	0.387
RFNet-4D++	**0.789**	**0.134**	<u>0.251</u>	**0.582**	**0.121**	<u>0.383</u>

supervised learning is extremely labour-intensive and requires complicated data capture setup. In our contrast, our method, due to its unsupervised learning nature, can open up possibilities for broader applicability and scalability in various scenarios where labelled data may be limited or unavailable.

We visualize several results of our methods and existing ones in Figure 3.7 and Figure 3.8. To illustrate these results, we apply the Multiresolution IsoSurface Extraction (MISE) algorithm [192] and the Marching Cubes algorithm [176] on reconstructed occupancy maps to generate surface meshes. As shown in the results, compared with existing methods, RFNet-4D++ achieves better reconstruction quality with more-detailed geometry recovery. For instance, in Figure 3.7, the reconstructed hands produced by our method are more complete. In addition, by coupling both spatial and temporal information, the poses of body parts, e.g., the head, and the lower arms, are well preserved by our method (in reference to corresponding ground truth meshes). RFNet-4D++also shows better flow estimation than existing works as clearly demonstrated in the predicted flows in the two hands in Figure 3.7.

3.4.3 Ablation Studies

In this section, we present ablation studies to verify various aspects of our method including: ❶ fusion technique used in the compositional encoder, ❷ variants of RFNet-4D++, ❸ impact of spatial and temporal resolutions, and ❹ effectiveness of RFNet-4D++'s components when applied in a

Table 3.3: **Fusion techniques**. Ablation study on various fusion techniques used to fuse the spatial and temporal features in the Compositional Encoder. For each evaluation metric, the best performance is highlighted. "CR" means cross-attention.

Dataset	Fusion technique	Test set S1: Seen Individuals, Unseen Motions			Test set S2: Unseen Individuals, Seen Motions		
		IoU↑	Chamfer↓ ($\times 10^{-2}$)	Corres.↓ ($\times 10^{-1}$)	IoU↑	Chamfer↓ ($\times 10^{-2}$)	Corres.↓ ($\times 10^{-1}$)
D-FAUST	concat	0.8547	0.0150	0.0883	0.8157	0.0159	**0.0864**
	single CR	0.8543	0.0147	0.0915	0.8221	0.0148	0.0933
	dual CR	**0.8658**	**0.0146**	**0.0871**	**0.8345**	**0.0142**	0.0886
Deform Thing4D	concat	0.7735	0.1491	0.3049	0.5693	0.1271	0.3871
	single CR	0.7745	0.1560	0.3288	0.5763	0.1356	0.3904
	dual CR	**0.7892**	**0.1342**	**0.2505**	**0.5821**	**0.1204**	**0.3832**

related baseline.

Fusion techniques

We experimented our RFNet-4D++ with well-known fusion manners to fuse the spatial and temporal information in the compositional encoder. Those fusion techniques include concatenation ("concat") used in our previous RFNet-4D [268], single cross-attention ("single CR") [49], and our proposed dual cross-attention ("dual CR"). Results of this experiment are presented in Table 3.3. We observed that, the single cross-attention performs on par with or slightly better than the simple concatenation operation. However, the dual cross-attention clearly and consistently boosts up the performance on both the datasets and in both the case studies (S1 and S2). This shows the ability of the dual cross-attention in learning of long-range contextual information.

Variants of RFNet-4D++

We conducted a series of experiments on different variants of our RFNet-4D++. ***First***, we verified our joint learning of spatio-temporal representations for 4D point cloud reconstruction and flow estimation by comparing it with the approach tackling these two tasks independently. ***Second***, we proved the improvement of flow learning in both forward (FW) and backward (BW) directions. ***Third***, we compared different distance metrics including the Sliced Wasserstein distance (SWD)

Table 3.4: **Ablation study** on various settings of our RFNet-4D++ on the D-FAUST dataset. For each evaluation metric, the best performance is highlighted.

Variant	**Detailed setting**	**IoU** ↑	**Chamfer**↓ ($\times 10^{-3}$)	**Corres.**↓ ($\times 10^{-2}$)
Separated tasks	Only temporal flows	-	-	1.5519
	Only spatial points	0.7712	0.5921	-
Learning direction	Only FW learning	0.4988	2.4887	3.5868
Flow loss	SWD loss	0.4305	4.4621	4.0711
	HD loss	0.7953	0.2103	1.3017
Flow learning	Supervised	0.8656	**0.0927**	**0.8125**
Full (RFNet-4D++)	Joint learning, FW-BW, Chamfer loss, unsupervised	**0.8658**	0.1462	0.8709

loss and the Hausdorff distance (HD) loss for the implementation of the flow loss in Equation 3.2, and validated our choice, *i.e.* the Chamfer distance loss. ***Fourth***, we experimented our model in both unsupervised and supervised fashions for flow estimation though our method is intentionally designed for unsupervised paradigm.

We summarise all the variants in Table 3.4. Note that, for each variant, only one factor being investigated was customised while other factors remained unchanged. For the variants using either temporal or spatial information (see the first two rows in Table 3.4), only the corresponding encoder and decoder (i.e., spatial/temporal encoder and decoder) were activated while the counterpart encoder and decoder were frozen. These variants correspond to solving the flow estimation and reconstruction tasks separately. To experiment our method with supervised learning approach for flow estimation, we followed the settings used in OFlow [202]. In particular, we used ground-truth point-to-point correspondences from the training data and ℓ_2 distance for the motion loss, i.e., replacing the Chamfer distance in Equation 3.2 by ℓ_2 distance. Note that, the D-FAUST dataset is fully annotated with point-to-point correspondences. When training our model using unsupervised mode, those point-to-point correspondences were not used. Experimental results in Table 3.4 clearly confirm the design of our RFNet-4D++ in both object reconstruction and flow estimation.

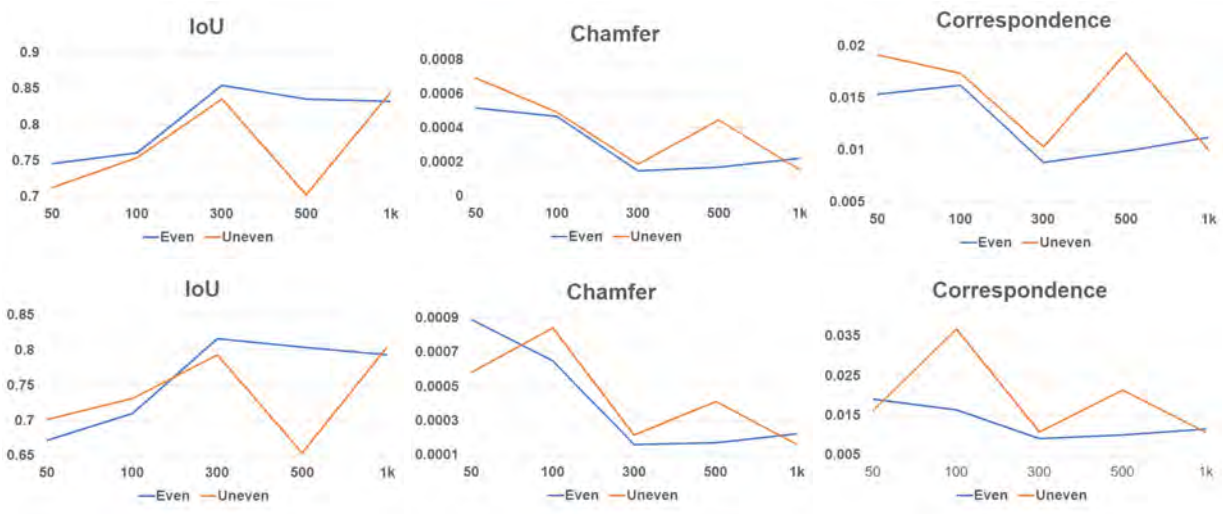


Figure 3.6: **RFNet-4D++ under various spatial/temporal resolutions** on both seen (top row) and unseen (bottom row) individuals test splits on the D-FAUST dataset in both object reconstruction and flow estimation.

Impact of spatial and temporal resolutions

We tested our method under various spatial and temporal resolutions on both seen and unseen individuals test splits, and in both reconstruction and flow estimation tasks. For spatial resolutions, we randomly sampled an input point cloud with 50, 100, 300 (the input used in all experiments), 500, and 1k point trajectories from ground-truth surfaces. For each spatial resolution, like LPDC, we varied the temporal resolutions by sampling each point cloud sequence in two ways: even sampling (as used in all experiments) and uneven sampling (where we randomly selected 6 frames with large variations between adjacent frames from a 50-time step segment). We present the results of this experiment in Figure 3.6.

The relationship between input point density with the performance of object reconstruction and flow estimation is clearly shown in Figure 3.6. In general, increasing the number of input points within the range of 50-300 points leads to improved performance. However, when the input density exceeds 300 points (which was the case in all experiments), a noticeable drop in performance occurs. This decline can be attributed to the complexities involved in establishing flow correspondences for denser point clouds. Despite this decrease, the performance of denser input points still outperforms sparser input points, indicating their relative advantage even in challenging scenarios. For the temporal resolution-related experiment, it is worth noting that even or uniformly sampled input points consistently yield superior performance compared to uneven sampling points. This is likely

Table 3.5: **Variants of LPDC using RFNet-4D++'s components** on the D-FAUST dataset. For each evaluation metric, the best performance is highlighted.

Baseline	**IoU** ↑	**Chamfer** ($\times 10^{-3}$) ↓	**Corres.** ($\times 10^{-2}$) ↓
LPDC-Add	0.8348	0.1865	**0.5539**
LPDC-Chamfer	0.6231	0.7359	1.3187
LPDC [261]	0.8511	0.1526	<u>0.7803</u>
RFNet-4D [268]	<u>0.8547</u>	<u>0.1504</u>	0.8831
RFNet-4D++	**0.8658**	**0.1462**	0.8709

due to the fact that point locations and flow correspondences can undergo significant changes or exhibit large variations, making flow estimation more challenging when unevenly distributed points are used.

Effectiveness of RFNet-4D++'s components

Several components of our proposed RFNet-4D++ are built upon the architecture of LPDC [261], e.g., the temporal decoder and occupancy decoder. To better understand the effectiveness of these components, we applied them to customise LPDC as follows: a) LPDC-Add: Like our RFNet-4D++, we added the features from the temporal decoder as an additional input to the occupancy decoder on top of LPDC; b) LPDC-Chamfer: we switched the ℓ_1 loss used in LPDC to our Chamfer distance loss. The motivations for integrating the temporal features into the occupancy decoder are as follows. (i) We hypothesized that jointly addressing two tasks by sharing information between corresponding decoders can leverage the individual tasks. (ii) As a consequence, the close collaboration of flow estimation and object reconstruction allows some relaxation of the supervision need. This is realised by tackling the flow estimation task in an unsupervised manner. In addition, our RFNet-4D++ also differs from LPDC in the joint prediction of motion and shape. Specifically, although both LPDC and our method jointly learn spatio-temporal representations for 4D point clouds, LPDC predicts the motion flows and object shapes in a decoupling way as the LPDC's decoders work independently. In contrast, our RF-Net4D++'s decoders collaborate closely by sharing information and thus can leverage each other. We report the results of this ablation study in Table 3.5.

Table 3.6: **Space and time complexity** of our method and existing ones on the D-FAUST dataset.

Method	**Memory** (GB)	**Training** (sec/iter)	**Inference** (sec/seq)
OFlow [202]	3.96	4.65s	0.95s
LPDC [261]	11.90	2.09s	0.44s
RFNet-4D [268]	14.20	1.33s	0.24s
RFNet-4D++	16.87	1.54s	0.29s

3.4.4 Complexity Analysis

We analysed the memory footprint and computational efficiency of our RFNet-4D++, its previous version RFNet-4D, and several existing models including OFlow [202] and LPDC [261] (best performed method other than our RFNet-4D and RFNet-4D++). In this experiment, we trained all the models with a batch size of 16, using a sequence of 17-time steps with consistent intervals. We report the memory footprint, training, and inference time of all the methods in Table 3.6. For the training time, we computed the average of batch training time throughout the first 100k iterations of training (seconds per iteration). For the inference time, we reported the average time required to infer using a batch size of 1 for 1k test sequences (seconds per sequence). As shown in Table 3.6, although our model take a larger memory footprint for training, its training time is approximately 3.5 times and 1.6 times faster than that of OFlow and LPDC respectively. Similarly, our model performs 1.9 times and 4 times faster than OFlow and LPDC in inference. We found that OFlow takes a much longer time for training since it makes use of an Ordinary Differential Equation solver requiring intensive computations and gradually increasing the number of iterations to fulfill error tolerance.

3.5 Discussion and Conclusion

This work proposes RFNet-4D++, a network architecture for jointly reconstruction of 3D objects and estimation of temporal flows from dynamic point clouds. We develop a compositional encoder effectively capturing informative spatio-temporal representations for 4D point clouds, and devise a joint learning paradigm leveraging sub-tasks to improve overall performance. We evaluated our method and compared it with existing works on two benchmark datasets including D-FAUST and DeformingThing4D datasets. Experimental results demonstratec the effectiveness and efficiency of our method in comparison with the current state-of-the-art.

There is also room for future research. First, we found that existing 4D reconstruction methods often suffer from large displacements between data frames. Second, their reconstruction quality tends to drop over time due to accumulated errors. It is also worthwhile to study 4D reconstruction for different types of objects and with more challenging input data types, e.g., LiDAR point clouds that are commonly used in autonomous driving applications.

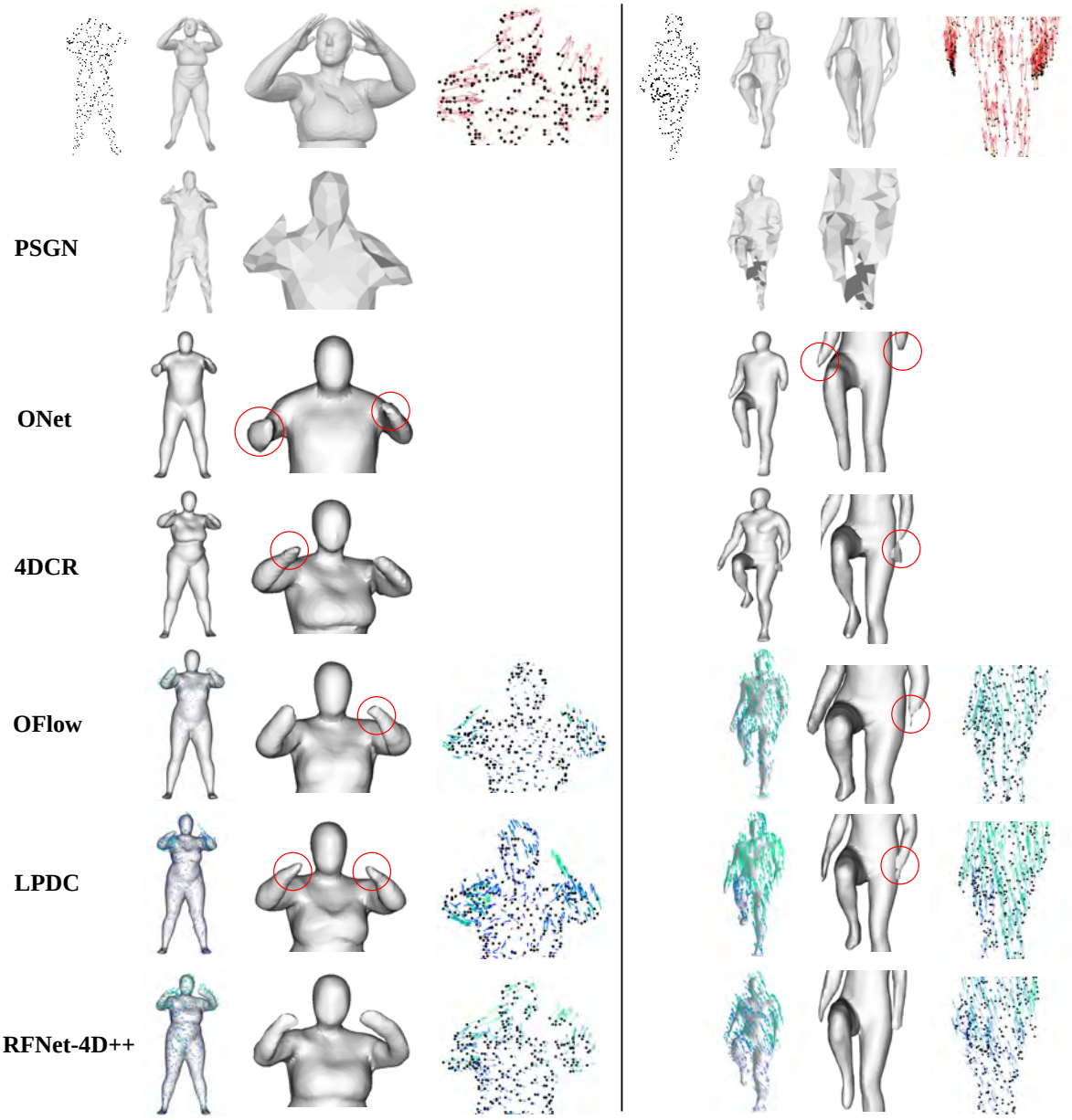


Figure 3.7: **Qualitative evaluation** of our method and existing methods on the DFAUST dataset. The first row includes (from left to right): input point cloud, ground truth mesh of entire body, ground truth mesh of upper/lower body, and ground-truth flows (darker vectors show stronger motions). Each following row represents corresponding reconstruction and flow estimation results. Severe errors are highlighted.

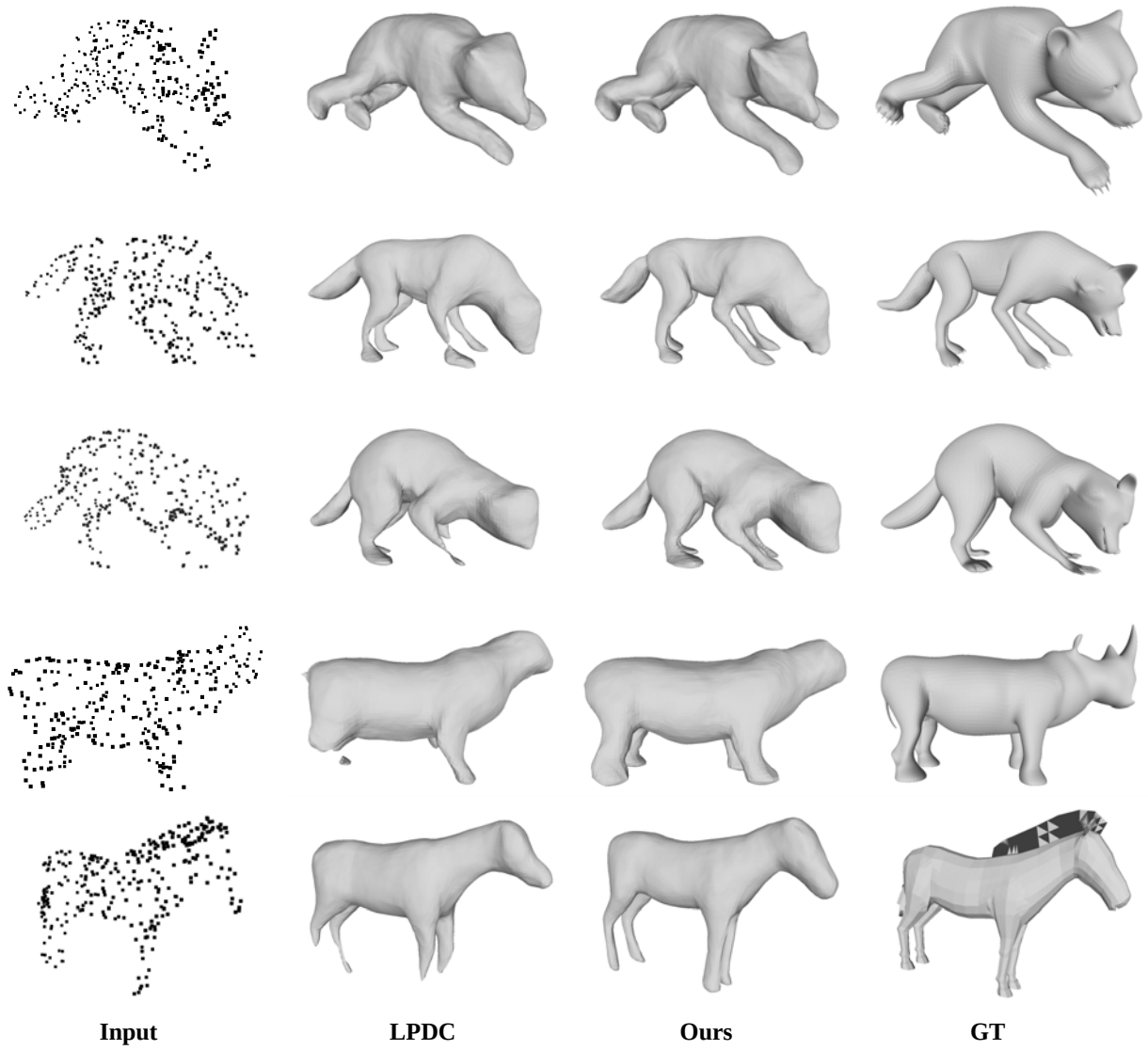


Figure 3.8: **Qualitative evaluation** of our method and LPDC [261] on the DeformingThing4D-Animals [156] dataset. Note that only LPDC is showcased here as we found it significantly outperforms methods other than our RFNet-4D and RFNet-4D++, e.g., PSGN-4D, ONet-4D, OFlow, 4DCR.

CHAPTER 4

TRANSPARENT AND REFLECTIVE OBJECT SEGMENTATION

4.1 Introduction

Figure 4.1: Sample cases of the types of transparent objects present around us.

Glass is used extensively to construct opaque and see-through objects, such as windows, vitrines, bottles, and walls. Segmenting images containing transparent materials might be challenging because their appearance depends on the surrounding visual background. Some samples of different transparent objects in real life are shown in Figure 4.1. Most systems used in robotics research [307, 308, 337] rely on sensor fusion techniques involving sonars or lidars, but the methods have often struggled to detect transparent objects, misinterpreted reflections as actual objects, leading to scan matching issues. This is because transparent objects present the properties of refraction and reflection, which throwback light and the appearance of the surrounding areas, so they mislead robot sensors

and negatively impact robot navigation, depth estimation, and 3D reconstruction. Hence, visual systems must deal with reflective surfaces, which would help them accurately identify glass barriers for effective collision prevention in workplaces, supermarkets, or hotels. Furthermore, in domestic and professional settings, visual systems should also be able to navigate fragile items such as vases and glasses. Therefore, a practical, robust, cost-effective, vision-based approach for transparent object segmentation is essential. However, current semantic segmentation algorithms [306, 348, 323, 24, 331] and even powerful foundation models, such as SAM [132], Semantic SAM [150, 22] were not designed to address transparent and reflective objects, resulting in decreased performance in the presence of such objects. It is important to note that the SAM model does not include semantics, or in other words, it cannot yield masks with semantic information. The SAM model also presents the challenge of over-segmentation, thereby leading to a higher likelihood of false positives (please check Section 4.4.6 for more detail).

Recent works on human visual system [127, 237, 236] prove that **"humans rely on specular reflections and boundaries as key indicators of a transparent layer"**. In the existing literature for segmentation, recent methods for segmenting transparent or glass objects have been proposed with different strategies for better results in dealing with glass objects. These strategies include focusing on the object's edges [308, 337, 256], using depth information [256, 239], analyzing reflections [162], looking at how light polarizes [304, 188], and utilizing the object's context or semantic information [164]. However, techniques that rely on polarization, depth, and semantic information [256, 239, 304, 188, 164] often need special equipment to gather data or a lot of human work to label the data, which isn't very efficient. As far as we know, no method has combined both visual cues of boundary and reflection to improve segmentation performance.

Therefore, we will focus on capturing the two visual cues into the segmentation models: boundary localization for shape inference and reflections for glass surface recognition. We introduce an *efficient transformer-based architecture* tailored for segmenting transparent and reflective objects along with general objects. Then, our method captures the **glass boundaries** based on their geometric cues and the **glass reflections** based on the appearance cues in an enhanced feature module in our network. In doing so, we developed a Boundary Feature Enhancement (BFE) module to learn and integrate glass boundary features that can help the localization and segmentation of the glass-like regions. We supervise this module by a new boundary loss that utilizes the Sobel kernel to extract boundaries based on the gradients of the predictions and ground-truth objects' masks. Then, we

introduce a Reflection Feature Enhancement (RFE) module, which decomposes reflections into foreground and background layers, providing the network with additional features to distinguish between glass-like and non-glass areas. By harnessing the power of transformer-based encoders and decoders, our framework could capture long-range contextual information, unlike previous methods, which relied heavily on stacked attention layers [73, 322] or combined CNN backbones with transformers [308, 348, 289]. These long-range visual cues are essential to reliably identify transparent objects, especially when they lack distinctive textures or share similar content with their surroundings [308]. More importantly, we demonstrate that our method is robust to both transparent object segmentation and generic semantic segmentation tasks, with state-of-the-art performance for both scenarios across various datasets.

In summary, our contributions are as follows:

- We introduce **TransCues**, an efficient transformer-based segmentation architecture that segments both transparent, reflective, and general objects.
- We propose the Boundary Feature Enhancement (BFE) module and boundary loss, improving the accuracy of glass detection performance.
- We present the Reflection Feature Enhancement (RFE) module, facilitating the differentiation between glass and non-glass regions.
- We conduct extensive experiments to demonstrate our method's competitive performance on diverse tasks, *e.g.* semantic glass segmentation, glass and mirror segmentation, and generic semantic segmentation.

4.2 Related Works

This section will discuss recent works for Transparent Object Sensing and Segmentation, Mirror Segmentation, and Transformer in Semantic Segmentation.

4.2.1 Transparent Object Sensing and Segmentation

In the transparency settings, the color intensities of both glass and their background often match, making it challenging to differentiate between them. Traditional visual aid systems, enhanced with ultrasonic sensors and RGB-D cameras, were developed to effectively identify transparent barriers like glass and windows [110]. On raw images, existing works have explored the use of

transmission differences [203], reflection cues [162], and polarization [304, 188] for detecting transparency. Moreover, transparency segmentation methods [308, 90, 186] cater to a range of objects from opaque entities like windows and doors to see-through items such as cups and eyeglasses, focusing on discerning reflections and their boundaries to detect and define transparent surfaces accurately. Recently, [308] introduced the Trans10K-v2 dataset, which prompts new research directions beyond conventional sensor fusion for transparent objects. This includes AdaptiveASPP [15] for enhanced feature extraction and EBLNet [90] for improved global form representation. Furthermore, Trans4Trans [337] is proposed to provide a lightweight, general network for real-world applications. *Drawing on these innovations*, our work aims to create an efficient, robust, transparent object segmentation solution suitable for general semantic segmentation and practical uses like robot navigation.

4.2.2 Mirror Segmentation

Closely related to glass segmentation is mirror segmentation, in which recent models have introduced high-level concepts to improve detection and localization. For precise localization, SANet [84] utilizes the semantic relationships between mirrors and their surrounding environment. SATNet [106] capitalizes on the natural symmetry between objects and their mirror reflections to accurately identify mirror locations. VCNet [258], on the other hand, explores 'visual chirality'—a unique property of mirror images—and incorporates this through a specialized transformation process for effective mirror detection. Lastly, HetNet [96] introduces a unique model that combines a contrasted module for initial mirror localization and a reflection semantic logical module for semantic analysis. *Unlike existing works*, we consider mirror segmentation as a subproblem in glass segmentation that our proposed framework can also address effectively.

4.2.3 Transformer in Semantic Segmentation

Since its introduction in natural language processing, transformers have been adopted and further investigated for computer vision tasks. One of the pioneers is Vision Transformer (ViT) [56], which applies transformer layers to sequences of image patches. SETR [348] and Segmenter [251] take inspiration from ViT and directly add upsampling and segmentation heads to learn long-range context information from the initial layer. MaX-DeepLab [281], and MaskFormer [32] study 2D image segmentation through the perspective of masked prediction and classification based on

recent advances of object detection using transformers [16]. As a result, several transformer-based methods for dense image segmentation have been developed [172, 306]. Pyramid architectures of vision transformers have been proposed by PVT [289] and SegFormer [306] as a method for gathering hierarchical feature representations. Both ECANet [322] and CSWin transformer [52] recommend applying a self-attention mechanism in either vertical or horizontal stripes to gain advanced simulation capacity while minimizing computing overheads. NAT [87], on the other hand, aims to simplify the standard attention mechanism, resulting in faster processing and reduced memory requirements. Recent methods have been trying to match the performance of transformer models using advanced CNN architectures. MogaNet [155] introduces two feature mixers with depthwise convolutions, efficiently processing middle-order information across spatial and channel spaces. InternImage [288] utilizes deformable convolution, providing a large effective receptive field essential for tasks like detection and segmentation, and offers adaptive spatial aggregation based on input and task specifics. *Collectively*, these approaches signify a shift towards more efficient, task-tailored CNN models that strive to replicate the success of transformers in various computer vision applications.

4.3 Our Proposed Method

4.3.1 Preliminary

Given an RGB image, represented as $\mathbf{I} \in \mathbb{R}^{H \times W \times 3}$, where H and W respectively denote the image height and width, glass segmentation aims to segment this image into semantic labels at each pixel, which can be expressed as $\mathbf{F} \in \mathbb{R}^{H \times W \times n_{class}}$, where n_{class} represents the number of classes. While glass segmentation can be typically defined in the binary space, defining it along with glass, mirror, and general non-glass objects makes for a semantic segmentation problem.

Existing works have not considered boundary and reflective cues in the same framework. In particular, while boundary or edge information was deemed to be captured in [307, 188, 256] for glass segmentation, the reflective information of glass objects was not considered of high priority because of negligible reflections in the datasets. Meanwhile, mirror segmentation has been chiefly analyzed regarding symmetric reflection to detect the mirroring in the image [324, 163, 96, 106].

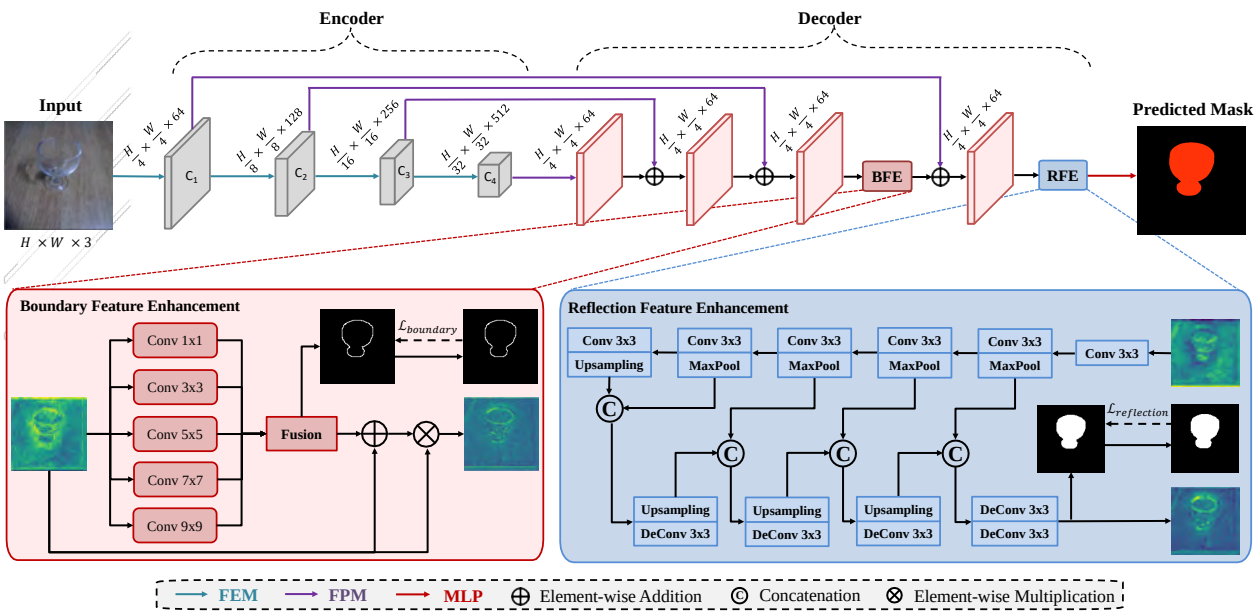


Figure 4.2: **Overview of our TransCues method.** An RGB image is processed by four FEM modules in the encoder for multi-scale feature extraction. These features are then refined by the decoder's FPM, BFE, and RFE modules, culminating in semantic labels via an MLP. Our main contributions, BFE and RFE modules, are elaborated in Section 4.3.3 and Section 4.3.4.

4.3.2 TransCues: Revealing Transparency via Edge and Reflection

As depicted in Figure 4.2, our network aims to segment glasses in an image into their corresponding labels at each pixel. To handle variations in input image sizes across datasets, we standardize the resolution to either 512×512 or 768×768, ensuring consistent position embedding dimensions throughout both training and testing phases. Our network's architecture adheres to the well-established encoder-decoder structure, comprising the Feature Extraction Module (FEM), Feature Parsing Module (FPM), Boundary Feature Enhancement (BFE), and Reflection Feature Enhancement (RFE) modules. Precisely, the FEM module, based on the PVT architecture [289, 290] in the encoder, efficiently captures multi-scale long-range dependency features from the input image. Drawing inspiration from [337], the FPM module offers a lightweight alternative to the FEM module, capturing detailed to abstract representations of transparent objects across C_1 to C_4. The detail of FEM and FPM can be found in Figure 4.3.

To enrich the feature learning capabilities of our network, we propose to capture the glass boundaries by a *geometric cue* (BFE module) and the glass reflections based on an *appearance cue* (RFE module) to differentiate glass from non-glass regions. Boundaries often present as high-contrast edges around transparent objects, a characteristic that aligns well with human visual

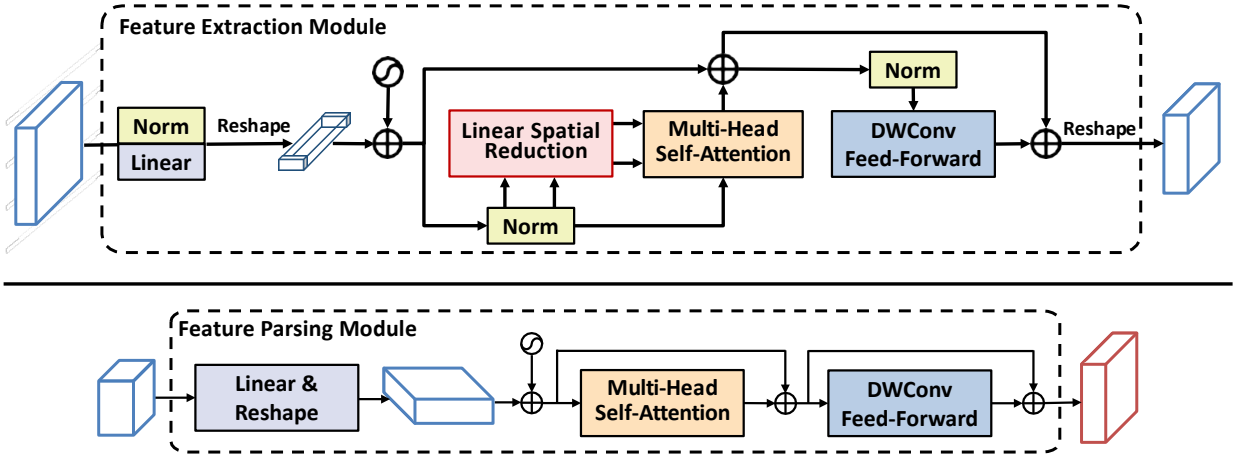


Figure 4.3: The architecture of the Feature Extraction Module (top) in our encoder and Feature Parsing Module (bottom) in our decoder. Zoom in for better visualization.

perception. In detail, we employ the BFE module to amplify the boundary characteristics inherent in transparent features. This enhancement of boundary cues facilitates more accurate segmentation of transparent objects.

On the other hand, reflections on glass surfaces may not always be prominent, making them a challenge in designing glass segmentation. Following this, we feed the boundary-enhanced features into the RFE module. This step is crucial because while most transparent objects exhibit reflections, not all reflective objects are transparent. The RFE module thus plays a pivotal role in distinguishing between these two categories. These cues enhance our network's ability to capture fine details and long-range contextual information for transparent features.

Consequently, as image data progresses through our decoder, it integrates contextual information of varying resolutions, preserving the fine-grained information of transparent and reflective features. By systematically addressing both boundary and reflection cues, our network achieves a nuanced and effective approach to segmenting these challenging object types. Finally, a compact MLP layer is employed to predict semantic labels for each pixel. As shown in Figure 4.4, throughout each stage in the encoder and the boundary and reflection modules, our feature maps show that the glass region, including its defining boundary and reflection, is clearly distinguishable from non-glass surfaces. The following sections will discuss more details of the BFE and RFE modules.

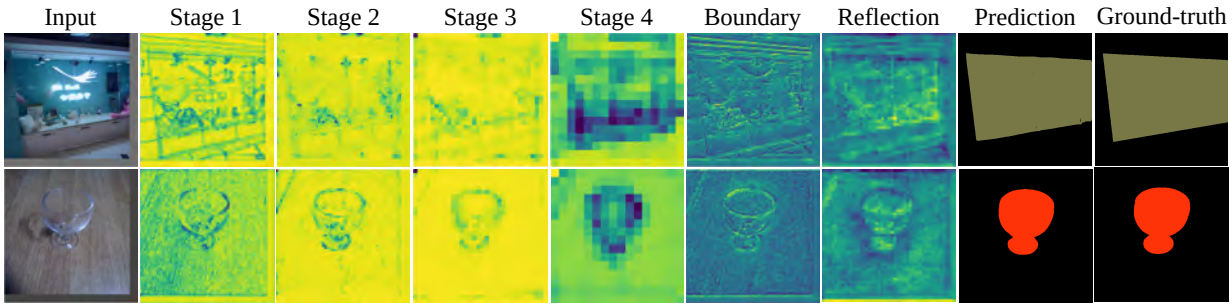


Figure 4.4: Visualization of feature maps of our method. Zoom in for better visualization.

4.3.3 Boundary Feature Enhancement Module

Inspired by human perception, incorporating boundary information can significantly benefit segmentation and localization tasks involving glass recognition [90, 186]. To implement this concept, the BFE module is based on ASPP module [24, 186], yet more specialized towards identifying and integrating boundary characteristics of glass into our transformer architecture. Contrary to the approach in [307], which uses an extra boundary stream (an encoder-decoder branch) for boundary feature extraction and integration with primary stream features, our BFE module is more streamlined. It derives boundary features directly from the targeted input features, bypassing the necessity for an additional stream. As shown in Figure 4.2, the BFE module is designed to enhance feature learning before the last layer of our decoder so that the reflection module can subsequently improve the features in the next step. We empirically found that this placement of the BFE module has better performance and reduced memory usage compared to the placement of BFE at earlier decoder layers (please check Section 4.4.5 for more detail).

The BFE begins by taking input features $\mathcal{X}_0$. These features are then processed through four parallel blocks, each dedicated to extracting multi-scale boundary features $\mathcal{F}_i(.)$ for $i = 1, 2, 3, 4, 5$. Within each block, a convolution layer ($C(.)$) with different kernels and paddings is followed by batch normalization ($BN(.)$) and ReLU activation ($ReLU(.)$) operations, resulting in $\mathcal{F}i = ReLU(BN(C(X)))$. These multi-scale boundary features are subsequently fused using the Fusion module ($\mathcal{F}_{fuse} = C(\mathcal{F}_1 + \mathcal{F}_2 + \mathcal{F}_3 + \mathcal{F}_4 + \mathcal{F}_5)$), effectively aggregating shape properties and forming the glass boundary features. The output of the Fusion module then undergoes a convolutional layer to predict the boundary map, supervised by the Boundary loss. Finally, the enhanced boundary features $\mathcal{X}_e$ are obtained by aggregating the output of the Fusion module with the input features to

precisely locate glass regions, especially their boundaries, as expressed by the following equation:

$$\mathcal{X}_e = (\mathcal{F}_{\text{fuse}}(\mathcal{F}_i(X)) + 1) \times \mathcal{X}_0 \tag{4.1}$$

where $+$ and $\times$ denote element-wise addition and multiplication, respectively.

Boundary loss. The Sobel kernel, sometimes called the Sobel-Feldman filter, is widely used in image processing and computer vision, mainly for edge detection. It highlights image boundaries by analyzing the 2D gradient and emphasizing high spatial frequency regions. Our Boundary loss ($\mathcal{L}_b$) leverages the Sobel filter to measure how closely the gradients of a predicted mask match those of the ground truth mask, employing the Dice loss [195]:

$$\mathcal{L}_b = \text{dice}(\nabla_x \hat{M} \oplus \nabla_y \hat{M}, \nabla_x M_{GT} \oplus \nabla_y M_{GT}) \tag{4.2}$$

where $\hat{M}$ is predicted object mask and M_{GT} is ground truth object mask. ∇_x and ∇_y denote the gradient along x-axis and y-axis computed by the Sobel filter. $\oplus$ represents the combination of the gradient maps into a single feature map. In our implementation, we define the combination $\oplus$ by:

$$a \oplus b = \max\left(\frac{1}{2}(a+b), \tau\right) \tag{4.3}$$

where τ is set to 0.01 to reduce noise in the gradient maps.

4.3.4 Reflection Feature Enhancement Module

To enhance the recognition of glass surfaces, we introduce the Reflection Feature Enhancement (RFE) module, capitalizing on the high reflectivity of glass when illuminated by light. These reflections provide valuable cues for recognizing glass surfaces in images [324, 187]. Note that if the reflection on the glass surface exhibits insufficient strength (poor or weak) to be discerned by our RFE module, our model may encounter challenges in accurately detecting the glass surface. In this scenario, correctly identifying glass surfaces poses a challenge, even for humans. Please check Section 4.4.6 for discussion and analysis on the need of the RFE module.

In our design, the RFE module is placed after the last layer of the decoder, after the boundary feature enhancement module (please check Section 4.4.5 for more detail). The RFE module employs a sophisticated convolution-deconvolution architecture [327], which takes input features Y and produces an enhanced feature map Y_e. This architecture allows the module to capture and process

information at multiple levels of abstraction, which is essential for handling complex visual cues like reflections. Unlike the other reflection removal model [339, 53, 249] that primarily addresses global reflections (assuming the entire input image is covered by glass), our RFE module targets detecting local reflections to locate glass surfaces.

In detail, the encoder network $\mathcal{E}$ is responsible for extracting relevant features from the input. It consists of five blocks, each composed of a convolutional layer followed by batch normalization, ReLU activation, and either a Max-Pooling $\mathcal{P}_{max}(.)$ or Upsampling layer $\mathcal{P}_{up}(.)$. Each encoder block can be defined as follows:

$$\mathcal{E}_i = \mathcal{P}^i\left(\text{ReLU}\left(\text{BN}\left(\text{C}(\mathcal{E}_{i-1})\right)\right)\right), \quad i \in [1..5] \tag{4.4}$$

where $\mathcal{E}_0 = \text{C}(Y)$, $\mathcal{P}^i$ is the Max-Pooling or Upsampling layer and when $i = 5$, $\mathcal{P}^i$ will be $\mathcal{P}_{Up}(.)$ instead of $\mathcal{P}_{max}(\cdot)$.

Consequently, the decoder network $\mathcal{D}$ works in conjunction with the encoder to reconstruct and enhance the features. It also comprises four blocks, interconnected by an Upsampling layer $\mathcal{P}_{up}(.)$ or a Deconvolutional layer $\text{DC}(\cdot)$, along with batch normalization and ReLU activation. Notably, the output of the preceding decoder block and the corresponding feature map $e_i = \mathcal{E}_i$ from the encoder block are concatenated before being fed into the subsequent decoder block. This facilitates the seamless flow of information across the network, enhancing its ability to capture and retain essential features of the reflective areas. Each decoder block can be formulated as follows:

$$\mathcal{D}_j = \mathcal{P}^j(\text{ReLU}(\text{BN}(\text{DC}(\mathcal{D}_{j-1} \otimes \mathcal{E}_{5-j})))), \quad j \in [1..4] \tag{4.5}$$

where $\mathcal{D}_0 = \mathcal{E}_5$, $\mathcal{P}^j$ is the Upsampling or Deconvolutional layer and when $j = 4$, $\mathcal{P}^j$ will be $\text{DC}(\cdot)$ instead of $\mathcal{P}_{Up}(\cdot)$.

The output of the decoder network is split into two tensors: the first tensor represents reflection mask M_{rf}, utilized for optimizing the reflection loss, while the second tensor contains the enhanced reflective features Y_e, which have been processed to capture and emphasize reflection-related information.

4.3.5 Loss Functions

We use the softmax cross-entropy loss as our semantic loss $\mathcal{L}_s$ for supervising the semantic mask prediction and the ground truth semantic mask. Our loss for semantic mask prediction is:

$$\mathcal{L}_s = \text{ce}(\hat{M}, M_{GT}) \tag{4.6}$$

where $\text{ce}(.)$ is the softmax cross-entropy loss.

To supervise reflection, we also use the softmax cross-entropy loss for our reflection loss $\mathcal{L}_r$. However, there is no ground truth for the reflection mask, and we assume pseudo ground truth for the reflection mask to span common categories with reflective appearance such as window, door, cup, bottle, *etc*. Therefore, we extract pseudo ground truth with the reflective appearance in the ground truth semantic map M_{GT}. Note that as our pseudo ground truth might contain opaque appearances, we empirically found that such noise is not severe enough to affect the performance of the RFE module negatively. The reflection loss is:

$$\mathcal{L}_r = \text{ce}(M_{rf}, \phi(M_{GT})) \tag{4.7}$$

where $\phi(.)$ is a function to extract pseudo ground truth with the reflective appearance in the ground truth semantic map M_{GT}.

The total loss for our training is:

$$\mathcal{L} = \alpha\mathcal{L}_s + \beta\mathcal{L}_b + \gamma\mathcal{L}_r \tag{4.8}$$

where α, β and γ are hyper-parameters and are empirically set as [1.0,0.1,0.1] according to the experimental results.

4.4 Experiments

4.4.1 Datasets

We comprehensively evaluated our proposed method on diverse datasets to demonstrate its exceptional performance and versatility. These datasets encompass a broad spectrum of segmentation tasks such as Glass (Transparent) datasets (Trans10k-v2 [308], RGBP-Glass [188], and GSD-S [164]), Mirror (Reflection) datasets (MSD [324], PMD [163], and RGBD-Mirror [187]), and generic

Table 4.1: Comparison between different datasets in our experiments. "P", "D", and "S" mean polarization images, depth images, and semantic maps, respectively. Note that our method **uses only RGB images** as input for both training and testing.

	Dataset	Modalities	No. of Images	Tasks	Types	FOV	Position
Glass	Trans10k-v2 [308]	RGB	10,428	semantic	both	both	random
	RGBP-Glass [188]	RGB-P	4,511	binary	transparent	far	random
	GSD-S [164]	RGB-S	3,009	binary	transparent	far	random
Mirror	MSD [324]	RGB	4,018	binary	reflective	both	center
	PMD [163]	RGB	6,461	binary	reflective	both	random
	RGBD-Mirror [187]	RGB-D	3,049	binary	reflective	both	center
Generic	TROSD [256]	RGB-D	11,060	semantic	both	near	center
	Stanford2D3D [3]	RGB-D	70,496	semantic	both	far	random

datasets, which consists of both glass and mirror objects (TROSD [256], and Stanford2D3D [3]), ranging from binary to semantic segmentation, with a particular focus on images featuring reflective, transparent, or both characteristics. Our evaluation also considers the varied positions and fields of view (FOV) of objects within the images. Objects of interest may appear near or far from the camera's perspective, positioned randomly or at the center of the frame, providing a rich and realistic testing environment. Furthermore, the datasets we utilized are substantial in size, ensuring coverage of a broad range of environmental and scenario complexities. This encompasses indoor and outdoor scenarios, as well as varying lighting conditions, diverse object scales, different viewpoints, and levels of occlusion. Our extensive evaluation showcases the robustness and adaptability of our method across a wide array of real-world conditions. The detail of each dataset is shown in Table 4.1.

4.4.2 Implementation Details

We implemented our method in PyTorch 1.8.0 and CUDA 11.2. We adopted AdamW optimizer [177] where the learning rate γ was set to 10^{-4} with epsilon 10^{-8} and weight decay 10^{-4}. Our model was trained with a batch size of 8 and on a single NVIDIA RTX 3090 GPU, but it still can be trained on an older 2080 Ti or 1080 Ti GPU with a smaller batch size, e.g., 4. We evaluated all the variants of our network on the validation set for every epoch during the training process. We used the best model of each variant on the validation set to evaluate the variant on the test set. The training process was completed once there were no further improvements achieved. We use

	Output Size	Layer Name	PVT-Tiny	PVT-Small	PVT-Medium	PVT-Large
Stage 1	$\frac{H}{4} \times \frac{W}{4}$	Patch Embedding	$P_1 = 4;\ C_1 = 64$			
		Transformer Encoder	$\begin{bmatrix} R_1 = 8 \\ N_1 = 1 \\ E_1 = 8 \end{bmatrix} \times 2$	$\begin{bmatrix} R_1 = 8 \\ N_1 = 1 \\ E_1 = 8 \end{bmatrix} \times 3$	$\begin{bmatrix} R_1 = 8 \\ N_1 = 1 \\ E_1 = 8 \end{bmatrix} \times 3$	$\begin{bmatrix} R_1 = 8 \\ N_1 = 1 \\ E_1 = 8 \end{bmatrix} \times 3$
Stage 2	$\frac{H}{8} \times \frac{W}{8}$	Patch Embedding	$P_2 = 2;\ C_2 = 128$			
		Transformer Encoder	$\begin{bmatrix} R_2 = 4 \\ N_2 = 2 \\ E_2 = 8 \end{bmatrix} \times 2$	$\begin{bmatrix} R_2 = 4 \\ N_2 = 2 \\ E_2 = 8 \end{bmatrix} \times 3$	$\begin{bmatrix} R_2 = 4 \\ N_2 = 2 \\ E_2 = 8 \end{bmatrix} \times 3$	$\begin{bmatrix} R_2 = 4 \\ N_2 = 2 \\ E_2 = 8 \end{bmatrix} \times 8$
Stage 3	$\frac{H}{16} \times \frac{W}{16}$	Patch Embedding	$P_3 = 2;\ C_3 = 320$			
		Transformer Encoder	$\begin{bmatrix} R_3 = 2 \\ N_3 = 5 \\ E_3 = 4 \end{bmatrix} \times 2$	$\begin{bmatrix} R_3 = 2 \\ N_3 = 5 \\ E_3 = 4 \end{bmatrix} \times 6$	$\begin{bmatrix} R_3 = 2 \\ N_3 = 5 \\ E_3 = 4 \end{bmatrix} \times 18$	$\begin{bmatrix} R_3 = 2 \\ N_3 = 5 \\ E_3 = 4 \end{bmatrix} \times 27$
Stage 4	$\frac{H}{32} \times \frac{W}{32}$	Patch Embedding	$P_4 = 2;\ C_4 = 512$			
		Transformer Encoder	$\begin{bmatrix} R_4 = 1 \\ N_4 = 8 \\ E_4 = 4 \end{bmatrix} \times 2$	$\begin{bmatrix} R_4 = 1 \\ N_4 = 8 \\ E_4 = 4 \end{bmatrix} \times 3$	$\begin{bmatrix} R_4 = 1 \\ N_4 = 8 \\ E_4 = 4 \end{bmatrix} \times 3$	$\begin{bmatrix} R_4 = 1 \\ N_4 = 8 \\ E_4 = 4 \end{bmatrix} \times 3$

Table 4.2: Detailed settings of PVTv1 series which is adopted from [289].

mean Intersection over Union (mIoU) as the main evaluation metric to measure the segmentation performance.

The hyper-parameters of backbones in our models are listed as follows:

- S_i: stride of overlapping patch embedding in Stage i;
- C_i: channel number of output of Stage i;
- L_i: number of encoder layers in Stage i;
- R_i: reduction ratio of SRA layer in Stage i;
- P_i: patch size of Stage i;
- N_i: head number of Efficient Self-Attention in Stage i;
- E_i: expansion ratio of Feed-Forward layer [277] in Stage i;

In addition, we describe a series of PVTv1 [289] backbones with different scales (Tiny, Small, Medium, and Large) in Table 4.2 and a series of PVTv2 [290] backbones with different scales (B1 to B5) in Table 4.3.

	Output Size	Layer Name	PVT-B1	PVT-B2	PVT-B3	PVT-B4	PVT-B5
Stage 1	$\frac{H}{4} \times \frac{W}{4}$	Overlapping Patch Embedding	$S_1 = 4$				
			$C_1 = 64$				
		Transformer Encoder	$R_1 = 8$ $N_1 = 1$ $E_1 = 8$ $L_1 = 2$	$R_1 = 8$ $N_1 = 1$ $E_1 = 8$ $L_1 = 3$	$R_1 = 8$ $N_1 = 1$ $E_1 = 8$ $L_1 = 3$	$R_1 = 8$ $N_1 = 1$ $E_1 = 8$ $L_1 = 3$	$R_1 = 8$ $N_1 = 1$ $E_1 = 4$ $L_1 = 3$
Stage 2	$\frac{H}{8} \times \frac{W}{8}$	Overlapping Patch Embedding	$S_2 = 2$				
			$C_2 = 128$				
		Transformer Encoder	$R_2 = 4$ $N_2 = 2$ $E_2 = 8$ $L_2 = 2$	$R_2 = 4$ $N_2 = 2$ $E_2 = 8$ $L_2 = 3$	$R_2 = 4$ $N_2 = 2$ $E_2 = 8$ $L_2 = 3$	$R_2 = 4$ $N_2 = 2$ $E_2 = 8$ $L_2 = 8$	$R_2 = 4$ $N_2 = 2$ $E_2 = 4$ $L_2 = 6$
Stage 3	$\frac{H}{16} \times \frac{W}{16}$	Overlapping Patch Embedding	$S_3 = 2$				
			$C_3 = 320$				
		Transformer Encoder	$R_3 = 2$ $N_3 = 5$ $E_3 = 4$ $L_3 = 2$	$R_3 = 2$ $N_3 = 5$ $E_3 = 4$ $L_3 = 6$	$R_3 = 2$ $N_3 = 5$ $E_3 = 4$ $L_3 = 18$	$R_3 = 2$ $N_3 = 5$ $E_3 = 4$ $L_3 = 27$	$R_3 = 2$ $N_3 = 5$ $E_3 = 4$ $L_3 = 40$
Stage 4	$\frac{H}{32} \times \frac{W}{32}$	Overlapping Patch Embedding	$S_4 = 2$				
			$C_4 = 512$				
		Transformer Encoder	$R_4 = 1$ $N_4 = 8$ $E_4 = 4$ $L_4 = 2$	$R_4 = 1$ $N_4 = 8$ $E_4 = 4$ $L_4 = 3$	$R_4 = 1$ $N_4 = 8$ $E_4 = 4$ $L_4 = 3$	$R_4 = 1$ $N_4 = 8$ $E_4 = 4$ $L_4 = 3$	$R_4 = 1$ $N_4 = 8$ $E_4 = 4$ $L_4 = 3$

Table 4.3: Detailed settings of PVTv2 series which is adopted from [290].

4.4.3 Evaluation metrics.

For our evaluation, we adopt four widely used metrics from [188] for quantitatively assessing the glass segmentation performance: mean intersection over union (mIoU), weighted F-measure (F_β^w), mean absolute error (MAE), and balance error rate (BER).

Intersection over Union (IoU) is a widely used metric in segmentation tasks, which is defined as:

$$\mathrm{IoU} = \frac{\sum_{i=1}^{H}\sum_{j=1}^{W}(G(i,j) * P_b(i,j))}{\sum_{i=1}^{H}\sum_{j=1}^{W}(G(i,j) + P_b(i,j) - G(i,j) * P_b(i,j))} \tag{4.9}$$

where G is the ground truth mask in which the values of the glass region are 1 while those of the non-glass region are 0; P_b is the predicted mask binarized with a threshold of 0.5; and H and W are the height and width of the ground truth mask, respectively.

Weighted F-measure (F_β^w) is adopted from the salient object detection tasks with $\beta = 0.3$. F-measure (F_β) is a measure of both the precision and recall of the prediction map. Recent studies [61] have suggested that the weighted F-measure (F_β^w) [184] can provide more reliable evaluation results than the traditional F_β. Thus, we report F_β^w in the comparison.

Mean Absolute Error (MAE) is widely used in foreground-background segmentation tasks, which calculates the element-wise difference between the prediction map P and the ground truth mask G :

$$MAE = \frac{1}{H \times W} \sum_{i=1}^{H} \sum_{j=1}^{W} |P(i,j) - G(i,j)|, \tag{4.10}$$

where $P(i,j)$ indicates the predicted probability score at location (i,j).

Balance Error Rate (BER) is a standard metric used in shadow detection tasks, defined as:

$$BER = (1 - \frac{1}{2}(\frac{TP}{N_p} + \frac{TN}{N_n})) \times 100 \tag{4.11}$$

where TP, TN, N_p, and N_n represent the numbers of true positive pixels, true negative pixels, glass pixels, and non-glass pixels, respectively.

4.4.4 Qualitative and Quantitative Results

We evaluated the performance of our method across three distinct tasks: glass segmentation, mirror segmentation, and generic segmentation. To ensure **fair comparisons**, we have carefully selected our model variants (Ours-X with X is postfixes: -T, -S, -M, -L, -B1, -B2, -B3, -B4, and -B5, represented the size of the model as PVTv1 Tiny, Small, Medium, Large, and PVTv2 B1-5, respectively) that have **similar model's size or complexity** used by other methods, as indicated in the respective tables. The experimental results show that our method consistently outperforms other state-of-the-art methods across all datasets as shown in Figure 4.5.

Comparison on Glass Object Segmentation

We benchmarked our method against recent glass segmentation methods on binary (RGBP-Glass and GSD-S dataset) and semantic segmentation (Trans10K-v2 dataset) tasks.

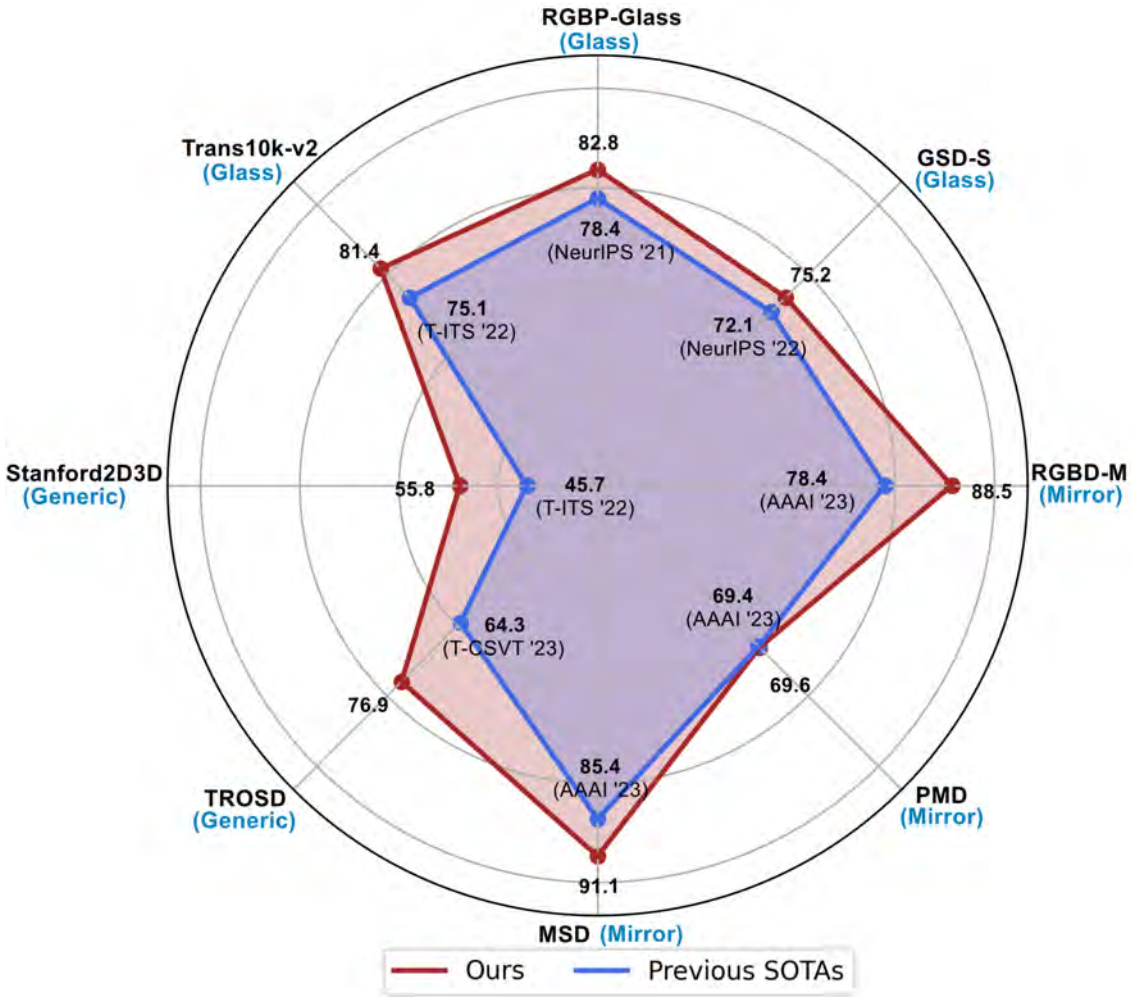


Figure 4.5: Quantitative performance of our proposed method with previous SOTAs on different datasets. Our method achieves competitive performance over previous methods on both glass, mirror, and generic segmentation tasks. To maintain fairness, we only compare with methods that using the same input (using only RGB image).

RGBP-Glass dataset. We extensively compare the effectiveness of our method with state-of-the-art methods, as shown in Table 4.4. All methods are retrained on RGBP-Glass dataset [188] for a fair comparison. EAFNet [304], Polarized Mask R-CNN (P.M. R-CNN) [121], and PGSNet [188] are the three methods that leverage polarization cues. SETR [348], SegFormer [306] are the two methods focusing on general semantic/instance segmentation tasks. GDNet [189], TransLab [307], Trans2Seg [308], and GSD [162] and our method are in-the-wild glass segmentation methods but only rely on RGB input. From Figure 4.6, we can see that our method outperforms all other methods. It should be noted that our method even outperforms previous works that utilize additional input signals such as polarization cues [304, 121, 188] while being efficient.

Table 4.4: Quantitative comparison against state-of-the-art on RGB-P dataset [188]. **Note that:** we only use RGB as input for our method, and the results are sorted by ascending of GFLOPS. (∗) denotes the glass segmentation methods with additional polarization images as input.

Method	**Backbone**	**GFLOPs** ↓	**mIoU** ↑	F_β^w ↑	**MAE** ↓	**BER** ↓
EAFNet [304] ∗	ResNet-18	18.93	53.86	0.611	0.237	24.65
P.M. R-CNN [121] ∗	ResNet-101	56.59	66.03	0.714	0.178	18.92
PGSNet [188] ∗	Conformer-B	290.62	<u>81.08</u>	<u>0.842</u>	<u>0.091</u>	<u>9.63</u>
Trans2Seg [308]	ResNet-50	49.03	75.21	0.799	0.122	13.23
TransLab [307]	ResNet-50	61.26	73.59	0.772	0.148	15.73
SegFormer [306]	MiT-B5	70.24	78.42	0.815	0.121	13.03
GSD [162]	ResNeXt-101	92.69	78.11	0.806	0.122	12.61
Ours-B5	PVTv2-B5	154.37	**82.77**	**0.879**	**0.042**	**9.59**
PGSNet [188]	Conformer-B	-	76.11	0.797	0.126	13.08
GDNet [189]	ResNeXt-101	271.53	77.64	0.807	0.119	11.79
SETR [348]	ViT-Large	240.11	77.60	0.817	0.114	11.46

Figure 4.6: Qualitative comparison of our method with other methods on RGB-P dataset [188]. (∗) denotes the glass segmentation method with additional polarization images as input.

GSD-S dataset. We compare our method with other recent methods in Table 4.5 and Figure 4.7, includes generic semantic segmentation methods (PSPNet [344], DeepLabV3+ [24], PSANet [345], DANet [73]), recent state-of-the-art models that utilize transformer technique (SETR [348], Swin [172], SegFormer [306], Twins [39]), and glass surface detection methods (GDNet [189], GSD [162], GlassSemNet [164]). For a fair comparison, all methods are retrained on the GSD-S dataset [164]. Our method outperforms all other methods and achieves comparative performance with GlassSemNet [164], which has additional semantic context information. GlassSemNet [164] points out that humans frequently use the semantic context of their surroundings

Table 4.5: Evaluation results on GSD-S dataset [164]. **Note that:** we only use RGB as input for our method. (†) denotes the glass segmentation method with additional semantic context information and post-processing refinement.

Method	**mIoU** ↑	F_β^w ↑	**MAE** ↓	**BER** ↓
PSPNet [344]	56.1	0.679	0.093	13.41
DeepLabV3+ [24]	55.7	0.671	0.100	13.11
PSANet [345]	55.1	0.656	0.104	12.61
DANet [73]	54.3	0.673	0.098	14.78
SCA-SOD [245]	55.8	0.689	0.087	15.03
SETR [348]	56.7	0.679	0.086	13.25
Segmenter [251]	53.6	0.645	0.101	14.02
Swin [172]	59.6	0.702	0.082	11.34
Tokens-to-Token ViT [330]	56.2	0.693	0.087	14.72
SegFormer [306]	54.7	0.683	0.094	15.15
Twins [39]	59.1	0.703	0.084	12.43
GDNet [189]	52.9	0.642	0.101	18.17
GSD [162]	72.1	0.821	0.061	10.02
Ours-B5	<u>75.2</u>	<u>0.859</u>	<u>0.046</u>	**9.04**
GlassSemNet [164] †	**75.3**	**0.860**	**0.035**	<u>9.26</u>

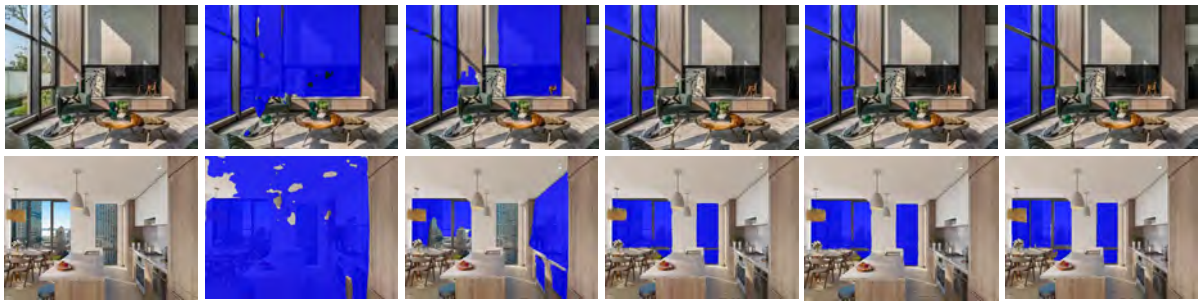


Figure 4.7: Qualitative comparison of our method with other methods on GSD-S dataset [164]. (†) denotes the glass segmentation method with semantic context and post-processing refinement.

to reason, as this provides information about the types of things to be found and how close they might be to one another. For instance, glass windows are more likely to be found close to other semantically related objects (walls and curtains) than to things (cars and trees). So, their method utilizes semantic context information as additional input to progressively learn the contextual correlations among objects spatially and semantically, boosting their performance. Then, their predictions are refined by

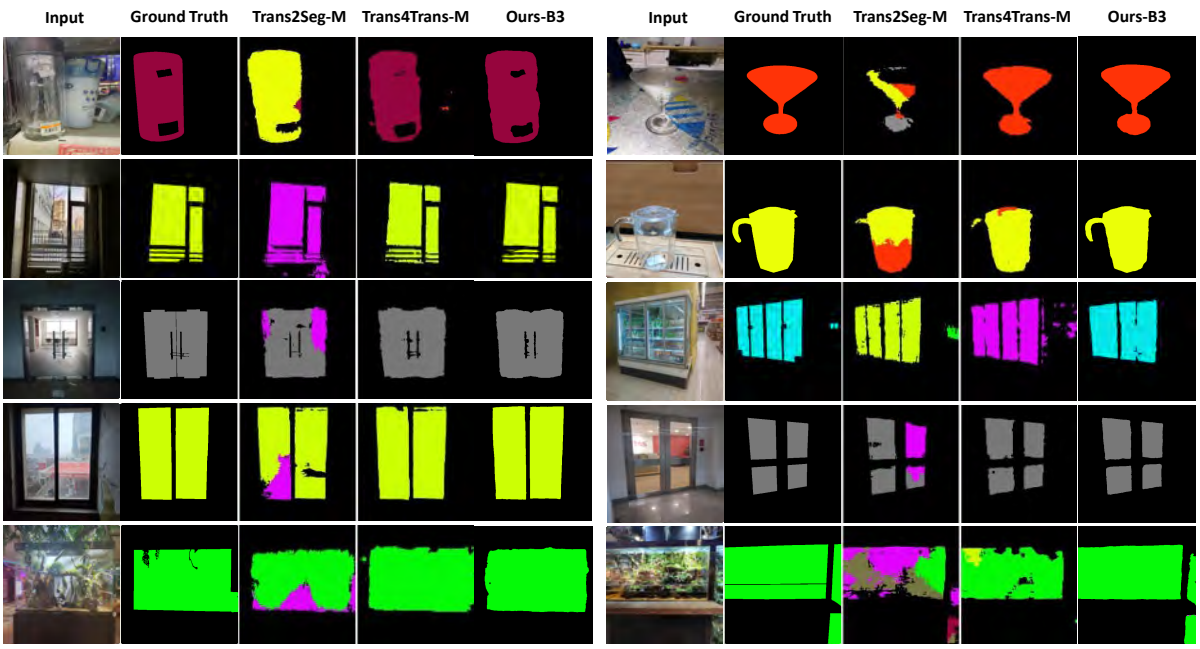


Figure 4.8: Qualitative comparison of our method and existing methods on Trans10K-v2 [308]. For a fair comparison, we used Ours-B3, which has the same network size as other methods (-M model).

Fully Connected Conditional Random Fields (CRF) [136] to improve their performance further.

Trans10k-v2 dataset. Shifting our focus to the semantic glass segmentation task, where the challenge extends beyond merely detecting glass areas to classifying them into 11 fine-grained categories, our method still reigns supreme, as shown in Table 4.6. The results in Figure 4.8 also confirm that our method achieves higher segmentation quality with better transparent features, e.g., segmentation of two overlapping doors is accurately obtained. These comprehensive evaluations underscore the effectiveness of our approach across diverse glass segmentation scenarios, affirming its position as a top-performing and computationally efficient choice for these tasks.

Comparison on Binary Mirror Segmentation

MSD and PMD datasets. We compare quantitative results of the state-of-the-art methods and our method on MSD and PMD datasets, including four RGB salient object detection methods CPDNet [302], MINet [206], LDF [296], and VST [169], and five mirror detection methods MirrorNet [324], PMDNet [163], SANet [84], VCNet [258], SATNet [106]. As shown in Table 4.7, our method achieves the best performance in terms of all the evaluation metrics. Significantly, we outperform the second-best method by 5.63% on the MSD dataset.

Table 4.6: Quantitative evaluation of our method and existing methods on the Trans10K-v2 dataset [308]. **Note that:** the results are sorting by ascending of GFLOPS.

Method	GFLOPs ↓	MParams ↓	ACC ↑	mIoU ↑
DFANet [151]	1.02	-	85.15	42.54
HRNet_w18 [283]	4.20	1.53	89.58	54.25
HarDNet [20]	4.42	-	90.19	56.19
LEDNet [292]	6.23	-	86.07	46.40
Trans4Trans-T [337]	10.45	-	93.23	68.63
Ours-T	10.50	12.72	**93.52**	**69.53**
ICNet [343]	10.64	8.46	78.23	23.39
BiSeNet [326]	19.91	13.3	89.13	58.40
Trans4Trans-S [337]	19.92	-	94.57	74.15
Ours-S	20.00	23.98	94.83	75.32
Ours-B1	21.29	14.87	**95.37**	**77.05**
Trans4Trans-M [337]	34.38	-	95.01	75.14
Ours-M	34.51	43.70	95.08	76.06
DenseASPP [323]	36.20	29.09	90.86	63.01
Ours-B2	37.03	27.59	**95.92**	**79.29**
DeepLabv3+ [24]	37.98	28.74	92.75	68.87
FCN [174]	42.23	34.99	91.65	62.75
OCNet [331]	43.31	-	92.03	66.31
RefineNet [161]	44.56	29.36	87.99	58.18
Trans2Seg [308]	49.03	56.20	94.14	72.15
Ours-L	50.54	60.86	95.28	77.35
TransLab [307]	61.31	42.19	92.67	69.00
Ours-B3	68.35	51.21	96.28	80.04
Ours-B4	79.34	67.11	**96.59**	**80.99**
U-Net [232]	124.55	13.39	81.90	29.23
DUNet [119]	123.69	-	90.67	59.01
Ours-B5	154.37	106.19	**96.93**	**81.37**
DANet [73]	198.00	-	92.70	68.81
PSPNet [344]	187.03	50.99	92.47	68.23

RGBD-Mirror dataset. Our method is also compared with seven RGB-D salient object detection methods such as HDFNet [205], S2MA [168], JL-DCF [74], DANet [73], BBSNet [68] and VST [169], and four mirror detection methods, including PDNet [187], SANet [84], VCNet [258], and PDNet [187] on the RGBD-Mirror dataset. Our method outperforms all the competing methods, even though we do not use depth information, which is shown in Table 4.8. We show the

Table 4.7: Quantitative results of our method with the state-of-the-art methods on Salient Object Detection and Mirror Detection on MSD and PMD datasets.

Method		MSD			PMD		
		IoU ↑	F_β^w ↑	**MAE** ↓	**IoU** ↑	F_β^w ↑	**MAE** ↓
Salient Object Detection	CPDNet [302]	57.58	0.743	0.115	60.04	0.733	0.041
	MINet [206]	66.39	0.823	0.087	60.83	0.798	0.037
	LDF [296]	72.88	0.843	0.068	63.31	0.796	0.037
	VST [169]	79.09	0.867	0.052	59.06	0.769	0.035
Mirror Detection	MirrorNet [324]	78.88	0.856	0.066	58.51	0.741	0.043
	PMDNet [163]	81.54	0.892	0.047	66.05	0.792	0.032
	SANet [84]	79.85	0.879	0.054	66.84	0.837	0.032
	VCNet [258]	80.08	0.898	0.044	64.02	0.815	0.028
	SATNet [106]	85.41	0.922	0.033	69.38	0.847	0.025
	Ours-B3	**91.04**	**0.953**	**0.028**	**69.61**	**0.853**	**0.021**

Table 4.8: Quantitative results of the state-of-the-art methods on RGBD-Mirror dataset.

Method		Input	IoU ↑	F_{fi}^w ↑	MAE ↓
Salient Object Detection	HDFNet [205]	RGB-D	44.73	0.733	0.093
	S2MA [168]	RGB-D	60.87	0.781	0.070
	DANet [73]	RGB-D	67.81	0.835	0.060
	JL-DCF [74]	RGB-D	69.65	0.844	0.056
	VST [169]	RGB-D	70.20	0.851	0.052
	BBSNet [68]	RGB-D	74.33	0.868	0.046
	PDNet [187]	RGB-D	77.77	0.878	0.041
Mirror Detection	VCNet [258]	RGB	73.01	0.849	0.052
	PDNet [187]	RGB	73.57	0.851	0.053
	SANet [84]	RGB	74.99	0.873	0.048
	SATNet [106]	RGB	78.42	0.906	0.031
	Ours-B3	RGB	**88.52**	**0.954**	**0.027**

visualization of all three mirror datasets in Figure 4.9.

Comparison on Generic Segmentation

Stanford2D3D dataset. As shown in Table 4.9, we show comparisons with other methods in different sizes of backbones. Our method outperforms other existing works by about 10.1% better

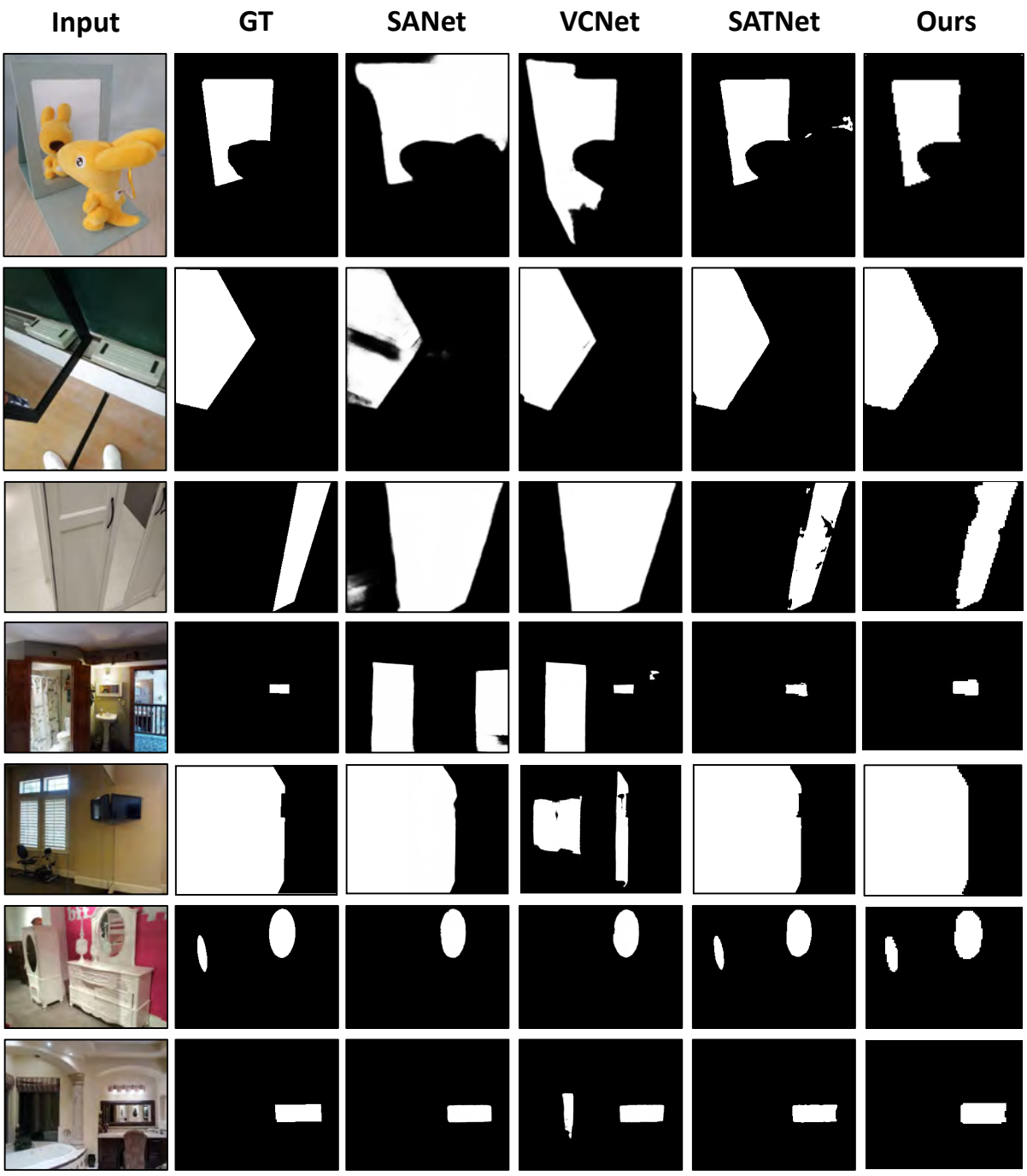


Figure 4.9: Qualitative comparison of our method with other methods on MSD, PMD, and RGBD-Mirror datasets.

performance in mIOU, which highlights the segmentation capacity of our network on the general scene where there is a presence of glass objects.

Table 4.9: Comparison with SOTAs on Stanford2D3D dataset. **Note that:** the results are sorting by ascending of GFLOPS.

Method	GFLOPs ↓	MParams ↓	mIoU ↑
PVT-T [289]	10.16	13.11	41.00
Trans4Trans-T [337]	10.45	12.71	41.28
Ours-T	10.50	12.72	47.11
Trans2Seg-T [308]	16.96	17.87	42.07
Ours-B1	21.99	14.87	**51.55**
PVT-S [289]	19.58	24.36	41.89
Trans4Trans-S [337]	19.92	23.95	44.47
Ours-S	20.00	23.98	50.17
Trans2Seg-S [308]	30.26	27.98	42.91
Ours-B2	37.03	27.59	**53.98**
Trans4Trans-M [337]	34.38	43.65	45.73
Ours-M	34.51	43.70	52.57
Trans2Seg-M [308]	40.98	30.53	43.83
PVT-M [289]	49.00	56.20	42.49
Ours-B3	68.35	51.21	**54.66**
Ours-L	50.54	60.86	53.75
Ours-B4	79.34	67.11	**55.21**
Ours-B5	154.37	106.19	**55.83**

TROSD dataset. We compared our method with SOTAs on the TROSD dataset [256] - a specific dataset for transparent and reflective objects. Table 4.10 provides an overview of our competitors and highlights their best results, achieved using their publicly available source codes. All methods utilized the same data augmentation strategy. The visualization is shown in Figure 4.10.

ADE20k and Cityscapes datasets. We conducted additional experiments on ADE20K and CityScapes datasets, with the results (mIoU) shown in Table 4.11 and sorted by ascending order of GFLOPs (512×512). As can be seen, our method performs well on both datasets, with mIoU 47.5% on ADE20K and 81.9% on CityScapes.

Failure Cases. Figure 4.11–left showcases failure cases of our method and other methods on Trans10K-v2. Our method would confuse and fail to segment the object with similar properties as others. In such a scenario, even human beings would struggle to differentiate between these transparent things. However, despite assigning the wrong label, our method can still maintain

Table 4.10: Performance comparison of different methods on TROSD. R: reflective objects. T: transparent objects. B: background.

Method	Input	IOU ↑			mIoU ↑	mAcc ↑
		R	T	B		
RefineNet [161]	RGB	21.32	37.32	92.37	50.34	63.59
ANNNet [360]	RGB	22.31	41.30	93.43	52.35	62.49
Trans4Trans [337]	RGB	27.69	39.22	94.16	53.69	61.82
PSPNet [344]	RGB	26.35	44.38	94.19	54.97	64.14
OCNet [331]	RGB	31.76	46.52	95.05	57.78	64.46
TransLab [307]	RGB	42.57	50.72	96.01	63.11	68.72
DANet [73]	RGB	42.76	54.39	95.88	64.34	70.95
TROSNet [256]	RGB	48.75	48.56	95.49	64.26	75.93
Ours	RGB	**66.16**	**66.83**	**97.71**	**76.90**	**87.62**
SSMA [276]	RGB-D	24.70	29.04	89.98	47.91	67.72
FRNet [358]	RGB-D	28.37	36.59	92.18	52.38	63.94
EMSANet [239]	RGB-D	27.53	44.10	96.14	55.92	71.63
FuseNet [88]	RGB-D	37.30	43.29	94.97	58.52	66.13
RedNet [118]	RGB-D	48.27	47.57	95.76	63.87	69.23
EBLNet [90]	RGB-D	51.75	50.12	94.57	65.49	67.39
TROSNet [256]	RGB-D	62.27	57.23	96.52	72.01	81.21

Table 4.11: Comparison (mIoU ↑) with SOTAs on ADE20k and Cityscapes datasets.

Method	GFLOPs↓	MParams↓	Backbone	ADE20K	CityScapes
Trans4Trans-M [290]	41.9	49.6	PVTv2-B3	-	69.3
Semantic FPN [337]	62.4	49.0	PVTv2-B3	47.3	-
Ours-B3	68.3	51.2	PVTv2-B3	47.5	81.9
MogaNet-S [155]	189	29.0	SemFPN	47.7	-
NAT-Mini [87]	900	50.0	UPerNet	46.4	-
InternImage-T [288]	944	59.0	UPerNet	**47.9**	**82.5**

the object's shape. We also show several failure instances (Figure 4.11–right) in our system that misinterpret non-glass areas as glass because they seem and behave the same, such as still in the door frame with reflection and distortion.

4.4.5 Ablation studies

In this section, we present ablation studies to verify different aspects of the design of our model and underscore the significance of each module within our model. Any alterations or omissions to the

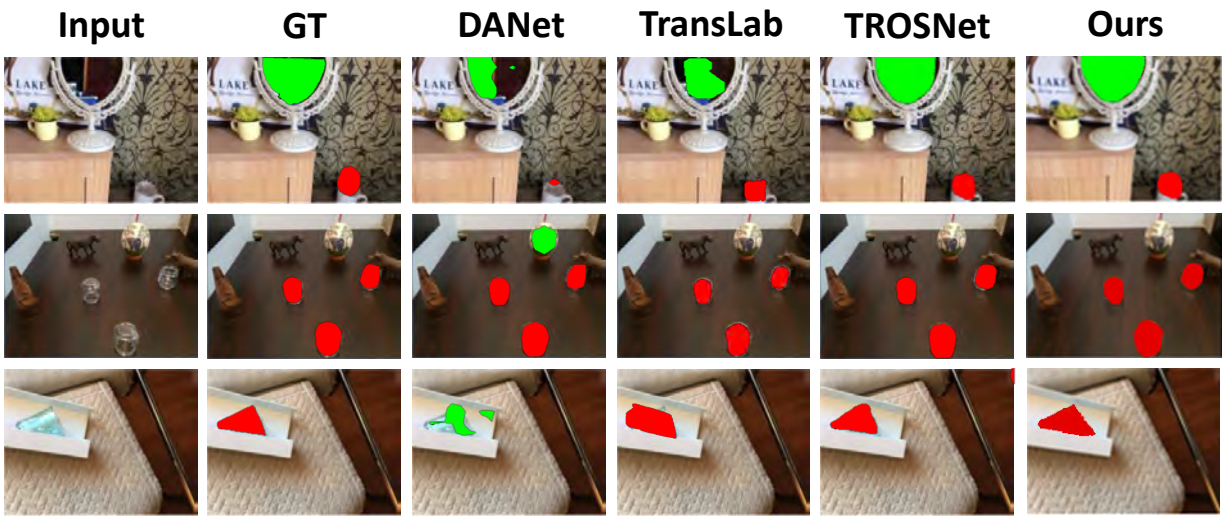


Figure 4.10: Qualitative comparison of our method with other methods on the TROSD dataset.

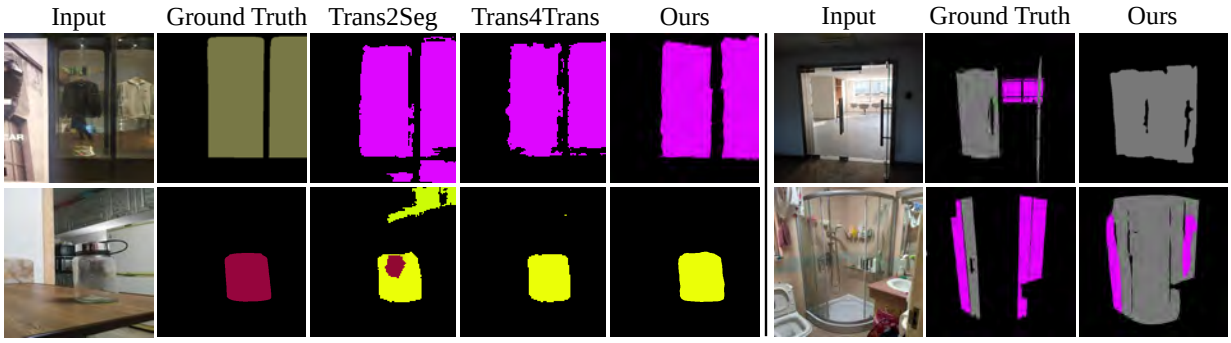


Figure 4.11: Failure cases of our method and existing methods on the Trans10K-v2 dataset.

proposed design led to noticeable drops in performance, which justifies our choice of transformer architecture and the boundary and reflection feature learning components.

Different combinations of network architecture. Table 4.12 presents the comparisons between various combinations of encoders and decoders, such as using only CNN architecture, using a combination of both CNN and Transformer, and using a fully transformer-based model. Our method, an encoder-decoder transformer-based model, outperforms other competitive networks, indicating the system's capability for effectively segmenting transparent objects. In this ablation study, we used Ours-M and Ours-B2 (not the best model Ours-B5), which has the same network size as other methods (-M model size), for a fair comparison.

Analysis of different backbones. We have conducted experiments using alternative backbones, as presented in Figure 4.12. Among these options, the PVT-v2 backbone [290] stands out with

Table 4.12: Effectiveness of different network architecture combinations. Models are evaluated on the Trans10K-v2 dataset [308]. **Note that:** the results are sorting by ascending of GFLOPS.

Method	Encoder		Decoder		GFLOPs	mIoU
	Transformer	CNN	Transformer	CNN		
Trans4Trans-M [337]	✓		✓		34.3	75.1
Ours-M	✓		✓		34.5	76.1
Ours-B2	✓		✓		37.0	**79.3**
Trans2Seg-M [308]		✓	✓		40.9	69.2
FCN [174]		✓		✓	42.2	62.7
OCNet [331]		✓		✓	43.3	66.3
PVT-M [289]	✓		✓		49.0	72.1

significantly higher mIoU and remarkably compact model size (MParams). Despite its higher complexity in terms of GFLOPs compared to the FocalNet backbone [320], it still manages to achieve better performance. Additionally, the PVT-v2 backbone [290] demonstrates a lower complexity than the DaViT backbone [49] while maintaining competitive mIoU results. These findings highlight the superiority of the PVT-v2 backbone [290] in achieving an optimal balance between performance and model size, making it a promising choice for our method. When comparing PVT-v1 [289] with other backbones, the PVT-v1 [289] backbone boasts a considerably smaller model size and lower complexity. Despite these advantages, its performance remains competitive and comparable to the other backbones. This demonstrates the efficiency of the PVT-v1 backbone [289], as it manages to deliver comparable performance while being more lightweight and less computationally demanding.

Effectiveness of different modules. To assess the contribution of both the proposed BFE and RFE modules to our architecture's performance, we systematically evaluated the model under various configurations: ❶ **Baseline Model** (PVTv1-T or PVTv2-B1 without BFE and RFE): this served as our control group, where both the BFE and RFE modules were excluded. Results indicate a foundational performance that the other configurations could be compared against. ❷ **Incorporation of BFE:** When only the BFE module was integrated into our network, we noticed a significant performance enhancement. However, this performance did not reach the potential of the combined BFE and RFE configuration. This proved that while BFE is essential, it works best in tandem with RFE. ❸ **Incorporation of RFE:** Similarly, adding only the RFE module to the baseline network also increased performance. This emphasized the value of detecting reflections in transparent objects for the segmentation task. ❹ **Combined Integration of BFE and RFE:** both modules were simultaneously integrated into our network. The performance gain observed in this configuration, as

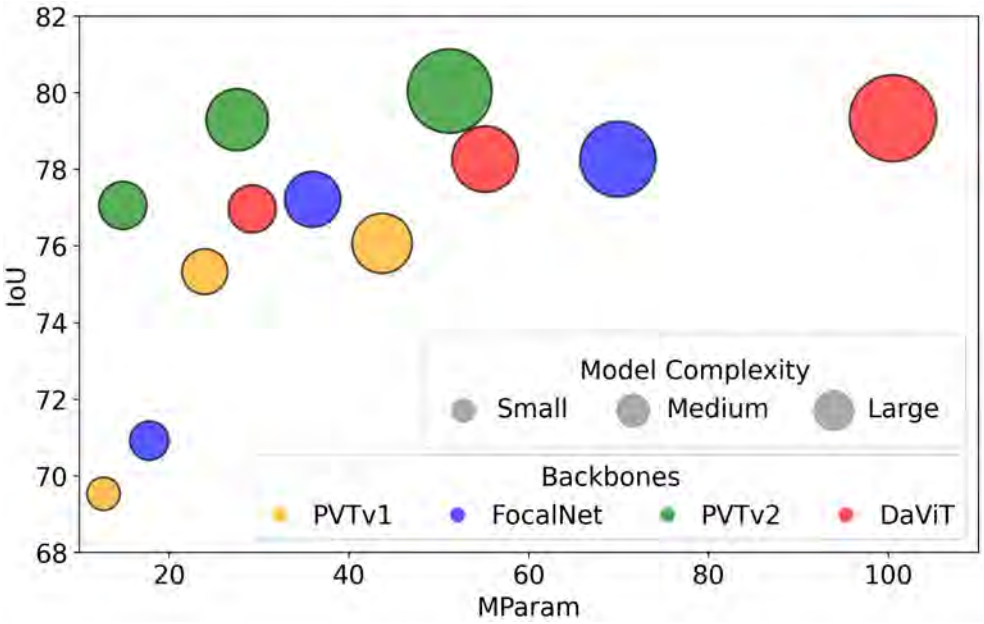


Figure 4.12: Our method with various backbones on Trans10K-v2 dataset. The bubble's size is its complexity in GFLOPs.

Table 4.13: Effectiveness of different modules of our method on Trans10K-v2 dataset [308] and Stanford2D3D dataset [3]. We reported mIoU(%) as a metric in this study. The last row corresponds to our method (Ours-B1).

Backbone	**FLOPs**	**Params**	**BFE**	**RFE**	**Stanford2D3D**	**Trans10K**
PVTv1-T	10.16	13.11	-	-	45.19	69.44
PVTv2-B1	11.48	13.89	-	-	46.79 +1.6	70.49 +1.05
PVTv2-B1	13.22	14.37	-	✓	48.12 +2.93	72.65 +3.21
PVTv2-B1	19.55	14.39	✓	-	50.22 +5.03	74.89 +5.45
PVTv2-B1	21.29	14.87	✓	✓	**51.55** +6.36	**77.05** +7.61

shown in Table 4.13, was the most pronounced, with gains of **6.36%** and **7.61%** in mIoU on the Trans10K-v2 and Stanford2D3D datasets, respectively. This confirms that the combined effects of boundary and reflection cues significantly augment the network's segmentation capabilities.

Interestingly, the ablation studies further explain why our method works well for generic segmentation as in the Stanford2D3D dataset. It can be seen in Table 4.13 that the boundary module yields the most significant performance gain compared to the reflection module. This means that for generic segmentation, where reflection feature learning has negligible improvement, our boundary feature learning remains effective for general semantic labels.

Table 4.14: Different designs with the proposed modules on Trans10k-v2 dataset. X $\rightarrow$ Y means placing X before Y in Figure 4.2; X // Y means placing them in parallel and concatenate their outputs in Figure 4.2.

Variants	**MParams** $\downarrow$	**mIoU** $\uparrow$
Baseline	13.89	70.49
+ RFE $\rightarrow$ BFE in Encoder	48.98	73.54
+ BFE $\rightarrow$ RFE in Encoder	48.99	74.12
+ RFE $\rightarrow$ BFE in Decoder	14.90	75.11
+ BFE // RFE in Decoder	14.91	75.44
TransCues (BFE $\rightarrow$ RFE in Decoder)	14.87	**77.05**

Placement of modules. It is not trivial to incorporate both visual cues into the same framework, as certain features may best be captured at certain stages. In our framework, we find the related features better captured at the end of the Decoder layers, with BFE following RFE module. Table 4.14 shows our results that the aforementioned modules' position and order of processing matters.

4.4.6 Further Analysis and Discussions

Effectiveness of the embedding channel. We experiment with the embedding channel with various values (64, 128, 256, 512) and report the mIoU and Accuracy of Ours-B1 model in Figure 4.13. Throughout the results, we proved that our model achieved better performance with a higher embedding channel (from 77.05% at 64 channels to 78.85% at 512 channels). Note that, due to memory limits, we can not perform experiments with higher embedding channels, *e.g.*, 1024, 2048, and to save computational resources, we used Ours-B1 in this ablation study.

Real-time performance. We calculate the inference speed of our models on different GPUs (NVIDIA GTX 1070, NVIDIA RTX 3090) with the resolution of 512 $\times$ 512 and batch size of 1. As shown in Figure 4.14, while Our-T model has lower computational cost than other versions, it's important to note that all these models deliver performance levels well-suited for deployment on robotic systems. In real-world situations, reaching a similar level of prediction accuracy on each frame is crucial because it makes it possible for a navigation system to be more responsive, improving the system's capacity to aid robots efficiently.

Incorporation with other modalities. Integrating our model with depth images (RGB-D) or polarization images (RGB-P) is a feasible enhancement. A naive method involves adding an extra

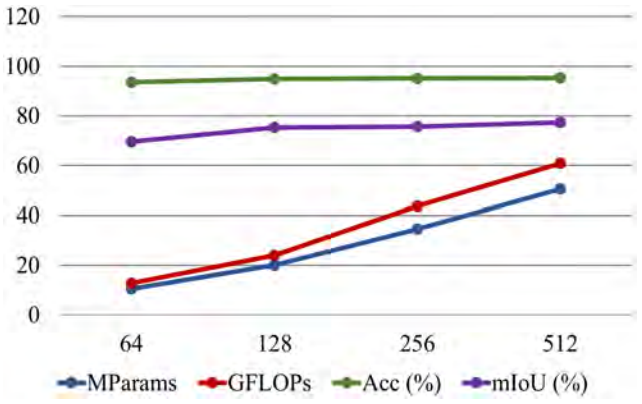


Figure 4.13: Effectiveness of the embedding channel in our method on Trans10K-v2 [308].

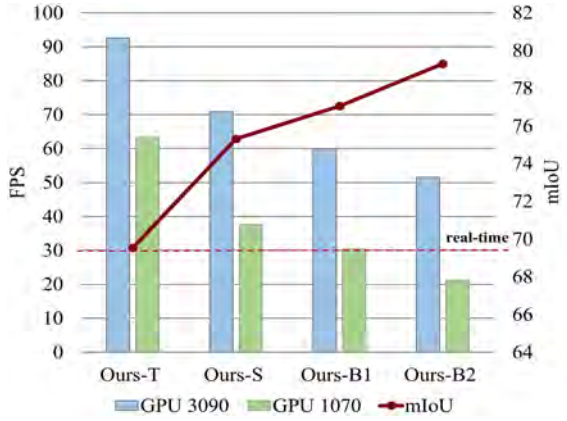


Figure 4.14: Inference time (FPS) of our proposed methods on Trans10K-v2.

encoder to extract features from depth or polarization data. These additional features would then be fused with RGB features prior to our FPM module. This strategy is in line with PDNet [187], TROSNet [256], and PGSNet [188], as detailed in the supplementary material. It is noteworthy that the inclusion of depth or polarization data in these models has led to significant performance enhancements, but also with additional computation costs. Specifically, with added depth information, PDNet and TROSNet improved by +4% and +8% mIoU, and with added polarization information, PGSNet experienced a +5% boost in mIoU.

Utilizing reflection removal methods for detecting reflections. Employing reflection removal techniques, as discussed in recent studies [53, 249], offers the potential to generate pseudo labels with distinct advantages. However, these methods are mainly designed to address global reflections when an image is entirely encompassed by glass. These methods have limitations regarding complex real-world situations where glass objects are distributed throughout the scene rather than occupying a dominant position. Our study introduces the RFE module, which can detect localized reflections and distinguish glass surfaces based on the semantic mask. This module is better suited for the diverse and unpredictable conditions found in real-world situations where reflections are specific to certain areas rather than uniformly distributed over the entire image, making it a better fit for real-world scenarios.

Comparison with foundation models. To fully evaluate our method's performance, we also compared our method with recent powerful foundation models such as SAM model, and the results are shown in Figure 4.15. It is important to note that the SAM model **does not include semantics, or in other words, it cannot yield masks with semantics**. The SAM model also presents the challenge

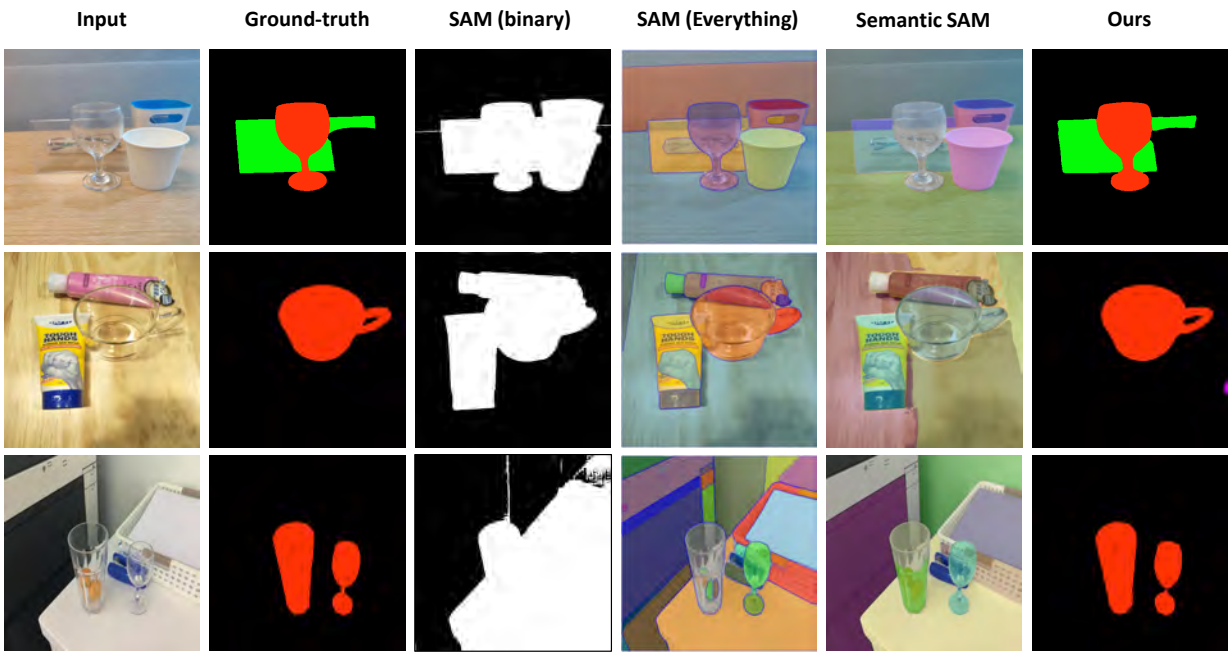


Figure 4.15: Qualitative comparison of our method with recent foundation models on Trans10k-v2 dataset.

of over-segmentation, thereby leading to a higher likelihood of false positives. As a result, we can see that the SAM model (binary and everything) can not distinguish between glass and non-glass regions compared to our method. It is the same for SAM variants with semantics [150, 22], which still fails and cannot generate reasonable semantics either.

Further analysis on our reflection RFE module. To provide a further verification of the effectiveness of the RFE module, we conducted the following additional experiments:

- We take a model with the RFE module that has already been trained. To prove that RFE is effective, we compare the feature maps before and after passing through the RFE module. Please see Figure 4.16 below. In our example, we can see that after passing through the RFE module, we can get a stronger reflection signal, such as the transparent glass area or the specular reflection appearing at the base of the wine glass.
- Using the same model, we try disabling the RFE module at inference by passing the feature map before RFE directly to the next step. Note that at training, the RFE module is well-trained as usual. Figure 4.17 shows that bypassing RFE results in a noisy feature map and wrong mask prediction. This means that our learning of RFE does not yield a trivial function, *e.g.* identity, and RFE does play an important role in processing the feature maps and output reflection

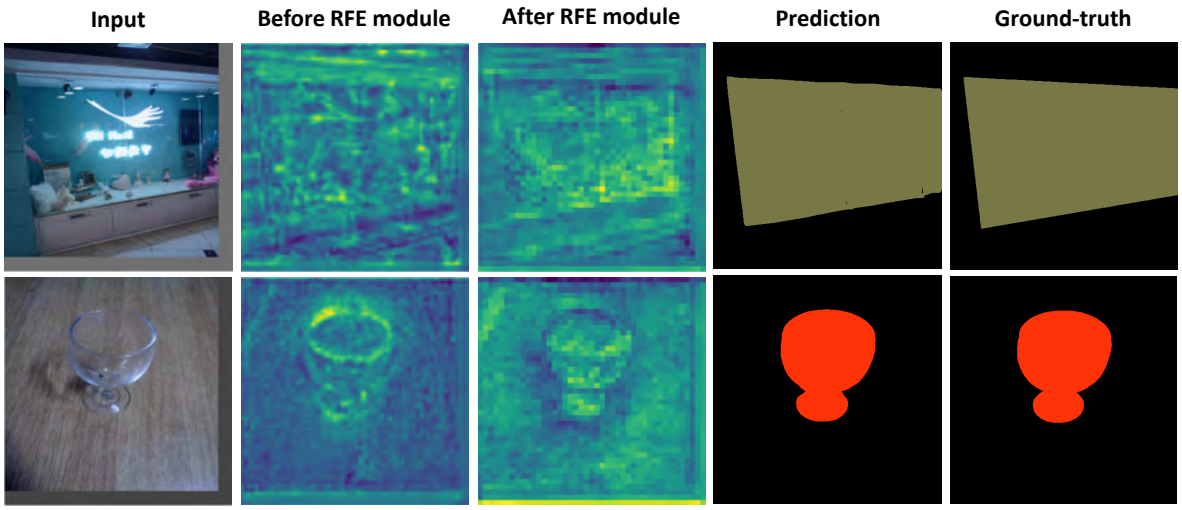


Figure 4.16: Comparison of the feature maps before and after passing through the RFE module on Trans10k-v2 dataset.

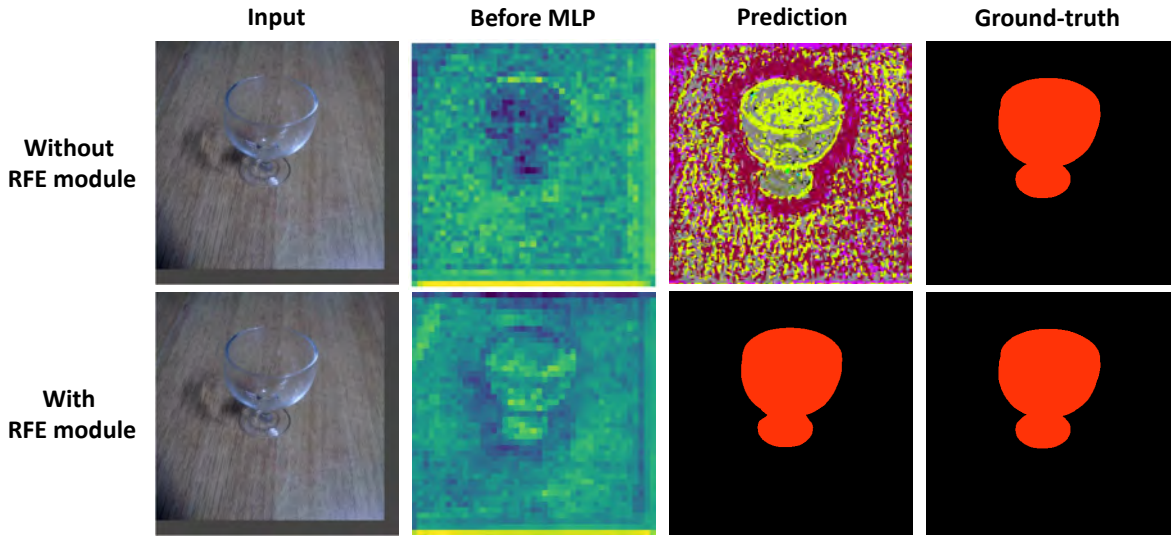


Figure 4.17: Comparison of the feature maps at inference by passing through the RFE module as usual (top row) and bypassing the RFE module (bottom row) on Trans10k-v2 dataset. Note that at training, the RFE module is well-trained as usual.

masks.

4.5 Conclusions

In conclusion, this work proposes a method to segment transparent and opaque objects along with general objects using pyramidal transformer architecture. Our method exploits two important visual

cues based on boundary and reflection features, significantly leading to performance gains in both transparent and generic segmentation tasks. We extensively evaluated our proposed method on several benchmark datasets, demonstrating its robustness in various scenarios.

Importantly, our framework is designed to be adaptable, accommodating a variety of well-known transformer backbones, as detailed in Figure 4.12. Additionally, our network can maintain its efficiency and deploy on compact and edge devices. as detailed in Figure 4.14.

Our architecture is a fully transformer-based method built upon the PVT. Therefore, some limitations still exist that lower our method's capabilities for visual tasks. Firstly, the position encoding of our network is fixed-size, requiring a resizing step, which will damage and distort the object's shape. Secondly, similar to other vision transformer-based methods, the computational cost of our network is relatively high when dealing with high-resolution images. Finally, as stated before, we use the same position embedding as ViT [56] and PVT [289], which is insufficient for arbitrary resolution of input images. In future work, we would like to investigate how to address the above limitations and improve failure cases. In addition, conventional segmentation approaches only consider static images because they are simple to capture and process. Therefore, we would like to investigate the extension of our method to other modalities, including depth images, event data, videos, and dynamic scenes.

CHAPTER 5

OPEN-VOCABULARY CAMOUFLAGED INSTANCE SEGMENTATION

5.1 Introduction

Camouflage is a powerful biological mechanism for avoiding detection and identification. In nature, camouflage tactics are employed to deceive the sensory and cognitive processes of both preys and predators. Wild animals utilise these tactics in various ways, ranging from blending themselves into the surrounding environment to employing disruptive patterns and colouration [199]. Identifying camouflage is pivotal in many wildlife surveillance applications [72, 316], as it assists in locating hidden individuals for study and protection.

In fact, localisation of camouflaged objects [65, 89], such as Camouflaged Instance Segmentation (CIS), is an important research topic in computer vision, whose challenge lies in the need to learn discriminative features that can be used to discern camouflaged target objects from their surroundings. Existing COD techniques can be utilised to roughly identify camouflaged objects at regional scales (via bounding boxes), but they are not designed for distinguishing individual instances at finer scales. CIS operates under the conditions where object features closely resemble each other, resulting in class-independent segmentation masks [210]. However, the diversity of camouflages within a single scene can lead to complex intertwining patterns, making the task especially more challenging in severe environmental conditions, *e.g*. terrestrial and aquatic environments, and poor imaging quality, *e.g*. occlusions, image blurriness, and low-light conditions in underwater-applications. These challenges also hinder the collection and annotation processes for high-quality data that can be used for training and testing CIS algorithms.

Meanwhile, as humans look at the world and can recognise a limitless number of target categories, open-vocabulary recognition has been developed to mimic human intelligence at unbounded understanding, yet current endeavors have only focused on generic objects and individuals [332, 77, 57, 76, 196, 140, 313]. For example, while Xu *et al*. [313] found that Internet-scale text-to-image diffusion models can be utilised to create a state-of-the-art open-vocabulary segmenter

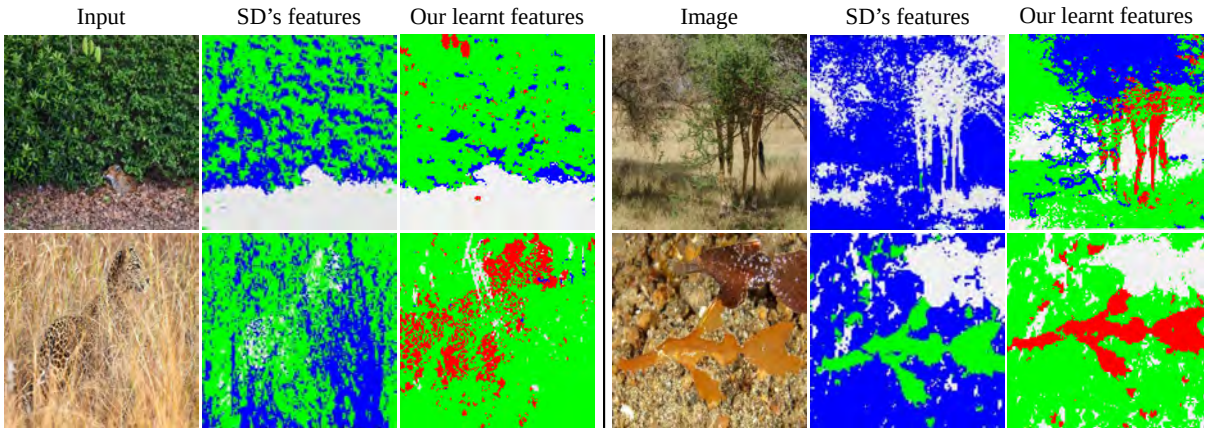


Figure 5.1: Illustration of textual-visual features of off-the-shelf Stable Diffusion when dealing with CIS and our learnt features. Given an input image, textual-visual features are extracted and clustered using a K-means clustering algorithm. As shown, camouflaged animals can be localised from the clustering results. We leverage these rich features to perform instance segmentation of camouflaged objects. This figure is best viewed in colour.

for many concepts, our investigations show that they demonstrate inconsistency and a lack of discernment when it comes to camouflaging effects, as shown by their pixel-wise embeddings in Figure 5.1. Existing works in open-vocabulary segmentation [50, 314, 313, 362, 363, 336] share the similar traits as the ability to detect camouflages are not primary to their designs. We believe the proposed framework may open new avenues in enhancing surveillance systems, wildlife monitoring, and military reconnaissance. Nevertheless, it is inherently challenging to enable the robust generalization across a diverse array of concealed targets.

In order to overcome the aforementioned hurdles, we propose a method that leverages text-to-image diffusion to address the problem of open-vocabulary CIS. Our method is inspired by the advanced representation learning ability of diffusion techniques and language-vision transferability of text-image models. Text-to-image diffusion models, *e.g.* the stable diffusion model by [230], are designed to learn essential object features against noise, so they can be useful in extracting features relevant to the target objects in noisy and cluttered backgrounds. While we observed that features learnt solely from the visual domain are weak to distinguish camouflaged objects from their surroundings, the features learnt by text-image discriminative models, *e.g.* CLIP [221], still contain rich information about the real world thanks to the variety of concepts in open-vocabulary training data. We hypothesise that an effective combination of features learnt from both the textual and visual domains would benefit representation learning of camouflaged objects. We illustrate the effectiveness of textual-visual representations for CIS in Figure 5.1. To the best of our knowledge,

such a cross-domain combination with open-vocabulary for CIS is *novel*, and ours is the *first framework* for localizing camouflaged object instances at such a scale.

To effectively learn textual-visual representations of camouflaged objects, our method assimilates an input image and a text prompt about the objects included in the input image, so the input image and its implicit caption (generated by a captioner) are integrated into a text-to-image diffusion model to extract visual features. While our method shares a similar high-level perspective with the works by [313, 346], These features are processed at multiple scales and fused into a visual feature map, which is then used to generate object masks. Simultaneously, textual features are extracted from the text prompt using a text encoder. These textual features are enriched from open-vocabulary category labels and proven to improve the discriminative power of camouflaged objects' representations against the background. our proposed textual-visual pipeline aggregates textual and visual features in a mask-out manner to recognise the masks of the target objects. The diffusion model utilises a cross-attention mechanism to link textual features with visual features and condition the feature learning process, so the learnt features are likely to be distinct and connected to high/mid-level semantic notions that may be expressed in the language part. Our pipeline is more specialised to CIS by designing camouflage-specialised modules.

In summary, we make the following contributions to our work:

- We introduce a new challenging task, open-vocabulary camouflaged instance segmentation (OVCIS) and indicate a possible lack of an effective transfer learning mechanism in open-vocabulary segmentation regarding camouflages.
- We propose a first method for CIS, which is built upon text-to-image diffusion and text-image transfer techniques with open-vocabulary utilization.
- We propose an open-vocabulary-based object representation learning paradigm, specifically through a Multi-scale Features Fusion (MSFF) module to encapsulate visual features from diffusion, a Textual-Visual Aggregation (TVA) module to utilise textual information that pronounces visual features, and a Camouflaged Instance Normalisation (CIN) module to adaptively capture textual-visual information that enhances the camouflaged representations.
- We conduct extensive experiments and ablation studies that demonstrate the advantages of our method over existing works.

5.2 Related Works

We start our review of related work with an overview of deep learning-based advances for camouflaged object understanding. Following it, we delve into contemporary research in text-to-image diffusion, thereby discussing their role in facilitating open-vocabulary computer vision. Then, we review prior research on generative models and their applications to visual segmentation. Our discussion extends to the notion of open-vocabulary recently emerging into the field of computer vision by the potential in making object representation learning at scale.

5.2.1 Camouflaged Object Understanding

The main aim of camouflaged object understanding lies in learning object representations that are difficult to dissimilate from their background. Existing research works have tried to address various tasks in camouflaged object understanding from images. For instance, [253] counted objects that blended seamlessly into backgrounds. Following closely, [181] identified salient image regions of hidden objects that align with the nuances of human perception. Camouflaged object detection (COD) was studied by [89], in which the authors decomposed learnt features into different frequency bands using learnable wavelets to identify the most informative features to differentiate target objects and backgrounds. In addition, an auxiliary *e.g.* reconstruction network was built to further boost up the discriminative power of the foreground's features against the background's ones. In the work by [64], a method for segmenting camouflaged objects was proposed to segment obscured objects without the necessity to pinpoint specific categories for the objects.

Camouflaged instance segmentation (CIS) was brought forth by [210] to emphasise the learning of object-vs-background-discriminative representations, which is different from general instance segmentation [309] that aims to maximise inter-object distances. Although this goal is common in existing camouflaged object understanding methods and various attempts have been made to address it in the literature, learning such representations from solely imagery data is challenging as it is the nature of visual camouflage. Our research differs by exploring the potential and richness of diffusion representations and textual data as additional cues to drive the open-vocabulary learning of CIS, thereby utilising them to make object representations adaptive to novel objects that are never seen in training. Thanks to the variety of concepts, textual features learnt from text prompts about the objects included in an input image can help to localise relevant visual features. In addition, an

effective combination of both textual and visual features would further enhance the robustness of object representations in camouflage. To our knowledge, ours is the first of such work.

5.2.2 Text-to-image Diffusion

Significant progress has been made in Artificial Intelligence (AI)-empowered picture creation with recent advances in large-scale text-to-image models, including Stable Diffusion [230], DALL-E 2 [223], and Imagen [233]. These models have demonstrated photo-realistic quality image generation by being trained on text-image datasets of substantial scale sourced from the Internet. They also have shown the ability to be conditioned on unrestricted text prompts in order to produce visuals that closely resemble real-life photographs. The utilisation of text-to-image diffusion models has facilitated the creation and manipulation of visual contents in an ever-easy and convenient manner via language-based interactions (*e.g.*, text prompts). This has enabled a wide spectrum of applications such as content-personalised customisation [139], zero-shot translation [209], content editing [97], and image generation [75].

In this work, we do not utilise the text-to-image diffusion technique for image creation and/or manipulation. Rather, we explore its capability of cross-domain feature learning. Most related to our work, Xu *et al*. [313] have also shown that pre-trained representations in diffusion models can be utilised for open-vocabulary segmentation in the wild. To address its limitations with regard to camouflages, which led to poor consistency and lack of boundary discernment, we devise a feature fusion strategy based on a state-of-the-art text-to-image diffusion architecture to fuse image features with implicit caption features at multiple scales. Our experiments show that such a fusion facilitates the learning of object-vs-background discriminative features, which are crucial for CIS.

5.2.3 Generative Models for Segmentation

Many studies related to our work in terms of applying image generative models, such as Generative Adversarial Networks (GANs) [60, 125] or diffusion models [98, 248, 48], to semantic segmentation [149, 7, 229]. For the use of GANs, a straightforward approach is to synthesise images and their corresponding semantic maps to train a segmentation network [149]. [229], segmentation is proceeded by training of a generative model on datasets with limited vocabulary. For example, in the diffusion-based framework, DDPMSeg [7] was also built upon the denoising diffusion probabilistic model (DDPM) [98] to learn a feature map for an input image. The feature map was then passed

to a pixel classifier to perform semantic or part segmentation. A small number of hand-annotated examples per category are then utilised to classify learnt representations into semantic regions. Similarly, Xu *et al*. [313] showed that pre-trained representations in diffusion models can be utilised for open-vocabulary segmentation in the wild. Their insight suggests that the internal representation of diffusion models can well-correlate to high/mid-level semantic concepts that can be described by language, hence dealing with the lack of spatial and relational understanding in traditional open-vocabulary segmentation. Hence, the approach introduced a new capacity for generative models, *e.g*. image generation-driven representation learning. However, we found that the diffusion-based pre-trained representations are not made for tackling camouflaging effects, yet the intermediate representations of a generative model could be learnt with high-level semantic concepts (*e.g*. the presence of an object in an input image) under specific feature constraints.

5.2.4 Open-vocabulary Detection and Segmentation

Numerous research studies have been proposed to incorporate vision-language models (VLMs) into open-vocabulary detection and segmentation [354, 332, 77, 57, 76, 196, 226, 140]. This has enabled detection and classification of novel objects from a vast conceptual domain with help of pre-trained VLMs [338, 300]. OVR-CNN was the first open-vocabulary object detection introduced by [333], which underwent pre-training with image-caption data in order to learn and identify unknown objects, followed by fine-tuning for zero-shot detection.

Following recent advances in VLMs [221, 115], ViLD [83] pioneered the incorporation of extensive representations of pre-trained CLIP [221] into an object detector, and many works [57, 140, 359] have followed the similar framework. [57] proposed DetPro, a sophisticated automated prompt learning method, to learn the presence of an object into a background via prompt training. F-VLM [140] adopted a frozen VLM to generate new object categories based on cropped CLIP features. [359] extended the ability of the well-known object detector, Faster R-CNN [228] to newly introduced object categories by replacing the classification weights (in the classification head) by fixed language embeddings learnt from open-vocabulary.

Despite the successes achieved, existing methods have limited capabilities against camouflaged objects due to the utilisation of small closed vocabularies and/or the incorporation of VLMs for generic object classes which are often distinguishable from the background. It is because the pre-trained representations are not designed for discerning camouflaged boundaries of individuals in the

wild but only on general classes of objects [50, 314, 313, 362, 363, 336]. While exploiting insights and advantages from prior studies, our work stands out in a specifically focused direction: tackling the challenge of open-vocabulary instance segmentation for camouflaged targets, yet without losing much representation localisation capability on general objects. In particular, our proposed method aims to segment novel object categories with a concealed appearance in the natural environment with support from an open-vocabulary set.

5.3 Proposed Method

We aim to build and train an instance segmentation model with a set of pre-defined object categories, referred to as $\mathbf{C}_{\text{train}}$. The instance segmentation model can work on a new domain with $\mathbf{C}_{\text{test}}$ object categories, where $\mathbf{C}_{\text{test}}$ and $\mathbf{C}_{\text{train}}$ may or may not share common object categories. In other words, $\mathbf{C}_{\text{test}}$ may include object categories previously unseen during the training of the instance segmentation model. Throughout the training process, it is presumed that binary mask annotations for target objects in each training image are available. Moreover, each mask is either associated with a category name or a caption presented in the text form. During the testing phase, however, neither the category label nor the caption is accessible for any test image. Only the names of the test categories in $\mathbf{C}_{\text{test}}$ are provided.

5.3.1 Preliminaries

We build our method upon two technical advances: text-to-image diffusion and text-image transfer. We first briefly summarise those techniques, then we describe how they can be applied to our method.

Text-to-image Diffusion facilitates the creation of high-quality images guided by text prompts. A text-to-image diffusion model is trained on a massive corpus of image-text pairs amassed through web crawling [201, 233, 313], with text inputs being encoded into embeddings using an established text encoder, *e.g.*, T5 [222]. In the forward process, an image is perturbed by iteratively more Gaussian noise at a controlled intensity through the diffusion network. The network is fine-tuned to reverse the noise application in the reverse process, utilising noisy images and associated text embeddings to diminish the distortion. In the inference phase, the model synthesises an image from inputs including pure Gaussian noise shaped to the image's dimensions and a user-provided description's text embedding. Through successive inference iterations, the model gradually denoises the input and finally results in a photo-realistic image of the user-provided text description.

In our work, we adopt the Stable Diffusion (SD) model developed by [230]. The SD model is composed of some elements: a captioner (realised by a pre-trained text encoder) that generates a text embedding for an input image; a pre-trained variational auto-encoder for learning of image representations; and a denoising time-conditional U-Net $\epsilon_\theta(\cdot)$, which applies progressive convolution operations to downsample and upsample feature maps of an input image with skip connections. Within the U-Net, textual-visual interactions are enabled by cross-attention. In detail, the captioner projects a text input y into an embedding, which is then transformed into `Key` and `Value` pairs. At the same time, a feature map of a noisy image undergoes a linear projection to form a `Query`. This design allows for iterative updates of input images conditioned on accompanying text descriptions.

Training of the SD model is outlined as follows. For a given pair $(\mathcal{I}, y)$ in a training dataset, the image $\mathcal{I}$ is encoded into a latent representation z and then subjected to noise, resulting in a noised vector $z^t := \alpha^t z + \sigma^t \epsilon$, where $\epsilon \sim \mathcal{N}(0,1)$ is a noise variable, and α^t, σ^t are parameters that manage the noise level and the fidelity of each sample. The training aims to fine-tune the time-conditional U-Net $\epsilon_\theta(\cdot)$ to anticipate the noise vector ϵ and to accurately reconstruct the initial latent vector z, while being conditioned on the text input y. The fine-tuning is performed by using a loss function that minimises the mean squared error of noise prediction:

$$\mathcal{L}_{\text{diffusion}} = \mathbb{E}_{z,\epsilon\sim\mathcal{N}(0,1),t,y}\left[\|\epsilon - \epsilon_\theta(z^t, t, y)\|_2^2\right], \tag{5.1}$$

where the time variable t is randomly selected from the set $\{1,\ldots,T\}$.

During the inference phase, the SD model synthesises an image by sequentially refining a latent vector $z^T \sim \mathcal{N}(0, I)$ conditioned on a text input y. Specifically, for each time step $t = 1,\ldots,T$ of the denoising sequence, z^{t-1} is derived from the current z^t and the U-Net's noise prediction, which in turn takes z^t and the text prompt y as inputs. Upon completion of the final denoising stage, the latent vector z^0 is transformed back to produce a final output image $\mathcal{I}'$. The SD model is heavily pre-trained on the LAION-5B dataset [238] and performs diffusion in a latent space. Training the diffusion model is extremely computationally expensive[230], however, as we showed in Figure 5.1, the embeddings of SD model **are not suitable** for camouflages **without extra processing**.

Text-image Transfer originally aims to learn directly from raw text about images. It leverages rich textual representations learnt from the textual domain to scale up representation learning in the visual domain. As shown in the literature, natural language can be used to supervise a wide set of visual concepts through its generality [234, 47, 340]. Recently, CLIP proposed by [221] offers the

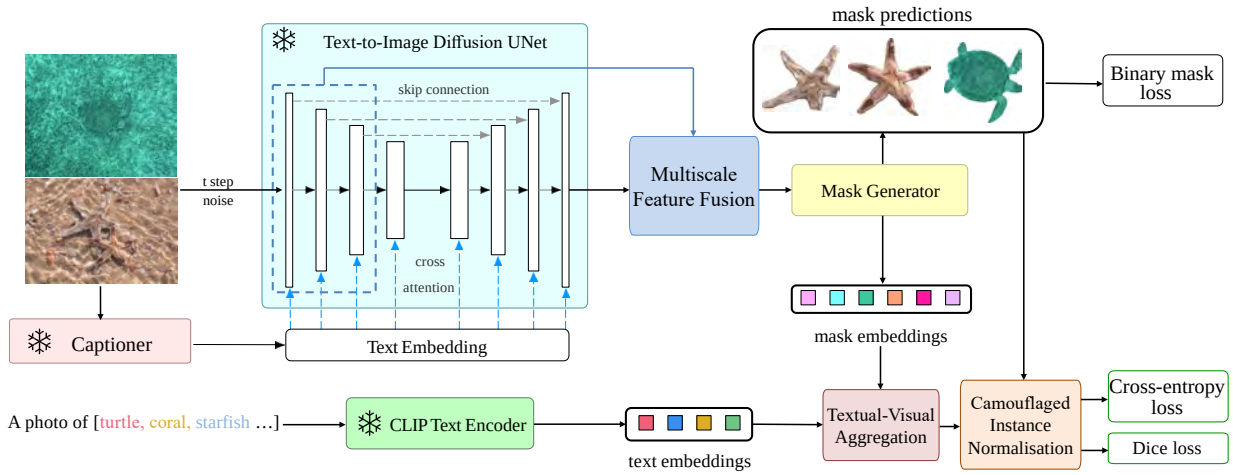


Figure 5.2: Pipeline of our proposed method for Camouflaged Instance Segmentation. Inputs include an image and a text prompt of target objects (novel or unseen in the training data). Outputs include instance masks of the target objects. We leverage state-of-the-art text-to-image diffusion and vision-language models to learn textual-visual features that facilitate representation learning for segmenting camouflaged objects.

text-image transferibility in both directions, *i.e.*, text-to-image and image-to-text.

In our work, we adopt a CLIP model [221] pre-trained on 400 million image-text pairs crawled from the Internet. This model is used to generate text embeddings for implicit captions of input images and text embeddings for text prompts associated with input images. We observed that these text embeddings, thanks to large-scale training, can provide considerable aids to improve the representation of camouflaged objects. The improved representations are also shown in Figure 5.1.

5.3.2 Our Pipeline

Figure 5.2 illustrates the pipeline of our method. At an abstract level, our method takes an image and a text prompt about target objects as inputs and produces instance masks with object categories for the target objects as outputs.

The visual image is first passed to the SD model, which is pre-trained and frozen, to extract latent features. The input image is also fed to the pre-trained and frozen CLIP model to calculate its implicit caption embedding. The caption embedding is inserted into the SD model at various scales (layers) and fused with the SD model's last layer to form image-guided features. These features are "image-guided features" with textual information, as the textual features from the implicit caption embedding are driven by the input image. These features are then combined at different scales by our proposed Multi-scale Features Fusion (MSFF) module, whose outputs are coupled with

annotated training masks, and serve as inputs to train a mask generator capable of producing instance masks for all potential categories within the input image. The instance masks are then used to locate object-relevant features in a mask-out manner. This step results in mask embeddings (*i.e.*, features extracted within masked regions).

The textual prompt is processed by CLIP, independently of the input image, to obtain its corresponding embeddings. These text embeddings are transferable to visual features yet extracted from the textual input, hence considered as "text-guided features".

The textual-visual representations are extracted under an aggregation process of text embeddings (text-guided features) and mask embeddings (image-guided features) using the Textual-Visual Aggregation (TVA) module. It aims to emphasise the learnt features towards foreground objects defined in the input text prompt, resulting in a textual-visual representation of the input image and text prompt. Through our proposed Camouflaged Instance Normalisation (CIN) module, the representations are normalised concerning the instance masks segmented by the mask generator and classified into object categories by a mask classifier.

Open-vocabulary capabilities are facilitated as the entire pipeline is trained with object categories in $\mathbf{C}_{\text{train}}$, while the frozen SD and CLIP models were pre-trained at Internet scales. The training of the entire pipeline is equivalent to learning parameters in modules specialised for camouflage instance segmentation (multi-scale feature fusion, mask generator, textual-visual aggregation, camouflaged instance normalisation). Once the training is completed, the inference process performs open-vocabulary instance segmentation, *i.e.*, instance segmentation of novel object categories in $\mathbf{C}_{\text{test}}$.

Towards open-vocabulary with CIS, we have developed several technical components to facilitate camouflaged object representation learning (see Section 5.3.3) and camouflaged instance normalisation (see Section 5.3.4).

5.3.3 Camouflaged Object Representation Learning

Given the features learnt by the SD model from the input image and the text embeddings produced by the CLIP from the input text prompt, we perform camouflaged object representation learning via three modules: ❶ multi-scale feature fusion (MSFF), ❷ mask generator, and ❸ TVA, where the mask generator is inspired from an existing work [29], and both MSFF and TVA are our proposed modules. These modules are described below.

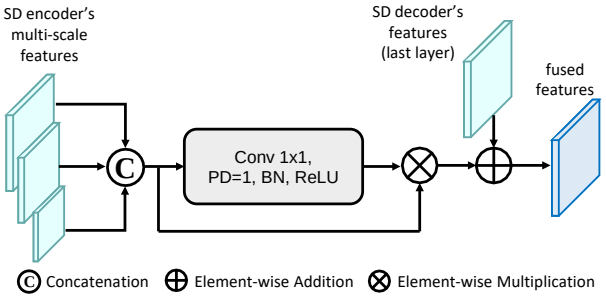


Figure 5.3: Architecture of the multi-scale features fusion (MSFF) module.

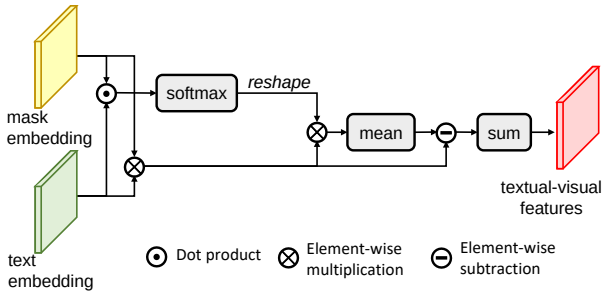


Figure 5.4: Architecture of the Textual-Visual Aggregation (TVA) module.

Multi-scale Features Fusion (MSFF). While the features obtainable from the last layer of SD may be comprehensive, fine-grained information may be lacking due to the blending effects of camouflage, as shown in Figure 5.1. Therefore, to capture boundary information better in general and camouflaging cases and also to distinguish foregrounds from backgrounds, we propose that MSFF fuse multi-scale features from the encoder part and the features from the last layer of the decoder part of the SD model. The architecture of MSFF is shown in Figure 5.3. The fusion process is realised via a series of operations, including concatenation (of the SD encoder's features at multiple scales), 1×1 convolution (of the concatenated features), element-wise multiplication (between the output of the convolution and the concatenated features), and element-wise addition (between the output of the element-wise multiplication and the SD decoder's features). The visually discriminative features are trained by adapting with the mask generator, similar to Mask2Former [29].

Mask Generator. We adopt the decoder in the mask-attention transformer, the core component in the Mask2Former architecture [29], to realise our mask generator. The mask generator receives input as a fused feature vector from the MSFF module and produces outputs including N class-agnostic binary masks $\{m_i^{pred}\}_{i=1}^{N}$ and their corresponding N mask embedding features $\{z_i^{pred}\}_{i=1}^{N}$ for all possible objects in the input image. We present the architecture of the mask generator in Figure 5.5.

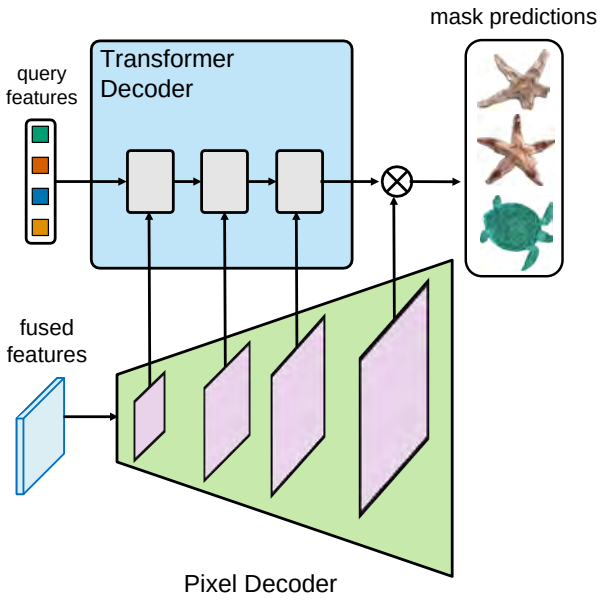


Figure 5.5: Architecture of the Mask Generator.

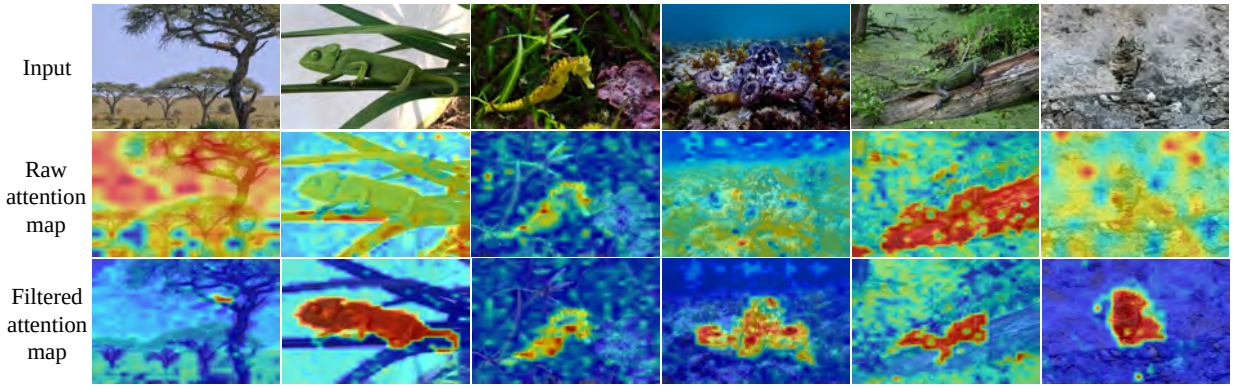


Figure 5.6: Visualization of interim and final results of the TVA module.

The mask generator employs a pixel decoder that progressively increases the resolution of fused features returned by the MSFF module and generates per-pixel high-resolution embeddings. This pixel decoder is designed with meticulous attention to detail, using multiple layers to capture fine-grained and broad contextual information. Following that, a Transformer's decoder processes intermediate feature maps in the pixel encoder to handle object queries, which are initialised randomly but then learnt through training. To effectively process the intermediate feature maps in the pixel decoder, the mask generator guides each feature map at a scale to an individual layer in the transformer's decoder. Consequently, each layer in the transformer's decoder focuses on a feature map at a specific scale in the range $\{1/32, 1/16, 1/8\}$. We observed that this strategy significantly enhances the ability of the mask generator to handle objects in various sizes.

Textual-Visual Aggregation (TVA). Given the mask embeddings of visual features and the textual features, instead of simply concatenating them for later segmentation like [313], we also propose a module specifically designed to highlight object-relevant features, thereby driving the learning of object representations towards identifying foreground objects in general. We show its architecture in Figure 5.4. As examples, we show in Figure 5.6 the qualitative raw attention maps before being processed by TVA and the attention maps after TVA.

The TVA module operates as follows. Similar to Mask R-CNN [91], for each object mask returned by the mask generator, we crop corresponding features from the MSFF module and perform mask pooling. This step results in mask embeddings (*i.e.*, embeddings are determined by masks). We then compute the interactions between these mask embeddings and the text embeddings produced by the CLIP. Nevertheless, instead of directly using a dot product to calculate the interaction between two embeddings as in CLIP [221], we apply a softmax operator to the dot product of the embeddings

to weight features, then apply mean-normalisation to remove irrelevant features prior to aggregating them by a channel-wise summation. Removing irrelevant features helps to mitigate the problem of noisy activations, making the learning process lean towards features relevant to the object categories specified in the input text prompt.

Figure 5.1 visualizes the learnt textual features by our method on several difficult cases. Despite how targets blend into backgrounds, the learnt textual-visual features on camouflaged objects can be well identified and located. It demonstrates our method's ability to learn the distinguishing object-vs-background features.

5.3.4 Camouflaged Instance Normalisation (CIN)

Inspired by adaptive instance normalization in neural networks [108, 210], we developed a CIN module to achieve final masks for the target objects. CIN aims to adaptively emphasize useful aspects of visual-textual foreground information with respect to the foreground-background difference during learning, thereby enhancing model performances at dealing with camouflaging effects. We present the architecture of the CIN module in Figure 5.7. The CIN module takes inputs as a textual-visual feature map from the TVA module and an object mask from the mask generator. The textual-visual feature map is first projected into a higher-dimensional space by a linear layer. Next, affine weights and biases are attained by applying two subsequent linear layers to the result of the first linear layer. The affine weights and biases are then combined, together with the input mask from the mask generator, to predict a final instance mask for the object specified in the input mask. Since CIS is category-agnostic, we use a confidence score for the existence of a camouflaged object, rather than a classification score like in generic segmentation.

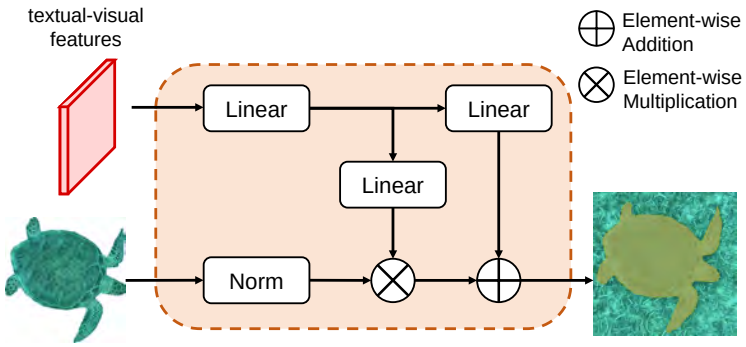


Figure 5.7: Architecture of the Camouflaged Instance Normalisation (CIN) module.

5.3.5 Training

We train the entire pipeline of our method by optimising the loss functions used in the mask generator and the CIN module with supervision. Specifically, we adopt a binary cross-entropy loss as our binary mask loss $\mathcal{L}_{\text{bce}}$ and a dice loss $\mathcal{L}_{\text{dice}}$ [195] for supervising binary mask predictions in the mask generator to remedy class imbalance.

The training of the CIN module is carried out under the conventional close-vocabulary training approach. Suppose that we can access the ground-truth category label for each object mask during the training phase. For each mask embedding z_i^{pred} produced by the mask generator, let $y_i^{cate} \in \mathbf{C}_{\text{train}}$ be the corresponding ground-truth category of z_i^{pred}. We invoke the text encoder $\mathcal{T}$ in the pre-trained CLIP model to encode the names of all categories in $\mathbf{C}_{\text{train}}$. This results in a set of text embeddings $\mathcal{T}(\mathbf{C}_{\text{train}}) = \left\{\mathcal{T}(c_1), \ldots, \mathcal{T}\left(c_{|\mathbf{C}_{\text{train}}|}\right)\right\}$ where $c_k \in \mathbf{C}_{\text{train}}$ represents a category name. Thus, the loss for embedding classification (*i.e.*, associating mask embeddings m_i^{pred} with their categories y_i^{cate}) is calculated as:

$$\mathcal{L}_{\text{ce}} = \frac{1}{N}\sum_{i=1}^{N} \text{CE}\left(\text{Softmax}\left(\frac{z_i^{\text{pred}}\mathcal{T}(\mathbf{C}_{\text{train}})}{\tau}\right), y_i^{cate}\right), \tag{5.2}$$

where τ is a learnable temperature parameter and CE is the cross-entropy loss for the classification of each training embedding.

The total loss for the training of our pipeline is finally defined as,

$$\mathcal{L} = \alpha\mathcal{L}_{\text{bce}} + \mathcal{L}_{\text{dice}} + \mathcal{L}_{\text{ce}}, \tag{5.3}$$

where α is a hyper-parameter, empirically set to 0.4. Furthermore, in line with work done by [30], we apply the Hungarian matching [138] to match predicted masks with ground-truth masks and compute the loss between matching pairs.

5.4 Experiments

5.4.1 Datasets

Following previous studies [349, 313, 51, 346], we used the instance segmentation part of the MS-COCO dataset [166] with 80 object categories to pre-train our model. Pre-training our proposed framework on the MS-COCO dataset helps to emphasise the discernment of prompted objects

from their backgrounds in the wild, specifically for open-vocabulary segmentation performance. For closed-set validation, fine-tuning the model on the 3,040 images from the training set of the COD10K-v3 dataset [64] further adapts the model to camouflaged objects and significantly boosts up the performance of our method.

We tested our method on two benchmark camouflaged object datasets: the test set of the COD10K-v3 (including 2,026 images) and the NC4K [181] (including 4,121 images). The NC4K dataset contains only test images. The training sets (for both pre-training and fine-tuning) and the test sets (for both the COD10K-v3 and NC4K) share only 6 common object categories (out of 80 and 69 object categories from the MS-COCO and COD10K-v3/NC4K, respectively). This setting, *i.e.*, cross-dataset training-testing, has been used widely in the evaluation of the generalisation ability of CIS models. It reflects the practicality of CIS, thus ensuring the reliability of evaluations.

We also evaluated our method on generic open-vocabulary datasets including the ADE20K [357] and Cityscapes [40]. For the ADE20K dataset, we used the validation set of the short version [355] covering 150 object categories and 2,000 images. The Cityscapes dataset contains a total of 19 classes, which are divided into 11 "stuff" and 8 "thing" classes. We conducted evaluations on the validation set of the Cityscapes, including 500 images. *Note that,* we pre-trained our method on the MS-COCO dataset and then directly evaluated the method on these open-vocabulary datasets without fine-tuning.

5.4.2 Implementation Details

We implemented our method in Pytorch and built it on the Detectron2 framework [301]. We trained our method for 90k iterations with a batch size of 64 on 4 NVIDIA A40 GPUs. All training images were resized to 512×512-pixels. Random jitters in the range $[0.1, 2.0]$ were applied to the training images. We froze both the SD and CLIP models during training. We adopted the Adam optimiser [177] with the learning rate γ set to 10^{-4} and weight decay of 0.05. We used a step learning rate scheduler and reduced the learning rate by a factor of 10 at 81k and 86k iterations. The training took 4.3 days to complete. Due to class imbalance in the COD10K-v3 dataset, we manually removed some extremely rare classes, *e.g.*, classes with less than five instances. In addition, we applied the `RepeatFactorTrainingSampler` from the Detectron2 framework, to allow a sample to appear more times than others based on its repeat factor.

5.4.3 Results

We evaluated our method and existing works using the average precision (AP) measured at different intersection-over-union (IOU) thresholds. In particular, we calculated the overall AP in the range $[50\%, 95\%]$ for the IOU thresholds (*i.e.*, for a threshold within the above range, a predicted instance is considered as true positive if there exists a true instance in the ground-truth such that their IOU is equal or greater than that threshold). We also measured detailed AP for the IOU thresholds of 50% (AP50) and 75% (AP75).

Open-Vocabulary Generalization on Camouflaged Object Datasets. We report the performance of our method on camouflaged object datasets (the COD10K-v3 and NC4K) in Table 5.1 (last row). Recall that, following the conventional setting in CIS, *e.g.*, [349, 313, 51, 346], we pre-trained our model on the MS-COCO dataset and then fine-tuned it on the training set of the COD10K-v3 dataset. To show the effectiveness of this strategy, we experimented with a variant of our method by skipping the fine-tuning phase. In particular, we pre-trained our method on the MS-COCO dataset and then evaluated it directly on the test set of the COD10K-v3 and the NC4K datasets. We show the performance of this strategy in the second last row, denoted as "Ours", in Table 5.1. Experimental results show that fine-tuning the method on a camouflaged object dataset denoted as "Ours (task-specific)", significantly improves its performance on all evaluation metrics.

We compare our method with existing instance segmentation methods on the CIS task in Table 5.1. We group existing methods into two groups: "closed-set supervised learning approach", which follows the traditional fashion of supervising an instance segmentation model on a training set and tests the model on a test set, and "open-vocab text-to-image approach", which includes methods using text-to-image diffusion techniques with open-vocabulary. The training and test sets of this approach are in the same domain and include imagery data only. Most existing instance segmentation methods lie in the first group, and we benchmark them on the training set of COD10K-v3. Our method and related works [50, 314, 313, 362, 363, 336] belong to the second group.

In Table 5.1, we show that our method with the pre-training setting significantly outperforms ODISE on all evaluation metrics, making a new state-of-the-art for open-vocabulary CIS. With pre-training and fine-tuning, our method also performs on par with DCNet [178] (best method of the "closed-set supervised learning approach"), while requiring much fewer parameters. We observed that, in general, open-vocabulary methods that do not utilise text-to-image diffusion, achieve lower

Table 5.1: Comparison of our method with existing instance segmentation methods on the test set of the COD10K-v3 and the NC4K datasets. Methods of the "closed-set supervised learning approach" are trained on the training set of the COD10K-v3 dataset. Methods of the "open-vocab text-to-image approach" are pre-trained on the MS-COCO dataset. We denote "Ours" and "Ours (task-specific)" for two variants of our method without and with fine-tuning on the training set of the COD10K-v3 dataset. Params (M) denotes the number of *trainable* parameters. The best results are **bold**, and the second best results are underline.

Method		COD10K-v3 Test			NC4K			Params (Milions)
		AP	AP50	AP75	AP	AP50	AP75	
closed-set supervised learning	Mask R-CNN [91]	25.0	55.5	20.4	27.7	58.6	22.7	43.9
	MS R-CNN [109]	30.1	57.2	28.7	31	58.7	29.4	60.0
	Cascade R-CNN [14]	25.3	56.1	21.3	29.5	60.8	24.8	71.7
	HTC [23]	28.1	56.3	25.1	29.8	59.0	26.6	76.9
	YOLACT [12]	24.3	53.3	19.7	32.1	65.3	27.9	35.3
	BlendMask [21]	28.2	56.4	25.2	27.7	56.7	24.2	35.8
	SOLOv2 [291]	32.5	63.2	29.9	34.4	65.9	31.9	46.2
	Condlnst [266]	30.6	63.6	26.1	33.4	67.4	29.4	34.1
	Querylnst [71]	28.5	60.1	23.1	33.0	66.7	29.4	172.5
	SOTR [85]	27.9	58.7	24.1	29.3	61.0	25.6	63.1
	MaskFormer [31]	38.2	65.1	37.9	44.6	71.9	45.8	45.0
	Mask2Former [29]	39.4	67.7	38.5	45.8	73.6	47.5	43.9
	Mask Transfiner [129]	28.7	56.3	26.4	29.4	56.7	27.2	44.3
	OSFormer [210]	41.0	71.1	40.8	42.5	72.5	42.3	46.6
	DCNet [178]	**45.3**	70.7	**47.5**	**52.8**	**77.1**	**56.5**	53.4
	Ours (task-specific)	44.9	**70.9**	47.2	52.7	76.6	55.8	28.7
open-vocab VLM (*w/o finetuning*)	MaskCLIP [50]	3.3	5.9	4.1	6.3	5.6	6.5	542.0
	MasQCLIP [314]	4.1	7.7	5.8	8.0	7.6	8.4	375.2
	X-Decoder [362]	7.7	12.9	7.5	3.9	8.1	3.4	38.3
	SEEM [363]	6.6	10.8	6.5	9.2	12.7	9.9	415.3
	OpenSeeD [336]	6.1	10.4	5.9	9.3	14.5	9.8	116.2
open-vocab T2I (*w/o finetuning*)	ODISE [313]	21.1	37.8	20.5	22.9	37.2	21.4	28.1
	Ours	**23.4**	**43.8**	**22.6**	**24.3**	**43.7**	**23.5**	28.7

performances. The main reason for this issue is their designs focus only on common objects like humans, butterflies, rabbits, birds, cats, ducks, *etc*., thus fail to describe concealed objects. As

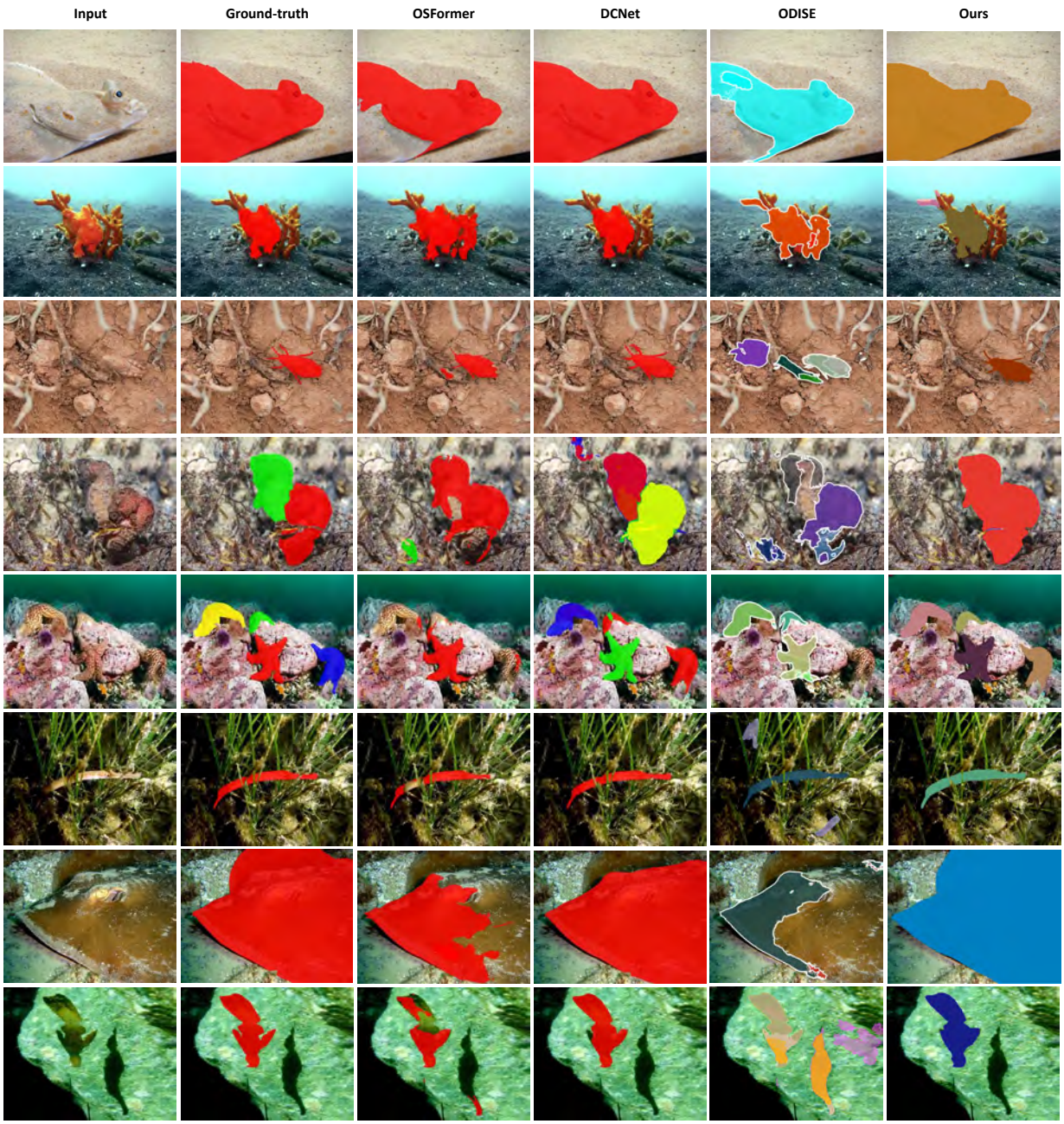


Figure 5.8: Qualitative comparison on COD10K-v3 and NC4K. Our method outperforms others and has competitive performance as DCNet [178] at a twice smaller model size.

shown, the Stable Diffusion model (SD), being trained with large-scale and open-vocabulary datasets, provides complementary information that can enrich representation learning of camouflaged objects.

In summary, with regard to both the segmentation accuracy and memory usage, our method is more advanced, compared with existing ones. Recall that only 6 object categories are shared between MS-COCO (80 categories) and COD10K-v3/NC4K (69 categories). This challenge shows the ability

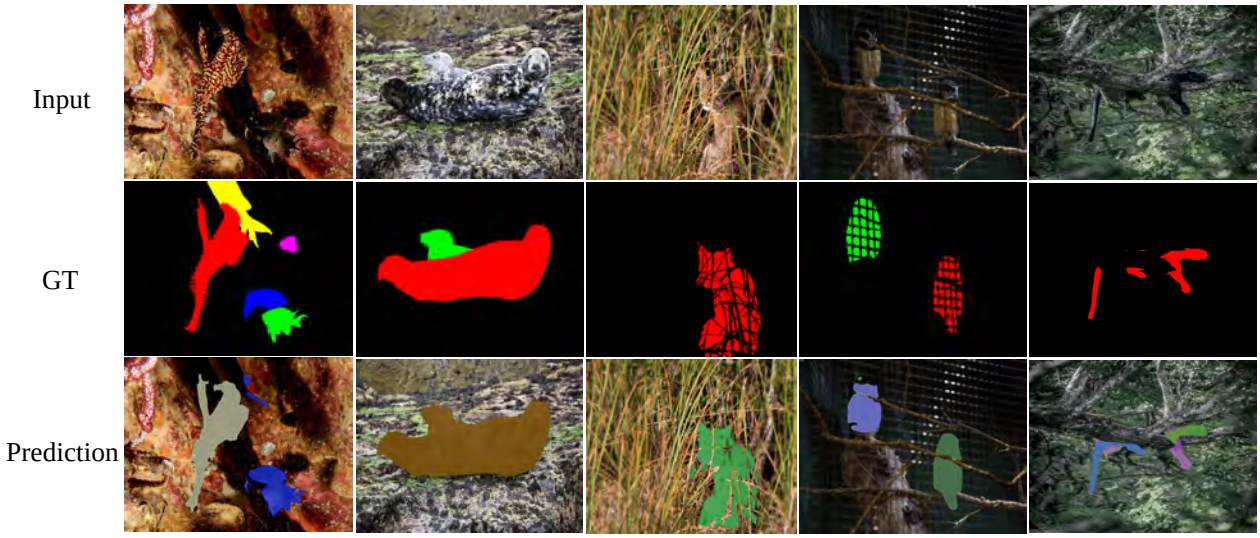


Figure 5.9: Failure cases on the COD10K-v3 dataset, illustrating that our method may fail to separate instances of occluded targets. For example, only the black panther's body parts can be observed in the last column.

of our method to handle open-vocabulary tasks. We visualize several results of our methods and existing ones in Figure 5.8, where our method excels at accurately delineating camouflaged objects along their blurry boundaries in cluttered backgrounds at significant proficiency.

Figure 5.9 illustrates failure cases of our method. In the first and second columns, our method fails to separate instances of nearby and similar objects, such as the yellow fish and two sea lions. Our method can detect and segment camouflaged objects in the third and fourth columns but with slightly less accurate boundaries. In the last column, our method struggles with the significant spatial separation of the black panther's body parts, leading to misclassification of the entire object. We found that our method would be ineffective in distinguishing and separating an object that shares very similar characteristics with others or consists of fragmented parts. However, such circumstances would also be challenging for human beings as well.

Open-Vocabulary Generalization on Generic Datasets. To showcase the versatility and generality of our method in various application domains (other than camouflaged objects), we evaluated our method on the ADE20K [357] and Cityscapes datasets [40], two widely used open-vocabulary benchmark datasets. Note that these datasets are not designed for camouflage detection and segmentation. We summarise the performance of our method and existing open-vocabulary instance segmentation methods in Table 5.2. Our method ranks second on both ADE20K and Cityscapes. Nevertheless, compared with the first ranked method, *i.e.*, OpenSeeD [336], our method uses approximately 4

Table 5.2: Comparison with existing open vocabulary instance segmentation methods using AP metric. The best and second results are **bold**, and underline.

Method	ADE20K	Cityscapes	Params (M)
MaskCLIP [50]	6.2	-	542.0
ODISE [313]	13.9	-	28.1
X-Decoder [362]	13.1	24.9	38.3
OpenSeeD [336]	**15.0**	**33.2**	116.2
Ours	14.1	25.6	28.7

Table 5.3: Ablation study of our method on applying prompt engineering to improve open-vocabulary CIS task on the COD10K-v3 dataset.

Prompt	AP	AP50	AP75
✗	22.8	43.1	22.1
✓	23.4 +0.6	43.8 +0.7	22.6 +0.5

times fewer parameters than OpenSeeD, while scarifying less than 1% and 8% of the overall AP on ADE20K and Cityscapes, respectively.

5.4.4 Ablation Studies

Our ablation studies were conducted to validate different aspects of our method, particularly into the impact of prompt engineering on open-vocabulary CIS and into the modules developed in our method to make it specialised to CIS.

Prompt Engineering for Open-vocabulary CIS. For open-vocabulary-based studies, an object category can be specified by multiple alternative text descriptions. For instance, the "cat" category can be described as "cat", "cats", "kitty", or "kitties". To improve the diversity of open-vocabulary in text prompts, we applied the identical prompt engineering method introduced by [77] to assemble a list of synonyms, subcategories, and plurals for the categories. Given a text prompt, the category is chosen as the one with the highest probability from an ensembling list of multiple alternative queries. We observed that the prompt engineering technique is simple yet effective in improving the segmentation accuracy of our method, as shown in Table 5.3.

CIS-specialised Modules. We developed several modules to make our method specialised to CIS. We refer the reader to Figure 5.2 for a recall of how the modules are configured in our pipeline. To confirm the importance of those modules, we experimented with different variants of our method; each variant is made by altering and omitting a module. We pre-trained the variants on the MS-COCO dataset for 30k iterations and then tested them on the test set of the COD10K-v3 dataset. We present the results of this ablation study in Table 5.4.

Table 5.4: Ablation study on the effectiveness of the proposed modules on COD10K-v3.

Variant	**AP**
no text (text embeddings = 0)	12.2 -7.1
skip the MSFF module (only the last layer of the diffusion U-Net is used)	18.4 -0.9
skip the CIN module (directly use the TVA's output for instance classification)	17.6 -1.7
skip the TVA module (element-wise dot product of mask embedding and text embedding)	18.8 -0.5
Full setting (Ours)	**19.3**

Text embeddings play crucial roles. We investigate by implementing a setting where text embeddings used in the model are set to zeros. A significant drop in the performance, resulting in the lowest AP (12.2), indicates the importance of providing contextual or semantic information that helps to identify camouflage through open-vocabulary text.

MSFF aggregates essential diffusion features. It is the MSFF module that fuses image-guided features learnt by the diffusion model at multiple scales. We found that without the proposed module (directly feeding the last layer of the diffusion U-Net to the mask generator), we can incur a performance loss. Compared with the full setting which fuses all the layers from both the encoder and decoder of the diffusion U-Net, the last layer of the diffusion U-Net apparently carries substantial information for the instance segmentation task.

TVA discerns camouflages against backgrounds via textual guidance. Through TVA, textual and visual features are aggregated alongside instance masks and consolidated against the background via feature weighting. We visualize the impact of the TVA module in Figure 5.6. To validate this module, we simplified its operation by applying an element-wise dot product on the input mask embeddings and text embeddings. We observed that, compared with other modules, the TVA module is less critical, evident by the least performance drop when the simplification is applied to its architecture.

CIN enhances the instance-level representations of camouflages. To validate CIN, we removed it from our pipeline by directly passing the outputs from the TVA module to mask prediction and classification. Without the CIN module, the AP of the pipeline would decrease dramatically (from 19.3 to 17.6).

5.5 Conclusion

This work advances the computer vision research for camouflaged instance segmentation by leveraging text-to-image diffusion and text-image transfer techniques. We aim to raise people's awareness about the possible lack of transfer effectiveness in open-vocabulary segmentation regarding camouflages. Furthermore, we propose a method that effectively integrates textual information learnt from open-vocabulary into the visual domain to enrich the representations of camouflaged objects. We evaluate our method and compare it with existing methods in both CIS and generic open-vocabulary segmentation on benchmark datasets. On the one hand, the method struggles with segmenting occluded objects, and under severe occlusions, a camouflaged object can be over-segmented into non-semantic fragments. Nevertheless, our experimental results show the effectiveness and advantages of our method over existing baselines in both tasks.

Despite proven strengths, the proposed method has limitations. While the learnt knowledge from natural language can be effective to distinguish an object from its background when visual cues are insufficient due to camouflage, it may not be helpful to separate touching/overlapping instances. Additionally, the method struggles with segmenting occluded objects. Under severe occlusions, a camouflaged object can be over segmented into non-semantic fragments, leading to misclassification of the object. Enhancing object representations with background-aware features from open-vocabulary (i.e., by using text prompts including both foreground and background information, e.g., "a lizard is on a tree") may help to address the aforementioned issues. We consider this research direction as our future work.

While open-vocabulary recognition has recently attracted considerable attention in the computer vision community, to the best of our knowledge, our work is the first to provide a framework for localising camouflaged object instances based on open-vocabulary. We believe the proposed framework will open an avenue for new research and developments in surveillance, wildlife monitoring, and military reconnaissance. Nevertheless, it is always challenging to learn general knowledge across a diverse array of concealed targets from open-vocabulary.

CHAPTER 6

VIDEO DATASET FOR CAMOUFLAGE ANIMAL UNDERSTANDING

6.1 Introduction

The continuously evolving neural networks (*e.g*. CNNs [94] and ViTs [55]) provide a powerful and efficient tool to perform visual understanding based on the captured imagery/videos. Enhancing both *data* and *algorithm* has achieved promising success and progress. Large-scale datasets (*e.g*. COCO [165], ADE20K [356] and Object365 [241]) with supervisions serve as the essential stimulus to foster various powerful visual perception algorithms [310] and benchmark various algorithms for revealing further research directions. Most existing datasets mainly contain our everyday objects (*e.g*. 80 categories in COCO). This work focuses on the camouflaged animals, which are currently less concern and explored.

Monitoring and understanding camouflaged animals is crucial for biodiversity conservation [224, 250], as it helps protect species that are otherwise difficult to detect and at risk of unnoticed population declines. Furthermore, studying camouflaged animals provides insights into evolutionary biology and adaptation mechanisms, enriching our scientific understanding of natural selection.

However, unlike everyday objects, collecting imagery/videos of camouflaged animals is more challenging, and further annotation procedures usually involve domain experts. Segmentation, generating precise masks for objects' interest, is the fundamental task in computer vision. Camouflaged animal segmentation helps accurately identify and isolate these animals from their backgrounds in images, facilitating detailed study and analysis. The yielded masks aid in gathering precise data on their behavior, habitat, and population dynamics, enhancing our overall understanding of their ecology. There are several efforts [310, 35, 142, 278] achieved to perform the camouflaged animal segmentation. Specifically, camouflage is a powerful biological mechanism for avoiding detection and identification, making it more challenging to perform precise segmentation.

Various datasets (*e.g*. CAMO-COCO [147], COD10K [64], CAM-LDR [179], S-COD [95]) have been collected for both image-level camouflaged animal segmentation. However, the image-level

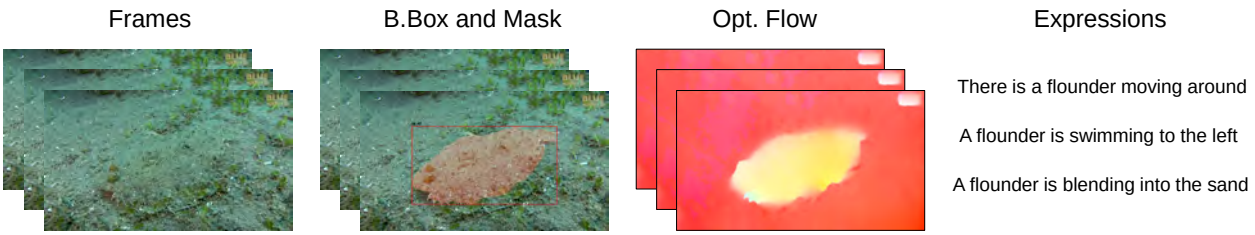


Figure 6.1: Example from our proposed **CamoVid60K** dataset with bounding box, mask, coarse optical flow, and expressions.

camouflaged animal segmentation cannot well satisfy biological monitoring and surveying purposes, where the activity and behavior [317] should be recorded. The MoCA dataset [143] is the most extensive compilation of videos featuring camouflaged objects, yet it only provides detection labels. We argue that only the bounding box annotations cannot well delineate the camouflaged animals, especially those with irregular boundaries, poses, and patterns (*e.g.* the transparent fins of the fish). Furthermore, despite the shifting from image to video, the data annotations remain insufficient in both volume and accuracy for developing a reliable video understanding model capable of effectively handling complex camouflaged situations.

To fill this gap and perform camouflaged animal video understanding (**CAVU**) in real-world scenarios, we present **CamoVid60K**, a comprehensive video dataset dedicated to the understanding of camouflaged animals. It comprises **218** videos with **62,774** finely annotated frames, covering **70** animal categories. Table 6.1 compares our proposed dataset with previous ones, showing that **CamoVid60K** *surpasses* all previous datasets in terms of the number of videos/frames and species included. Unlike previous datasets that annotated every fifth frame, our dataset offers annotations for *every single frame*. Additionally, we provide *a wider variety of annotation types* (animal categories, bounding box, annotated mask, coarse optical flow, expression), making it a more effective benchmark for CAVU tasks. Our dataset supports *a broad range of downstream tasks* as shown in Figure 6.1, including classification, detection, segmentation (semantic, referring, motion), and optical flow estimation, *etc*.

We also propose baselines for each task and corresponding benchmarks to explore the boundary of these advanced algorithms to perform robust and precise video understanding. Our **CamoVid60K** stands as a novel and important testing set for both computer vision, animal conservation, and wildlife animal research communities.

Table 6.1: Comparison with existing video animal datasets. Class.: Classification Label, B.Box: Bounding Box, Motion: Motion of Animal, Coarse OF: Coarse Optical Flow, Expres.: Expression. ***Note that***, MVK [267] dataset mostly consists of normal marine animals with only some camouflaged animals. The frequency of annotations refers to how often each frame is annotated. For instance, MoCA-Mask provides annotations for **every five frames**, resulting in 4,691 annotated frames. In contrast, our CamoVid60K dataset offers a significantly larger volume of data with more frequent annotations and a wider variety of annotation types.

Dataset	Venue	# videos / frames	# species	Frequency	Class.	B.Box	Mask	Motion	Coarse OF	Expres.
CAD [214]	ECCV'16	9 / 839	6	every 5 frames	✓		✓			
MoCA [143]	ACCV'20	141 / 37,250	67	**every frames**	✓	✓		✓		
MoCA-Mask [35]	CVPR'22	87 / 22,939	44	every 5 frames	✓		✓			
MVK [267]	MMM'23	1379 / 992,880	-	every 30 frames	✓					✓
CamoVid60K	**-**	**218 / 62,774**	**70**	**every frames**	✓	✓	✓	✓	✓	✓

Our main contributions are summarized as follows:

- We present a **large-scale**, **comprehensive** video dataset dedicated for camouflaged animal understanding, with **a significantly larger** data and annotation types than the existing datasets.
- We propose a **simple pipeline** for camouflaged animal detection and segmentation with comparable performance.
- We **benchmark various** camouflaged animal understanding tasks, including image classification, object detection, and motion segmentation based on several state-of-the-art models.

6.2 Related Works

6.2.1 Camouflaged Scene Understanding

Camouflaged scene understanding (CSU) is a hot computer vision topic aiming to learn discriminative features that can be used to discern camouflaged target objects from their surroundings [66]. CSU tasks can be divided into image-level and video-level categories. Image-level CSU tasks include five main types: camouflaged object counting [254], camouflaged object localization [180, 179], camouflaged object segmentation [64, 113, 89], camouflaged instance ranking [180, 179], and camouflaged instance segmentation [210, 146]. These tasks can be further categorized based on their semantic focus: object-level and instance-level. Object-level tasks focus on identifying objects, while instance-level tasks aim to differentiate various entities. Additionally, camouflaged object counting is considered a sparse prediction task due to its nature, while the other tasks are classified as dense prediction tasks.

In addition, CSU video-level task includes video camouflaged object segmentation [114, 305, 35] and video camouflaged object detection [143, 141, 317, 310, 193, 135]. Overall, the progress of video-level CSU has been somewhat slower than image-level CSU, primarily because the process of collecting and labeling video data is labor-intensive and time-consuming.

6.2.2 Video Camouflaged Object Detection and Segmentation

We review two kinds of perception tasks for camouflaged animal videos: detection [143, 141, 317, 310, 193, 135] and segmentation [114, 305, 35, 142]. The former video camouflaged object detection (VCOD) yields BBOX sequences for the camouflaged animals, while the latter video camouflaged object segmentation (VCOS) generates dense pixel-level masks. MoCA [143] proposed the first large-scale moving camouflaged animals video dataset with BBOX annotations and additional optical flows to boost the detection of camouflaged animals. Further work [141] incorporated visual appearance from a static scene as additional clues to promote the ability of the model to detect camouflaged animals. However, the BBOX annotations could not accurately describe camouflaged objects' pose, appearance, and patterns. To address this issues, Xie *et al*. [305] proposed a novel pixel-trajectory RNN to cluster fore-ground pixels and generate dense segmentation masks for object discovery in videos. MoCA-Mask [35] proposed the first large-scale dataset and benchmark with pixel-level handcrafted ground truth masks for camouflaged animal videos. However, MoCA-Mask provides bounding boxes and pixel-wise masks for **only every fifth frame**, totaling just 4,691 frames, which is insufficient for deep learning approaches. In contrast, our dataset offers annotations for **every frame**, resulting in 62,774 annotated frames (**13 times larger**). This substantial increase can significantly enhance the performance of various downstream tasks. Our dataset and benchmark pave the way for future exploration and a deeper understanding of camouflaged animal analysis.

6.3 CamoVid60K Dataset

Collecting video datasets for camouflaged animals is quite challenging, even without focusing on long-form videos. This is because manually collecting, observing, and annotating videos with several annotation types is labor-intensive, time-consuming, and expensive. In addition to the costs, ensuring visual data diversity and high-quality annotations adds to the difficulty. This section proposes a staged data collection and processing pipeline in Figure 6.2.

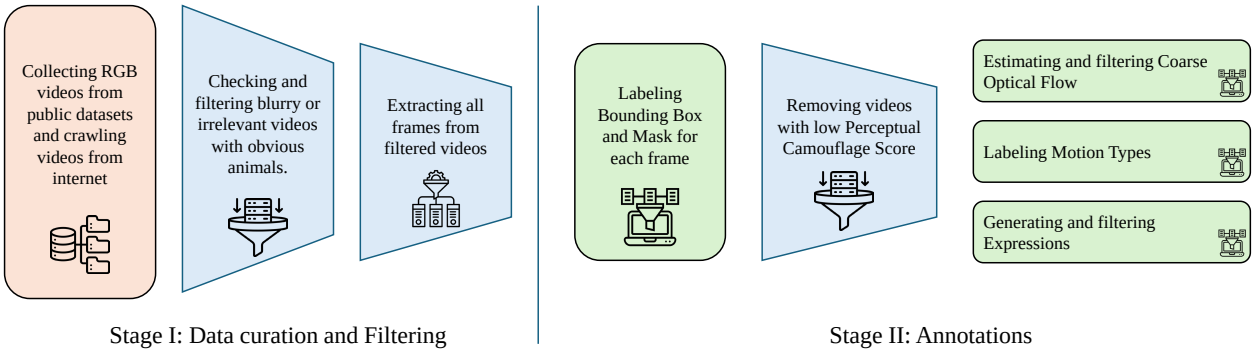


Figure 6.2: **CamoVid60K** data pipeline. Stage I includes data curation, filtering irrelevant videos, and extracting all frames. Stage II includes data annotation, generation, and filtering.

6.3.1 Data Construction and Processing

Data sources. We built our dataset from previous datasets (Table 6.1) and crawled videos from the internet to cover various camouflaged animals. To crawl videos from the internet, we curated a list of animals' names that potentially have camouflage abilities. Then, we created a template for searching and downloading videos *'video of camouflage + animals' name'*. Combining with videos from the above datasets, we collected 1,929 videos in total.

Pre-Processing. We manually checked and filtered blurry, irrelevant videos with obvious animals. Next, we extracted every frame (instead of every five frames proposed in existing datasets, see Table 6.1) of each video before annotating them. At the end, our dataset comprises **218** videos with **62,774** frames of **70** animals.

BBOX and Mask Annotation. We utilized annotation tool from [352] which heavily based on Segment Anything Model (SAM) [133] for mask initialization and bounding box and XMem [33] for mask and bounding box propagation. Then, we manually check and refine every frame to provide accurate bounding boxes and segmentation masks.

Annotation Filtering. We adopt the perceptual camouflage score (S_p) from [142] to quantify the effectiveness of animals' camouflage, *i.e.* how successfully an animal blends into its background. Based on the perceptual camouflage score, we will keep the videos that are higher than the threshold ($S_p > 0.5$). In detail, we explain how to adopt the perceptual camouflage score S_p as follows:

$$S_p = (1 - \alpha)S_R + \alpha S_B \tag{6.1}$$

Algorithm 2: Optical Flow Computation and Filtering

Input: Sequence of frames

Output: Sequence of computed optical flows

1: **for** each pair of frames (i, j) **do**
2: Computing all pairwise optical flows using RAFT [265]
3: Computing DINO features [204] for each frame
4: Filtering flows using cycle consistency and appearance consistency check
5: Applying chain cycle consistent correspondences to create denser correspondences
6: **end for**

where S_R, S_B, α are the reconstruction fidelity score, the boundary score, and the weighting parameter, respectively.

In detail, given an image I and segmentation mask m_S, the reconstruction fidelity score S_R is computed by assessing the difference value between the foreground region and its reconstruction. Specifically, it counts the number of foreground pixels ($I_{fg} = I \odot \text{erode}(m_s)$) that have been successfully reconstructed from the background ($I_{bg} = I \odot (1 - \text{dilate}(m_s))$):

$$S_R(I, m_s) = \frac{1}{N_{fg}} \sum_{(i,j) \in I_{fg}} R(i, j) \tag{6.2}$$

$$R(i, j) = \begin{cases} 1 & \text{if } \|I_{fg} - \Psi_{I_{bg}}(I_{fg})\|_2 < \lambda \|I_{fg}\|_2 \\ 0 & \text{otherwise} \end{cases} \tag{6.3}$$

where $\Psi_{I_{bg}}(.)$ denotes the reconstruction operation, $N_{fg} = |\text{erode}(m_s)|$ is the total number of pixels in the foreground region, and λ is a threshold.

Then, the boundary visibility score aims to measure the animal's boundary properties (or contour visibility) by penalizing the boundary pixels that are predicted as contour in both images' contour (C) and the ground truth animal's contour (C_{gt}) with F1 metric:

$$S_B(I, m_s) = 1 - F1(m_b \odot C_{gt},\ m_b \odot C) \tag{6.4}$$

where $m_b = \text{dilate}(m_s) - \text{erode}(m_s)$.

We used the same values for parameters as in [142], such as $\alpha = 0.35, \lambda = 0.2$.

Note that, due to the characteristics of camouflaged animals and video's resolution, some frames

or some videos will contain errors/mislabelled at the boundary of animals and background. We will keep improving the quality of the mask annotations and provide rotated bounding boxes (RBbox) in the next version. RBbox excels in traditional axis-aligned bounding boxes in three main areas: better localization (accurate fit for elongated and rotated objects), reduced overlap, and improved isolation of objects (capturing the proper aspect ratio and containing fewer background pixels).

Coarse Optical Flow Annotation. Previous optical flow datasets, *e.g*. Flying Chair [54], KITTI [190], Sintel [13] utilized either simulation software or real images with other heavy sensors information (depth, LiDAR, *etc*.) and algorithms to create optical flow ground-truth. It is time-consuming and requires extreme effort. In addition, with the development of deep learning techniques, many methods [265, 286] can produce accurate estimated optical flow. Therefore, we utilized these methods for our coarse optical flow annotation with the pseudo algorithm shown in Algorithm 2.

Note that, even though our processing pipeline for optical flow annotation will produce accurate and dense optical flow, it is still **estimated** optical flow, so it is reasonable and capable of use as *additional input* to boost performance for other tasks such as motion segmentation task. It is **not recommended** to use it as ground truth for evaluation.

Motion Annotation. Following [143], we manually labeled our dataset by their types of motion, as shown below. Based on the motion types, We can further annotate the camouflage methods of animals, which we plan to provide in the next version.

- *Locomotion*: when the animal has movement(s) that significantly changes its location.
- *Deformation*: when the animal engages in a more delicate movement that only changes its pose while remaining in the same location.
- *Still*: when the animal remains still.

Expression Annotation. We first utilized GPT-4V [264] to create a concise description within 30 words that accurately represents the target object for every frame. However, we found that the captions of aquatic animals are less accurate; therefore, we utilized MarineGPT [353], a first vision-language model specially designed for the marine domain for aquatic animals. After the initial annotation, we verified and refined all annotations and chose the best three captions for each video sequence. Objects that could not be localized using language expressions were removed.

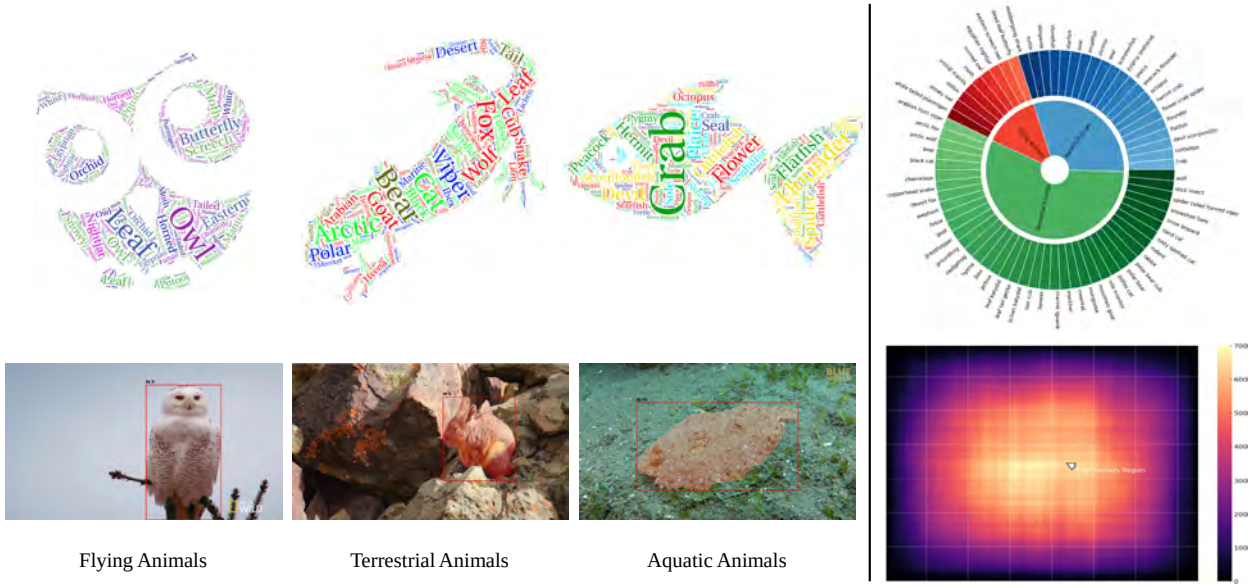


Figure 6.3: **Left-Top:** Word cloud of category distribution of camouflaged animals. **Right-Top:** Taxonomic structure of our dataset by their biology-inspired hierarchical categorization. It encompasses various animals, spanning 70 categories across flying, terrestrial, and aquatic groups. **Left-Bottom:** Some examples with different animals' positions. **Right-Bottom:** Spatial distribution of animals' position based on bounding box. It reveals that annotations are more densely concentrated in the central region, while there is a comparatively lower density of annotations towards the edges.

6.3.2 Dataset Specifications and Statistics

Data Organization. As shown in Figure 6.4, we split our dataset based on displacement into two subsets: small displacement (every single frame) and large displacement (every fifth frame). This division is beneficial for evaluating motion segmentation methods, as it provides a robust framework for analyzing algorithms' performance under varying motion and displacement conditions. Each subset will include training and testing sets with images, pre-computed optical flows, and annotations. We name every image as follows: 'SuperClass-

```
CamoVid60k
|
├── Small_Displacement
|   ├── Train_Set (168 videos)
|   |   ├── Aquatic-Crab-3
|   |   |   ├── Images (Aquatic-Crab-3-Loco-0001.jpg)
|   |   |   ├── Flows (Aquatic-Crab-3-Loco-0001.flo)
|   |   |   └── Annotations
|   |   |   |   ├── annotations.json (bounding boxes and masks)
|   |   |   |   └── exp_annotations.json (expressions)
|   |   ├── Terrestrial-Wolf-1
|   |   └── Flying-Owl-1
|   └── Test_Set (50 videos)
|
└── Large_Displacement
    ├── Train_Set
    └── Test_Set
```

Figure 6.4: Data organization of our dataset.

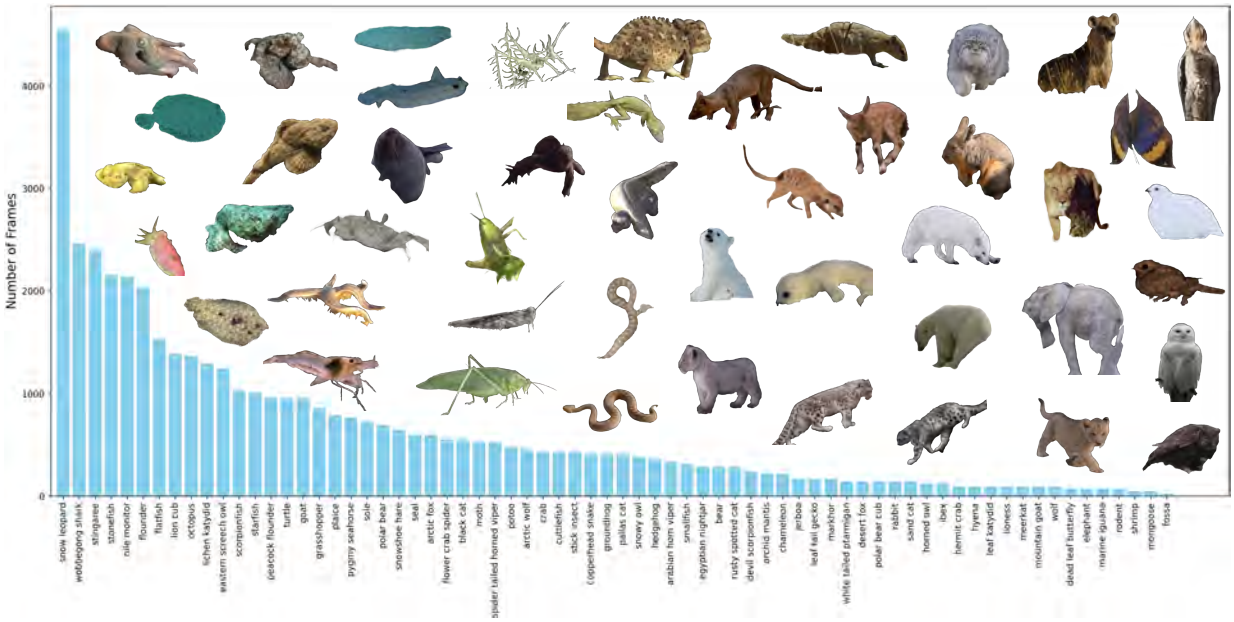


Figure 6.5: Category distribution (100 to 4,500 frames) and visual examples (extracted animal masks) of our dataset. The variety ensures diversity of camouflaged animals, allowing for comprehensive evaluation across various scenarios. We will keep adding more data to balance the distribution.

SubClass-SubNumber-MotionType-FrameNumber.‘ This systematic naming convention ensures clarity and ease of reference within the dataset.

Category Diversity. The distributions of camouflaged animals by the biology-inspired hierarchical categorization within three super groups are visually represented through word clouds in Figure 6.3-Top and Figure 6.5. Additionally, Figure 6.3-Bottom showcases some examples with different animals' positions and the total sum of normalized bounding boxes across the entire dataset.

Evaluation Protocol. Our dataset supports a broad range of downstream tasks. Therefore, we will evaluate each task using different metrics.

- *Motion Segmentation:* we adopt the same metrics as in [35] to assess the pixel-wise masks: Mean Absolute Error (M), Enhanced-alignment measure (E_ϕ) [63], Structure-measure (S_α) [62], Weighted F-measure (F_β^w) [185], mean Intersection Over Union (mIoU), mean Dice (mDic).
- *Object Detection:* we use the mean Average Precision (mAP).
- *Image Classification:* we use the mean Accuracy (mAcc).
- *Referring Segmentation:* we utilize the mIoU, region similarity J and contour accuracy F, and their average J&F for video object segmentation.

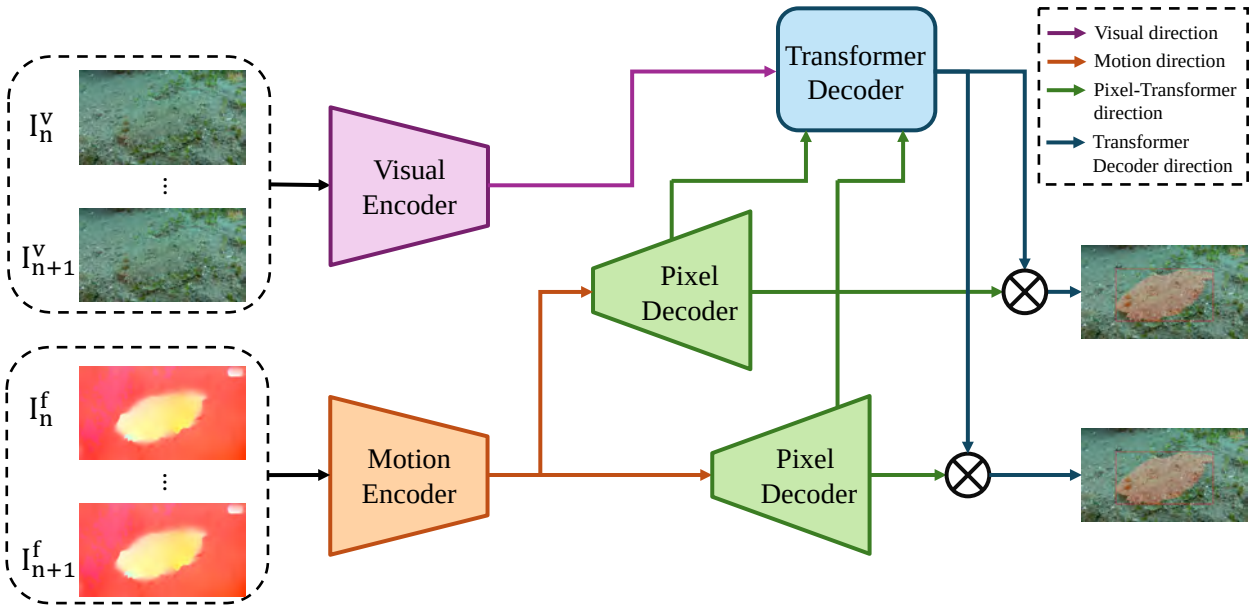


Figure 6.6: Our simple pipeline takes a sequence of images/video and the associated optical flow as input. They are fed into separated encoders for feature extraction. Then, the motion features with spatial and temporal positional encoding are passed to Pixel Decoders to produce a set of enriched motion features. Next, the Transformer Decoder takes the visual features and enriched motion features to produce mask embedding for the moving object and bounding box.

6.4 A simple pipeline to discern camouflaged animals

After constructing the dataset, we propose a simple pipeline based on Mask2Former architecture [29] for both object detection and motion segmentation tasks. As shown in Figure 6.6, our proposed simple pipeline processes a sequence of images or videos by employing any off-the-shelf flow estimation methods. In our case, we directly take the refined optical flow in our dataset instead of utilizing the RAFT method [265] to estimate raw optical flow as [142]. The images and associated estimated flows are passed into two separated encoders for feature extraction. Subsequently, each timestamp's image and flow features are aggregated before going through the decoder to predict the segmentation mask.

Visual Encoder. We adopt the SINet-v2 [64] architecture that takes RGB sequence as input $I^v = \{I_1^v, I_2^v, \ldots, I_n^v\} \in \mathbb{R}^{n \times d_v \times h \times w}$, where n is the number of frames, d_v is the dimension of frame, $h\&w$ are the height & width and outputs visual features $\{f_1^v, f_2^v, \ldots, f_n^v\} = \Phi_{visual}(I^v)$.

Motion Encoder. Inspired by the motion segmentation architecture [141], we use light-weight convNet that takes as input a sequence of optical flows $I^f = \{I_1^f, I_2^f, \ldots, I_n^f\} \in \mathbb{R}^{n \times d_f \times h \times w}$, where

d_f is the dimension of flow field and output motion features $\{f_1^m, f_2^m, \ldots, f_n^m\} = \Phi_{motion}(I^m)$. Then, concatenating motion features with learned spatial and temporal positional encodings to output a set of enriched motion features.

Decoder. We adopt Mask2Former [29] architecture, which includes Transformer and Pixel Decoders. The Transformer decoder combines a trainable query for mask embedding with the results of the motion encoder and visual features. Like Mask2Former, this query focuses on multi-scale motion features and visual features, resulting in mask embedding for the moving object. In addition, similar to the pixel decoder in Mask2Former, a ConvNet decoder with low computational complexity utilizes skip-connections to generate high-resolution segmentation masks and bounding boxes from the motion features and mask embedding.

Training and Loss. To optimize our pipeline, we utilized the L1 loss for the bounding box regression, cross-entropy for the confidence score, and binary cross entropy (BCE) loss for motion segmentation. The total loss for the training of our pipeline is finally defined as follows:

$$L = L_{bce} + L_{L1} + L_{ce}, \tag{6.5}$$

6.5 Experiments

This section introduces the baselines and details of the training for each task. We thoroughly analyze each task in our experiments and discuss each method's effectiveness, including ours.

6.5.1 Baselines

For motion segmentation task, we selected recent SOTAs to compare, including two frame-based methods (PraNet [67], SINet-v2 [64]) and two video-based methods (MG [317], SLT-Net [35]). For a fair comparison, we utilize the implementations provided by the authors and train all methods using the same training set.

For object detection task, we compare with four well-known detection methods, such as Faster-RCNN [227], DETR [16], DINO [335]. We followed the so-called 1× setting (12-epoch setting) for training and used the same ResNet50 [94] as the backbone for all methods.

Table 6.2: Quantitative results of motion segmentation on CamoVid60K dataset. Our model consistently achieves better performance than other competitors on all metrics.

Methods		$S_\alpha \uparrow$	$F_\beta^w \uparrow$	$E_\phi \uparrow$	$M \downarrow$	mDic ↑	mIoU ↑
Image	PraNet [67]	0.526	0.161	0.547	0.045	0.198	0.144
	SINet-v2 [64]	0.529	0.166	0.553	0.042	0.206	0.149
Video	MG [317]	0.522	0.153	0.541	0.043	0.197	0.141
	SLT-Net [35]	**0.576**	**0.253**	**0.591**	**0.039**	0.268	0.249
	Ours	0.566	0.249	0.589	0.041	**0.270**	**0.252**

Table 6.3: Quantitative results of object detection on our CamoVid60K dataset.

Methods	AP ↑
F-RCNN [227]	28.71
DETR [16]	37.56
DINO [335]	**39.84**
Ours	38.39

For zero-shot image classification task, we tested recent three methods, including CLIP [221], UniCL [321] and K-LITE [242]. We used the Swin-T model for both UniCL and K-LITE (pre-trained on ImageNet-21K dataset [46]) and the ViT-B/32 pre-trained model from OpenAI CLIP.

All methods are trained and tested on the same NVIDIA 3090 GPU, except the pre-trained models in the zero-shot image classification task.

6.5.2 Benchmarks and Discussions

Comparison with image-based and video-based motion segmentation methods. We report the performance of our method with other methods in Table 6.2. Compared to image-based approaches, our method demonstrates significantly superior performance thanks to the incorporation of temporal information. When evaluated against video-based approaches, our method also surpasses MG [317], which relies solely on estimated optical flows as input. However, compared to the recent state-of-the-art method SLT-Net [35], our method performs better on certain metrics. This is because SLT-Net excels at modeling both short-term dynamics and long-term temporal consistency from videos, allowing for joint optimization of motion and camouflaged object segmentation through a single optimization target.

Comparison with object detection methods. As shown in Table 6.3, the proposed model demonstrates performance comparable to other specialized methods, owing to its dual capability in object detection and motion segmentation. Specifically, our method significantly outperforms CNN-like methods. This advantage stems from dual optimizations in the detection and segmentation streams, along with the integration of additional optical flow information. However, when compared to

Table 6.4: Ablation study on the impact of flow information on our method.

	no OF	raw OF	refined OF
mIoU	28.34	32.16	**32.81**

Table 6.5: Zero-shot Image Classification performance on our CamoVid60K dataset.

	CLIP [221]	UniCL [321]	K-LITE [242]
mAcc	30.06	30.89	**31.44**

DETR-like methods, our approach shows mixed results. It surpasses the standard DETR model [16] yet falls short of DINO, an advanced variant of DETR. DINO [335] enhances performance through several innovative techniques: it employs contrastive denoising training to refine one-to-one matching, a mixed query selection method to initialize the queries better, and a 'look forward twice' method that utilizes gradients from subsequent layers to adjust parameters more accurately.

Additional Analysis. As shown in Table 6.4, optical flow plays a crucial role in the motion segmentation of camouflaged animals because optical flow can detect subtle movements by analyzing the motion vectors between frames, distinguishing moving animals from static backgrounds, which is particularly useful in identifying the slight movements of camouflaged animals.

State-of-the-art methods, including foundation models (trained on large datasets), struggle with zero-shot image classification of camouflaged animals, as shown in Table 6.5. This is due to their subtle and complex patterns, lack of specific training data, and difficulty in generalizing across different backgrounds and lighting conditions. Improving these methods involves curating specialized training data (or fine-tuning on our dataset), using enhanced techniques like data augmentation, few-shot learning, and developing context-aware models.

6.6 Conclusion

In this work, we introduced a large-scale video dataset for camouflaged animal understanding, named **CamoVid60K**, to foster further research on animals. This dataset provides a considerable benchmark for camouflaged animal video understanding tasks, enabling the evaluation of various algorithms and methods. We also plan to scale up our dataset and utilize it to build a foundational model for studying camouflaged animals.

Limitations and Future Works. As mentioned in Section 6.3, the annotation quality in some cases is suboptimal. We plan to enhance these annotations and also provide more types of annotations

in the future. Additionally, our current pipeline requires images and pre-computed optical flow as inputs, which restricts the inclusion of new data to our pipeline due to the necessity of pre-computed optical flow. To address this limitation, we will propose a learnable module to estimate the implicit optical flow field.

New Benchmark. The **CamoVid60K** is a diverse and comprehensive benchmark curated from publicly accessible datasets and the internet to enhance the assessment and exploration of camouflaged animal understanding. It includes various camouflaged animals across various environments, providing a robust framework for testing and developing new models.

Impact on Animal Study. By providing detailed and varied data on camouflaged animals, the **CamoVid60K** dataset significantly contributes to studying animal behavior, ecology, and evolution. Researchers can utilize this dataset to explore how different species utilize camouflage in their natural habitats, leading to deeper insights into predator-prey interactions and survival strategies. Furthermore, this dataset can aid conservation efforts by improving the detection and monitoring of endangered species in their natural environments.

Broader Impact. The study of camouflaged objects has several important applications, such as identifying and safeguarding rare animal species, preventing wildlife trafficking, detecting medical conditions like polyps or lung infections, and aiding in search-and-rescue operations. Our dataset deliberately excludes any military or sensitive scenes, ensuring its focus remains on benign and beneficial applications. Besides the significant applications mentioned, our work advances the understanding of video content in the presence of distorted motion information, contributing to the broader field of video analysis and computer vision.

Licenses. We built our dataset from previous datasets and crawled online videos. Therefore, we will follow their Term of Use or Licenses (MoCA, MVK) for our dataset, which is CC-BY-4.0. The copyright remains with the original owners of the video. In addition, anyone shall use the dataset only for non-commercial research and educational purposes.

CHAPTER 7

CONCLUSION AND FUTURE WORKS

In conclusion, this thesis addresses critical challenges in computer vision and graphics, with a particular focus on transparent objects, dynamic human and animal movement, and camouflaged animals for both indoor, outdoor, terrestrial, and underwater (marine) environments.

In Chapter 2, we introduced Test-time augmentation for enhancing downstream tasks in both indoor and outdoor environments, encompassing autonomous driving scenarios and utilizing synthetic as well as real-world data types. Moving on to Chapter 3, we delved into the joint reconstruction of objects (human and animals) and estimation of flow for dynamic point clouds. Then, in Chapter 4, we explored the utilization of Boundary and Reflection cues for Transparent Objects Segmentation. Next, we leveraging text-to-image diffusion to address the problem of open-vocabulary for camouflaged instance segmentation in Chapter 5. Later in Chapter 6, to fill the gap of lacking camouflage animal video dataset and also perform camouflaged animal video understanding in real-world scenarios, we present a comprehensive video dataset dedicated to the understanding of camouflaged animals.

The proposed methodologies and datasets showcase exceptional performance improvements and efficiency gains, significantly advancing the field. By combining cutting-edge techniques with traditional approaches, this research paves the way for a new era of innovation and progress in computer vision and graphics.

Although the methods proposed in this thesis are novel and have achieved promising results in their specific tasks, there are several interesting research topics related to transparent, dynamic objects and camouflaged animals. Potential future works include:

- **Enhanced Transparent Object Detection and Tracking:** Developing more robust algorithms that can accurately detect and track transparent objects in real-time, even in complex and cluttered environments.
- **Multi-Modal Sensor Fusion:** Investigating the integration of data from various sensors, such as RGB cameras, depth sensors, and thermal cameras, to improve the accuracy and reliability of transparent object detection and segmentation.

- **3D Reconstruction of Transparent Objects:** Creating innovative methods for reconstructing transparent objects in 3D, which has applications in fields such as robotic manipulation in various industrial and domestic settings.
- **Dynamic Environment Adaptation:** Extending the research to handle dynamic and changing environments, ensuring the developed techniques remain effective in real-world scenarios with moving objects and varying lighting conditions.
- **Transfer Learning for Rare Object Detection:** Exploring transfer learning techniques to improve the detection and segmentation of rare or less frequently occurring camouflaged animals to study and protect wildlife, particularly camouflaged animals in their natural habitats.
- **Augmented Reality and Virtual Reality:** Utilizing the advancements in transparent object detection, tracking and reconstruction to enhance the realism and interactivity of augmented reality (AR) and virtual reality (VR) experiences.

By addressing these future research directions, we can continue to push the boundaries of what is possible in computer vision and graphics, ultimately leading to more intelligent and capable systems.

REFERENCES

[1] M. Alexa, J. Behr, D. Cohen-Or, S. Fleishman, D. Levin, and C. T. Silva, "Computing and rendering point set surfaces," *IEEE TVCG*, 2003. 12

[2] T. Alldieck, M. Magnor, W. Xu, C. Theobalt, and G. Pons-Moll, "Video based reconstruction of 3D people models," in *CVPR*, 2018. 30

[3] I. Armeni, A. Sax, A. R. Zamir, and S. Savarese, "Joint 2d-3d-semantic data for indoor scene understanding," *ArXiv e-prints*, 2017. 66, 81

[4] I. Armeni, O. Sener, A. R. Zamir, H. Jiang, I. Brilakis, M. Fischer, and S. Savarese, "3d semantic parsing of large-scale indoor spaces," in *CVPR*, 2016, pp. 1534–1543. 10, 19, 21, 22, 24

[5] M. Armin, H. Kim, J.-Y. Guillemaut, and A. Hilton, "Temporally coherent 4d reconstruction of complex dynamic scenes," in *CVPR*, 2016. 30, 34

[6] M. S. Ayhan and P. Berens, "Test-time data augmentation for estimation of heteroscedastic aleatoric uncertainty in deep neural networks," in *International conference on Medical Imaging with Deep Learning*, 2018. 13

[7] D. Baranchuk, A. Voynov, I. Rubachev, V. Khrulkov, and A. Babenko, "Label-efficient semantic segmentation with diffusion models," in *Proceedings of the International Conference on Learning Representations*, 2022. 91

[8] J. Behley, M. Garbade, A. Milioto, J. Quenzel, S. Behnke, C. Stachniss, and J. Gall, "Semantickitti: A dataset for semantic scene understanding of lidar sequences," in *ICCV*, 2019. 10, 19, 20, 21, 22, 23

[9] H. Ben-Hamu, H. Maron, I. Kezurer, G. Avineri, and Y. Lipman, "Multi-chart generative surface modeling," *ACM Transactions on Graphics (TOG)*, vol. 37, no. 6, pp. 1–15, 2018. 11

[10] S. Biasotti, A. Cerri, A. Bronstein, and M. Bronstein, "Recent trends, applications, and perspectives in 3D shape similarity assessment," *Comput. Graph. Forum*, 2016. 35

[11] F. Bogo, J. Romero, G. Pons-Moll, and M. J. Black, "Dynamic FAUST: Registering human bodies in motion," in *CVPR*, 2017. 42

[12] D. Bolya, C. Zhou, F. Xiao, and Y. J. Lee, "Yolact: Real-time instance segmentation," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2019, pp. 9157–9166. 103

[13] D. J. Butler, J. Wulff, G. B. Stanley, and M. J. Black, "A naturalistic open source movie for optical flow evaluation," in *ECCV*, 2012. 115

[14] Z. Cai and V. Nuno, "Cascade r-cnn: High quality object detection and instance segmentation," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 43, no. 5, pp. 1483–1498, 2019. 103

[15] Y. Cao, Z. Zhang, E. Xie, Q. Hou, K. Zhao, X. Luo, and J. Tuo, "Fakemix augmentation improves transparent object detection," *arXiv preprint arXiv:2103.13279*, 2021. 58

[16] N. Carion, F. Massa, G. Synnaeve, N. Usunier, A. Kirillov, and S. Zagoruyko, "End-to-end object detection with transformers," in *ECCV*, 2020. 59, 119, 120, 121

[17] R. Chabra, J. E. Lenssen, E. Ilg, T. Schmidt, J. Straub, S. Lovegrove, and R. Newcombe, "Deep local shapes: Learning local sdf priors for detailed 3d reconstruction," in *ECCV*, 2020, pp. 608–625. 11

[18] R. Chabra, J. E. Lenssen, E. Ilg, T. Schmidt, J. Straub, S. Lovegrove, and R. Newcombe, "Deep local shapes: Learning local SDF priors for detailed 3D reconstruction," in *ECCV*, 2020. 33

[19] A. X. Chang, T. A. Funkhouser, L. J. Guibas, P. Hanrahan, Q.-X. Huang, Z. Li, S. Savarese, M. Savva, S. Song, H. Su, J. Xiao, L. Yi, and F. Yu, "Shapenet: An information-rich 3d model repository," *arXiv preprint arXiv:1512.03012*, 2015. 10, 19, 20, 21, 24, 25, 26

[20] P. Chao, C.-Y. Kao, Y.-S. Ruan, C.-H. Huang, and Y.-L. Lin, "HarDNet: A low memory traffic network," in *ICCV*, 2019. 74

[21] H. Chen, K. Sun, Z. Tian, C. Shen, Y. Huang, and Y. Yan, "Blendmask: Top-down meets bottom-up for instance segmentation," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020, pp. 8573–8581. 103

[22] J. Chen, Z. Yang, and L. Zhang, "Semantic segment anything," https://github.com/fudan-zvg/Semantic-Segment-Anything, 2023. 56, 84

[23] K. Chen, J. Pang, J. Wang, Y. Xiong, X. Li, S. Sun, W. Feng, Z. Liu, J. Shi, W. Ouyang *et al.*, "Hybrid task cascade for instance segmentation," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019, pp. 4974–4983. 103

[24] L.-C. Chen, Y. Zhu, G. Papandreou, F. Schroff, and H. Adam, "Encoder-decoder with atrous separable convolution for semantic image segmentation," in *ECCV*, 2018. 56, 62, 71, 72, 74

[25] R. T. Q. Chen, Y. Rubanova, J. Bettencourt, and D. K. Duvenaud, "Neural ordinary differential equations," in *NeurIPS*, 2018. 35

[26] Y. Chen, H. Huang, T.-A. Vu, K. C. Shum, and S.-K. Yeung, "Stylecity: Large-scale 3d urban scenes stylization," in *Proceedings of the European Conference on Computer Vision*, 2024. 7

[27] Y. Chen, V. T. Hu, E. Gavves, T. Mensink, P. Mettes, P. Yang, and C. G. M. Snoek, "Pointmixup: Augmentation for point clouds," in *ECCV*, 2020. 13

[28] Z. Chen and H. Zhang, "Learning implicit fields for generative shape modeling," in *CVPR*, 2019. 30

[29] B. Cheng, I. Misra, A. G. Schwing, A. Kirillov, and R. Girdhar, "Masked-attention mask transformer for universal image segmentation," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022, pp. 1290–1299. 96, 97, 103, 118, 119

[30] B. Cheng, I. Misra, A. G. Schwing, A. Kirillov, and R. Girdhar, "Masked-attention mask transformer for universal image segmentation," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022, pp. 1280–1289. 100

[31] B. Cheng, A. Schwing, and A. Kirillov, "Per-pixel classification is not all you need for semantic segmentation," *Advances in Neural Information Processing Systems*, vol. 34, pp. 17 864–17 875, 2021. 103

[32] B. Cheng, A. G. Schwing, and A. Kirillov, "Per-pixel classification is not all you need for semantic segmentation," *arXiv*, 2021. 58

[33] H. K. Cheng and A. G. Schwing, "XMem: Long-term video object segmentation with an atkinson-shiffrin memory model," in *ECCV*, 2022. 113

[34] S. Cheng, Z. Leng, E. D. Cubuk, B. Zoph, C. Bai, J. Ngiam, Y. Song, B. Caine, V. Vasudevan, C. Li, Q. V. Le, J. Shlens, and D. Anguelov, "Improving 3d object detection through progressive population based augmentation," in *ECCV*, A. Vedaldi, H. Bischof, T. Brox, and J.-M. Frahm, Eds., 2020. 13

[35] X. Cheng, H. Xiong, D.-P. Fan, Y. Zhong, M. Harandi, T. Drummond, and Z. Ge, "Implicit motion handling for video camouflaged object detection," in *CVPR*, 2022. 109, 111, 112, 117, 119, 120

[36] J. Chibane, T. Alldieck, and G. Pons-Moll, "Implicit functions in feature space for 3D shape reconstruction and completion," in *CVPR*, 2020. 33

[37] J. Choe, C. Park, F. Rameau, J. Park, and I. S. Kweon, "Pointmixer: Mlp-mixer for point cloud understanding," in *ECCV*, 2022. 19

[38] C. B. Choy, D. Xu, J. Gwak, K. Chen, and S. Savarese, "3d-r2n2: A unified approach for single and multi-view 3d object reconstruction," in *ECCV*, 2016, pp. 628–644. 11

[39] X. Chu, Z. Tian, Y. Wang, B. Zhang, H. Ren, X. Wei, H. Xia, and C. Shen, "Twins: Revisiting the design of spatial attention in vision transformers," in *NeurIPS*, 2021. 71, 72

[40] M. Cordts, M. Omran, S. Ramos, T. Rehfeld, M. Enzweiler, R. Benenson, U. Franke, S. Roth, and B. Schiele, "The cityscapes dataset for semantic urban scene understanding," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2016, pp. 3213–3223. 101, 105

[41] H. Coskun, F. Achilles, R. DiPietro, N. Navab, and F. Tombari, "Long short-term memory kalman filters: Recurrent neural estimators for pose regularization," in *ICCV*, 2017. 30

[42] A. Dai, A. X. Chang, M. Savva, M. Halber, T. Funkhouser, and M. Niessner, "Scannet: Richly-annotated 3d reconstructions of indoor scenes," in *CVPR*, 2017, pp. 5828–5839. 10

[43] A. Dai and M. Nießner, "3dmv: Joint 3d-multi-view prediction for 3d semantic scene segmentation," in *ECCV*, 2018, pp. 452–468. 10

[44] A. Dai, C. Ruizhongtai Qi, and M. Nießner, "Shape completion using 3d-encoder-predictor cnns and shape synthesis," in *CVPR*, 2017, pp. 5868–5877. 11

[45] H. de Vries, F. Strub, J. Mary, H. Larochelle, O. Pietquin, and A. C. Courville, "Modulating early visual processing by language," in *NeurIPS*, 2017. 40

[46] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "Imagenet: A large-scale hierarchical image database," in *2009 IEEE Conference on Computer Vision and Pattern Recognition*, 2009, pp. 248–255. 120

[47] K. Desai and J. Johnson, "VirTex: Learning visual representations from textual annotations," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021, pp. 11 162–11 173. 94

[48] P. Dhariwal and A. Q. Nichol, "Diffusion models beat gans on image synthesis," in *Proceedings of the Advances in Neural Information Processing Systems*, 2021, pp. 8780–8794. 91

[49] M. Ding, B. Xiao, N. Codella, P. Luo, J. Wang, and L. Yuan, "Davit: Dual attention vision transformers," in *ECCV*, 2022. 37, 38, 47, 80

[50] Z. Ding, J. Wang, and Z. Tu, "Open-vocabulary universal image segmentation with maskclip," in *Proceedings of the International Conference on Machine Learning*, 2023. 88, 93, 102, 103, 106

[51] Z. Ding, J. Wang, and Z. Tu, "Open-vocabulary universal image segmentation with MaskCLIP," in *Proceedings of the International Conference on Machine Learning*, 2023, pp. 8090–8102. 100, 102

[52] X. Dong, J. Bao, D. Chen, W. Zhang, N. Yu, L. Yuan, D. Chen, and B. Guo, "Cswin transformer: A general vision transformer backbone with cross-shaped windows," in *CVPR*, June 2022. 59

[53] Z. Dong, K. Xu, Y. Yang, H. Bao, W. Xu, and R. W. Lau, "Location-aware single image reflection removal," in *2021 IEEE/CVF International Conference on Computer Vision (ICCV)*, 2021, pp. 4997–5006. 64, 83

[54] A. Dosovitskiy, P. Fischer, E. Ilg, P. Häusser, C. Hazırbaş, V. Golkov, P. v.d. Smagt, D. Cremers, and T. Brox, "FlowNet: Learning optical flow with convolutional networks," in *ICCV*, 2015. 115

[55] A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, S. Gelly *et al.*, "An image is worth 16x16 words: Transformers for image recognition at scale," *arXiv preprint arXiv:2010.11929*, 2020. 109

[56] A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, S. Gelly, J. Uszkoreit, and N. Houlsby, "An image is worth 16x16 words: Transformers for image recognition at scale," *ICLR*, 2021. 58, 86

[57] Y. Du, F. Wei, Z. Zhang, M. Shi, Y. Gao, and G. Li, "Learning to prompt for open-vocabulary object detection with vision-language model," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022, pp. 14 064–14 073. 87, 92

[58] V. Dumoulin, I. Belghazi, B. Poole, O. Mastropietro, A. Lamb, M. Arjovsky, and A. Courville, "Adversarially learned inference," in *ICLR*, 2017. 40

[59] P. Erler, P. Guerrero, S. Ohrhallinger, N. J. Mitra, and M. Wimmer, "Points2Surf: Learning implicit surfaces from point clouds," in *ECCV*, 2020. 33

[60] P. Esser, R. Rombach, and B. Ommer, "Taming transformers for high-resolution image synthesis," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021, pp. 12 873–12 883. 91

[61] D.-P. Fan, M.-M. Cheng, Y. Liu, T. Li, and A. Borji, "Structure-measure: A new way to evaluate foreground maps," in *ICCV*, Oct 2017. 69

[62] D.-P. Fan, M.-M. Cheng, Y. Liu, T. Li, and A. Borji, "Structure-measure: A New Way to Evaluate Foreground Maps," in *ICCV*, 2017. 117

[63] D.-P. Fan, C. Gong, Y. Cao, B. Ren, M.-M. Cheng, and A. Borji, "Enhanced-alignment Measure for Binary Foreground Map Evaluation," in *IJCAI*, 2018. 117

[64] D.-P. Fan, G.-P. Ji, M.-M. Cheng, and L. Shao, "Concealed object detection," *IEEE T-PAMI*, 2022. 90, 101, 109, 111, 118, 119, 120

[65] D.-P. Fan, G.-P. Ji, G. Sun, M.-M. Cheng, J. Shen, and L. Shao, "Camouflaged object detection," in *CVPR*, 2020. 87

[66] D.-P. Fan, G.-P. Ji, P. Xu, M.-M. Cheng, C. Sakaridis, and L. Van Gool, "Advances in deep concealed scene understanding," *Visual Intelligence (VI)*, 2023. 111

[67] D.-P. Fan, G.-P. Ji, T. Zhou, G. Chen, H. Fu, J. Shen, and L. Shao, "Pranet: Parallel reverse attention network for polyp segmentation," in *International conference on medical image computing and computer-assisted intervention*. Springer, 2020, pp. 263–273. 119, 120

[68] D.-P. Fan, Y. Zhai, A. Borji, J. Yang, and L. Shao, "Bbs-net: Rgb-d salient object detection with a bifurcated backbone strategy network," in *ECCV*, 2020. 74, 75

[69] H. Fan, H. Su, and L. J. Guibas, "A point set generation network for 3d object reconstruction from a single image," in *CVPR*, 2017, pp. 605–613. 11

[70] H. Fan, H. Su, and L. J. Guibas, "A point set generation network for 3D object reconstruction from a single image," in *CVPR*, 2017. 33, 41, 44, 45

[71] Y. Fang, S. Yang, X. Wang, Y. Li, C. Fang, Y. Shan, B. Feng, and W. Liu, "Instances as queries," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2021, pp. 6910–6919. 103

[72] P. J. S. Fleming, P. D. Meek, G. Ballard, P. B. Banks, A. W. Claridge, J. G. Sanderson, and D. E. Swann, *Camera Trapping: Wildlife Management and Research*. CSIRO Publishing, 2014. 87

[73] J. Fu, J. Liu, H. Tian, Y. Li, Y. Bao, Z. Fang, and H. Lu, "Dual attention network for scene segmentation," in *CVPR*, 2019. 57, 71, 72, 74, 75, 78

[74] K. Fu, D.-P. Fan, G.-P. Ji, and Q. Zhao, "Jl-dcf: Joint learning and densely-cooperative fusion framework for rgb-d salient object detection," in *CVPR*, 2020. 74, 75

[75] R. Gal, M. Arar, Y. Atzmon, A. H. Bermano, G. Chechik, and D. Cohen-Or, "Encoder-based domain tuning for fast personalization of text-to-image models," *ACM Transactions on Graphics*, vol. 42, no. 4, pp. 1–13, 2023. 91

[76] M. Gao, C. Xing, J. C. Niebles, J. Li, R. Xu, W. Liu, and C. Xiong, "Open vocabulary object detection with pseudo bounding-box labels," in *Proceedings of the European Conference on Computer Vision*, 2022, pp. 266–282. 87, 92

[77] G. Ghiasi, X. Gu, Y. Cui, and T. Lin, "Scaling open-vocabulary image segmentation with image-level labels," in *Proceedings of the European Conference on Computer Vision*, 2022, pp. 540–557. 87, 92, 106

[78] R. Girdhar, D. Fouhey, M. Rodriguez, and A. Gupta, "Learning a predictable and generative vector representation for objects," in *ECCV*, 2016. 33

[79] X. Glorot, A. Bordes, and Y. Bengio, "Deep sparse rectifier neural networks," in *Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics*, ser. Proceedings of Machine Learning Research, vol. 15, 2011. 40

[80] B. Graham, M. Engelcke, and L. Van Der Maaten, "3d semantic segmentation with submanifold sparse convolutional networks," in *CVPR*, 2018, pp. 9224–9232. 10

[81] A. Gropp, L. Yariv, N. Haim, M. Atzmon, and Y. Lipman, "Implicit geometric regularization for learning shapes," in *Proceedings of Machine Learning and Systems 2020*, 2020, pp. 3569–3579. 15, 16, 23

[82] T. Groueix, M. Fisher, V. G. Kim, B. Russell, and M. Aubry, "AtlasNet: A Papier-Mâché Approach to Learning 3D Surface Generation," in *CVPR*, 2018, pp. 216–224. 11

[83] X. Gu, T. Lin, W. Kuo, and Y. Cui, "Open-vocabulary object detection via vision and language knowledge distillation," in *Proceedings of the International Conference on Learning Representations*, 2022. 92

[84] H. Guan, J. Lin, and R. W. Lau, "Learning semantic associations for mirror detection," in *CVPR*, 2022. 58, 73, 74, 75

[85] R. Guo, D. Niu, L. Qu, and Z. Li, "Sotr: Segmenting objects with transformers," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2021, pp. 7157–7166. 103

[86] N.-T. Hai, N. E-Ro, V. Tuan-Anh, H. Binh-Son, T. Minh-Triet, and Y. Sai-Kit, "Improving referring image segmentation using vision-aware text features," in *arXiv preprint arXiv:2404.08590*, 2024. 7

[87] A. Hassani, S. Walton, J. Li, S. Li, and H. Shi, "Neighborhood attention transformer," in *CVPR*, 2023. 59, 78

[88] C. Hazirbas, L. Ma, C. Domokos, and D. Cremers, "Fusenet: incorporating depth into semantic segmentation via fusion-based cnn architecture," in *ACCV*, 2016. 78

[89] C. He, K. Li, Y. Zhang, L. Tang, Y. Zhang, Z. Guo, and X. Li, "Camouflaged object detection with feature decomposition and edge reconstruction," in *CVPR*, 2023. 87, 90, 111

[90] H. He, X. Li, G. Cheng, J. Shi, Y. Tong, G. Meng, V. Prinet, and L. Weng, "Enhanced boundary learning for glass-like object segmentation," in *ICCV*, 2021. 58, 62, 78

[91] K. He, G. Gkioxari, P. Dollár, and R. B. Girshick, "Mask R-CNN," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2017, pp. 2980–2988. 98, 103

[92] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *CVPR*, 2016. 12

[93] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *CVPR*, 2016. 36, 40

[94] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *IEEE/CVF conference on Computer Vision and Pattern Recognition (CVPR)*, 2016, pp. 770–778. 109, 119

[95] R. He, Q. Dong, J. Lin, and R. W. Lau, "Weakly-supervised camouflaged object detection with scribble annotations," in *AAAI*, 2023. 109

[96] R. He, J. Lin, and R. W. Lau, "Efficient mirror detection via multi-level heterogeneous learning," in *AAAI*, 2023. 58, 59

[97] A. Hertz, R. Mokady, J. Tenenbaum, K. Aberman, Y. Pritch, and D. Cohen-Or, "Prompt-to-prompt image editing with cross attention control," in *Proceedings of the International Conference on Learning Representations*, 2023. 91

[98] J. Ho, A. Jain, and P. Abbeel, "Denoising diffusion probabilistic models," in *Proceedings of the Advances in Neural Information Processing Systems*, 2020, pp. 6840–6851. 91

[99] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Computation*, pp. 1735–1780, 1997. 35

[100] A. G. Howard, "Some improvements on deep convolutional neural network based image classification," *arXiv 1312.5402*, 2013. 12, 13

[101] Q. Hu, B. Yang, L. Xie, S. Rosa, Y. Guo, Z. Wang, N. Trigoni, and A. Markham, "Randla-net: Efficient semantic segmentation of large-scale point clouds," in *CVPR*, 2020, pp. 11 108–11 117. 10, 19, 20, 22, 23

[102] H. Huang, D. Li, H. Zhang, U. Ascher, and D. Cohen-Or, "Consolidation of unorganized point clouds for surface reconstruction," *ACM TOG*, 2009. 12

[103] H. Huang, S. Wu, M. Gong, D. Cohen-Or, and U. Ascher, "Edge-aware point set resampling," *ACM TOG*, 2013. 12

[104] J. Huang and S. You, "Point cloud labeling using 3d convolutional neural network," in *2016 23rd International Conference on Pattern Recognition (ICPR)*, 2016, pp. 2670–2675. 10

[105] Q. Huang, W. Wang, and U. Neumann, "Recurrent slice networks for 3d segmentation of point clouds," in *CVPR*, 2018, pp. 2626–2635. 10

[106] T. Huang, B. Dong, J. Lin, X. Liu, R. W. Lau, and W. Zuo, "Symmetry-aware transformer-based mirror detection," in *AAAI*, 2023. 58, 59, 73, 75

[107] X. Huang and S. J. Belongie, "Arbitrary style transfer in real-time with adaptive instance normalization," *ICCV*, 2017. 34

[108] X. Huang and S. J. Belongie, "Arbitrary style transfer in real-time with adaptive instance normalization," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2017, pp. 1510–1519. 99

[109] Z. Huang, L. Huang, Y. Gong, C. Huang, and X. Wang, "Mask scoring r-cnn," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019, pp. 6409–6418. 103

[110] Z. Huang, K. Wang, K. Yang, R. Cheng, and J. Bai, "Glass detection and recognition based on the fusion of ultrasonic sensor and RGB-D sensor for the visually impaired," in *Target and Background Signatures IV*. SPIE, 2018. 57

[111] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," in *ICML*, 2015. 40

[112] A. Jaegle, F. Gimeno, A. Brock, O. Vinyals, A. Zisserman, and J. Carreira, "Perceiver: General perception with iterative attention," in *ICML*, 2021, pp. 4651–4664. 37, 38

[113] G.-P. Ji, D.-P. Fan, Y.-C. Chou, D. Dai, A. Liniger, and L. Van Gool, "Deep gradient learning for efficient camouflaged object detection," *Machine Intelligence Research*, 2023. 111

[114] P. Ji, Y. Zhong, H. Li, and M. Salzmann, "Null space clustering with applications to motion segmentation and face clustering," in *2014 IEEE International Conference on Image Processing (ICIP)*, 2014, pp. 283–287. 112

[115] C. Jia, Y. Yang, Y. Xia, Y. Chen, Z. Parekh, H. Pham, Q. V. Le, Y. Sung, Z. Li, and T. Duerig, "Scaling up visual and vision-language representation learning with noisy text supervision," in *Proceedings of the International Conference on Machine Learning*, 2021, pp. 4904–4916. 92

[116] B. Jiang, Y. Zhang, X. Wei, X. Xue, and Y. Fu, "Learning compositional representation for 4D captures with neural ODE," in *CVPR*, 2021. 34, 41, 44, 45

[117] C. M. Jiang, A. Sud, A. Makadia, J. Huang, M. Nießner, and T. Funkhouser, "Local implicit grid representations for 3D scenes," in *CVPR*, 2020. 33

[118] J. Jiang, L. Zheng, F. Luo, and Z. Zhang, "Rednet: Residual encoder-decoder network for indoor rgb-d semantic segmentation," *arXiv preprint*, 2018. 78

[119] Q. Jin, Z. Meng, T. D. Pham, Q. Chen, L. Wei, and R. Su, "DUNet: A deformable network for retinal vessel segmentation," *Knowledge-Based Systems*, 2019. 74

[120] E. Kalogerakis, M. Averkiou, S. Maji, and S. Chaudhuri, "3d shape segmentation with projective convolutional networks," in *CVPR*, 2017, pp. 3779–3788. 10

[121] A. Kalra, V. Taamazyan, S. K. Rao, K. Venkataraman, R. Raskar, and A. Kadambi, "Deep polarization cues for transparent object segmentation," in *2020 CVPR*, 2020. 70, 71

[122] A. Kanazawa, M. J. Black, D. W. Jacobs, and J. Malik, "End-to-end recovery of human shape and pose," in *CVPR*, 2018, pp. 7122–7131. 11

[123] A. Kanazawa, S. Tulsiani, A. A. Efros, and J. Malik, "Learning category-specific mesh reconstruction from image collections," in *ECCV*, 2018. 33

[124] A. Kanazawa, J. Y. Zhang, P. Felsen, and J. Malik, "Learning 3D human dynamics from video," in *CVPR*, 2019. 30

[125] T. Karras, S. Laine, M. Aittala, J. Hellsten, J. Lehtinen, and T. Aila, "Analyzing and improving the image quality of stylegan," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020, pp. 8107–8116. 91

[126] H. Kato, Y. Ushiku, and T. Harada, "Neural 3D mesh renderer," in *CVPR*, 2018. 33

[127] T. Kawabe, K. Maruya, and S. Nishida, "Perceptual transparency from image deformation," *Proceedings of the National Academy of Sciences*, vol. 112, no. 33, pp. E4620–E4627, 2015. 56

[128] M. Kazhdan, M. Bolitho, and H. Hoppe, "Poisson Surface Reconstruction," in *Symposium on Geometry Processing*, 2006. 15, 16, 23, 25, 33

[129] L. Ke, M. Danelljan, X. Li, Y.-W. Tai, C.-K. Tang, and F. Yu, "Mask transfiner for high-quality instance segmentation," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022, pp. 4412–4421. 103

[130] I. Kim, Y. Kim, and S. Kim, "Learning loss for test-time augmentation," in *NeurIPS*, 2020. 13

[131] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," in *ICLR*, 2015. 43

[132] A. Kirillov, E. Mintun, N. Ravi, H. Mao, C. Rolland, L. Gustafson, T. Xiao, S. Whitehead, A. C. Berg, W.-Y. Lo *et al.*, "Segment anything," *IEEE/CVF International Conference on Computer Vision (ICCV)*, 2023. 56

[133] A. Kirillov, E. Mintun, N. Ravi, H. Mao, C. Rolland, L. Gustafson, T. Xiao, S. Whitehead, A. C. Berg, W.-Y. Lo *et al.*, "Segment anything," in *ICCV*, 2023. 113

[134] R. Klokov and V. Lempitsky, "Escape from cells: Deep kd-networks for the recognition of 3d point cloud models," in *ICCV*, 2017, pp. 863–872. 26

[135] M. Kowal, M. Siam, M. A. Islam, N. D. Bruce, R. P. Wildes, and K. G. Derpanis, "A deeper dive into what deep spatiotemporal networks encode: Quantifying static vs. dynamic information," in *CVPR*, 2022. 112

[136] P. Krähenbühl and V. Koltun, "Efficient inference in fully connected crfs with gaussian edge potentials," in *NeurIPS*, J. Shawe-Taylor, R. Zemel, P. Bartlett, F. Pereira, and K. Weinberger, Eds., 2011. 73

[137] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *NeurIPS*, 2012. 12

[138] H. W. Kuhn, "The Hungarian Method for the Assignment Problem," *Naval Research Logistics Quarterly*, vol. 2, no. 1, pp. 83–97, March 1955. 100

[139] N. Kumari, B. Zhang, R. Zhang, E. Shechtman, and J.-Y. Zhu, "Multi-concept customization of text-to-image diffusion," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2023, pp. 1931–1941. 91

[140] W. Kuo, Y. Cui, X. Gu, A. Piergiovanni, and A. Angelova, "F-vlm: Open-vocabulary object detection upon frozen vision and language models," in *Proceedings of the International Conference on Learning Representations*, 2023. 87, 92

[141] H. Lamdouar, W. Xie, and A. Zisserman, "Segmenting invisible moving objects," in *BMVC*, 2021. 112, 118

[142] H. Lamdouar, W. Xie, and A. Zisserman, "The making and breaking of camouflage," in *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 2023, pp. 832–842. 109, 112, 113, 114, 118

[143] H. Lamdouar, C. Yang, W. Xie, and A. Zisserman, "Betrayed by motion: Camouflaged object discovery via motion segmentation," *ACCV*, 2020. 110, 111, 112, 115

[144] L. Landrieu and M. Simonovsky, "Large-scale point cloud semantic segmentation with superpoint graphs," in *CVPR*, 2018, pp. 4558–4567. 11

[145] T. Le and Y. Duan, "Pointgrid: A deep network for 3d shape understanding," in *CVPR*, 2018, pp. 9204–9214. 11

[146] T.-N. Le, Y. Cao, T.-C. Nguyen, M.-Q. Le, K.-D. Nguyen, T.-T. Do, M.-T. Tran, and T. V. Nguyen, "Camouflaged instance segmentation in-the-wild: Dataset, method, and benchmark suite," *IEEE T-IP*, 2021. 111

[147] T.-N. Le, T. V. Nguyen, Z. Nie, M.-T. Tran, and A. Sugimoto, "Anabranch network for camouflaged object segmentation," *CVIU*, 2019. 109

[148] V. Leroy, J.-S. Franco, and E. Boyer, "Multi-view dynamic shape refinement using local temporal integration," in *ICCV*, 2017. 30, 34

[149] D. Li, H. Ling, S. W. Kim, K. Kreis, S. Fidler, and A. Torralba, "Bigdatasetgan: Synthesizing imagenet with pixel-wise annotations," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022, pp. 21 298–21 308. 91

[150] F. Li, H. Zhang, P. Sun, X. Zou, S. Liu, J. Yang, C. Li, L. Zhang, and J. Gao, "Semantic-sam: Segment and recognize anything at any granularity," 2023. 56, 84

[151] H. Li, P. Xiong, H. Fan, and J. Sun, "DFANet: Deep feature aggregation for real-time semantic segmentation," in *CVPR*, 2019. 74

[152] H. Li, P. Wang, C. Shen, and G. Zhang, "Show, attend and read: A simple and strong baseline for irregular text recognition," in *AAAI*, vol. 33, 2019, pp. 8610–8617. 13

[153] R. Li, X. Li, P.-A. Heng, and C.-W. Fu, "PointAugment: An auto-augmentation framework for point cloud classification," in *CVPR*, 2020. 13

[154] R. Li, X. Li, P.-A. Heng, and C.-W. Fu, "Point cloud upsampling via disentangled refinement," in *CVPR*, 2021, pp. 344–353. 9

[155] S. Li, Z. Wang, Z. Liu, C. Tan, H. Lin, D. Wu, Z. Chen, J. Zheng, and S. Z. Li, "Efficient multi-order gated aggregation network," *ArXiv*, vol. abs/2211.03295v2, 2023. 59, 78

[156] Y. Li, H. Takehara, T. Taketomi, B. Zheng, and M. Nießner, "4dcomplete: Non-rigid motion estimation beyond the observable surface." *ICCV*, 2021. 34, 42, 54

[157] Y. Li, R. Bu, M. Sun, W. Wu, X. Di, and B. Chen, "Pointcnn: Convolution on x-transformed points," in *NeurIPS*, 2018, pp. 820–830. 10

[158] Y. Liao, S. Donne, and A. Geiger, "Deep marching cubes: Learning explicit surface representations," in *CVPR*, 2018, pp. 2916–2925. 11

[159] Y. Liao, S. Donné, and A. Geiger, "Deep marching cubes: Learning explicit surface representations," in *CVPR*, 2018. 33

[160] C.-H. Lin, C. Kong, and S. Lucey, "Learning efficient point cloud generation for dense 3d object reconstruction," in *AAAI*, vol. 32, 2018. 11

[161] G. Lin, A. Milan, C. Shen, and I. D. Reid, "RefineNet: Multi-path refinement networks for high-resolution semantic segmentation," in *CVPR*, 2017. 74, 78

[162] J. Lin, Z. He, and R. W. Lau, "Rich context aggregation with reflection prior for glass surface detection," in *CVPR*, 2021. 56, 58, 70, 71, 72

[163] J. Lin, G. Wang, and R. W. Lau, "Progressive mirror detection," in *CVPR*, 2020. 59, 65, 66, 73, 75

[164] J. Lin, Y.-H. Yeung, and R. W. Lau, "Exploiting semantic relations for glass surface detection," *NeurIPS*, 2022. 56, 65, 66, 71, 72

[165] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick, "Microsoft coco: Common objects in context," in *European Conference on Computer Vision (ECCV)*. Springer, 2014, pp. 740–755. 109

[166] T. Lin, M. Maire, S. J. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick, "Microsoft COCO: common objects in context," in *Proceedings of the European Conference on Computer Vision*, 2014, pp. 740–755. 100

[167] Y. Lipman, D. Cohen-Or, D. Levin, and H. Tal-Ezer, "Parameterization-free projection for geometry reconstruction," *ACM TOG*, 2007. 12

[168] N. Liu, N. Zhang, and J. Han, "Learning selective self-mutual attention for rgb-d saliency detection," in *CVPR*, 2020. 74, 75

[169] N. Liu, N. Zhang, K. Wan, L. Shao, and J. Han, "Visual saliency transformer," in *ICCV*, 2021. 73, 74, 75

[170] X. Liu, M. Yan, and J. Bohg, "MeteorNet: Deep learning on dynamic 3D point cloud sequences," in *ICCV*, 2019. 36

[171] Y. Liu, B. Fan, S. Xiang, and C. Pan, "Relation-shape convolutional neural network for point cloud analysis," in *CVPR*, 2019, pp. 8895–8904. 26

[172] Z. Liu, Y. Lin, Y. Cao, H. Hu, Y. Wei, Z. Zhang, S. Lin, and B. Guo, "Swin transformer: Hierarchical vision transformer using shifted windows," in *ICCV*, 2021. 59, 71, 72

[173] Z. Liu, H. Tang, Y. Lin, and S. Han, "Point-voxel cnn for efficient 3d deep learning," in *NeurIPS*, 2019, pp. 965–975. 11

[174] J. Long, E. Shelhamer, and T. Darrell, "Fully convolutional networks for semantic segmentation," in *CVPR*, 2015. 74, 80

[175] W. E. Lorensen and H. E. Cline, "Marching cubes: A high resolution 3d surface construction algorithm," *SIGGRAPH*, 1987. 15

[176] W. E. Lorensen and H. E. Cline, "Marching cubes: A high resolution 3D surface construction algorithm," *SIGGRAPH Comput. Graph.*, 1987. 33, 46

[177] I. Loshchilov and F. Hutter, "Decoupled weight decay regularization," in *ICLR*, 2019. 66, 101

[178] N. Luo, Y. Pan, R. Sun, T. Zhang, Z. Xiong, and F. Wu, "Camouflaged instance segmentation via explicit de-camouflaging," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2023, pp. 17 918–17 927. 102, 103, 104

[179] Y. Lv, J. Zhang, Y. Dai, A. Li, N. Barnes, and D.-P. Fan, "Towards deeper understanding of camouflaged object detection," *IEEE T-CSVT*, 2023. 109, 111

[180] Y. Lv, J. Zhang, Y. Dai, A. Li, B. Liu, N. Barnes, and D.-P. Fan, "Simultaneously localize, segment and rank the camouflaged objects," in *CVPR*, 2021. 111

[181] Y. Lyu, J. Zhang, Y. Dai, A. Li, B. Liu, N. Barnes, and D.-P. Fan, "Simultaneously localize, segment and rank the camouflaged objects," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021, pp. 11 586–11 596. 90, 101

[182] A. Lyzhov, Y. Molchanova, A. Ashukha, D. Molchanov, and D. Vetrov, "Greedy policy search: A simple baseline for learnable test-time augmentation," in *Proceedings of the 36th Conference on Uncertainty in Artificial Intelligence (UAI)*, 2020. 13

[183] M. Lüthi, T. Gerig, C. Jud, and T. Vetter, "Gaussian process morphable models," *TPAMI*, 2018. 35

[184] R. Margolin, L. Zelnik-Manor, and A. Tal, "How to evaluate foreground maps?" in *CVPR*, June 2014. 69

[185] R. Margolin, L. Zelnik-Manor, and A. Tal, "How to evaluate foreground maps?" in *CVPR*, 2014. 117

[186] H. Mei, X. Yang, L. Yu, Q. Zhang, X. Wei, and R. H. Lau, "Large-field contextual feature learning for glass detection," *TPAMI*, 2023. 58, 62

[187] H. Mei, B. Dong, W. Dong, P. Peers, X. Yang, Q. Zhang, and X. Wei, "Depth-aware mirror segmentation," in *CVPR*, 2021. 63, 65, 66, 74, 75, 83

[188] H. Mei, B. Dong, W. Dong, J. Yang, S.-H. Baek, F. Heide, P. Peers, X. Wei, and X. Yang, "Glass segmentation using intensity and spectral polarization cues," in *CVPR*, June 2022. 56, 58, 59, 65, 66, 68, 70, 71, 83

[189] H. Mei, X. Yang, Y. Wang, Y. Liu, S. He, Q. Zhang, X. Wei, and R. W. Lau, "Don't hit me! glass detection in real-world scenes," in *CVPR*, June 2020. 70, 71, 72

[190] M. Menze and A. Geiger, "Object scene flow for autonomous vehicles," in *CVPR*, 2015. 115

[191] L. Mescheder, M. Oechsle, M. Niemeyer, S. Nowozin, and A. Geiger, "Occupancy networks: Learning 3d reconstruction in function space," in *CVPR*, 2019, pp. 4460–4470. 9, 11, 15, 17

[192] L. Mescheder, M. Oechsle, M. Niemeyer, S. Nowozin, and A. Geiger, "Occupancy networks: Learning 3D reconstruction in function space," in *CVPR*, 2019. 30, 33, 40, 41, 43, 44, 45, 46

[193] E. Meunier, A. Badoual, and P. Bouthemy, "Em-driven unsupervised learning for efficient motion segmentation," *IEEE T-PAMI*, 2022. 112

[194] M. Michalkiewicz, J. K. Pontes, D. Jack, M. Baktashmotlagh, and A. Eriksson, "Implicit surface representations as layers in neural networks," in *ICCV*, 2019. 30

[195] F. Milletari, N. Navab, and S.-A. Ahmadi, "V-net: Fully convolutional neural networks for volumetric medical image segmentation," in *3DV*, 2016, pp. 565–571. 63, 100

[196] M. Minderer, A. A. Gritsenko, A. Stone, M. Neumann, D. Weissenborn, A. Dosovitskiy, A. Mahendran, A. Arnab, M. Dehghani, Z. Shen, X. Wang, X. Zhai, T. Kipf, and N. Houlsby, "Simple open-vocabulary object detection with vision transformers," in *Proceedings of the European Conference on Computer Vision*, 2022, pp. 728–755. 87, 92

[197] N. Moshkov, B. Mathe, A. Kertesz-Farkas, R. Hollandi, and P. Horvath, "Test-time augmentation for deep learning-based cell segmentation on microscopy images," *Scientific reports*, vol. 10, no. 1, pp. 1–7, 2020. 13

[198] A. Mustafa, H. Kim, J.-Y. Guillemaut, and A. Hilton, "General dynamic scene reconstruction from multiple view video," in *ICCV*, 2015. 30, 34

[199] T. T. T. Nguyen, A. C. Eichholtzer, D. A. Driscoll, N. I. Semianiw, D. M. Corva, A. Z. Kouzani, T. T. Nguyen, and D. T. Nguyen, "Sawit: A small-sized animal wild image dataset with annotations," *Multimedia Tools and Applications*, pp. 1–26, 2023. 87

[200] C. Nhat Minh, G. Sensen, V. Tuan-Anh, Z. Jie, L. Aishan, L. Yun, D. Jin Song, and G. Qing, "Towards transferable attacks against vision-llms in autonomous driving with typography," in *arXiv preprint arXiv:2405.14169*, 2024. 7

[201] A. Q. Nichol, P. Dhariwal, A. Ramesh, P. Shyam, P. Mishkin, B. McGrew, I. Sutskever, and M. Chen, "GLIDE: towards photorealistic image generation and editing with text-guided diffusion models," in *Proceedings of the International Conference on Machine Learning*, 2022, pp. 16 784–16 804. 93

[202] M. Niemeyer, L. M. Mescheder, M. Oechsle, and A. Geiger, "Occupancy flow: 4D reconstruction by learning particle dynamics," in *ICCV*, 2019. 30, 34, 35, 36, 39, 41, 42, 43, 44, 45, 48, 51

[203] A. Okazawa, T. Takahata, and T. Harada, "Simultaneous transparent and non-transparent object segmentation with multispectral scenes," in *IROS*, 2019. 58

[204] M. Oquab, T. Darcet, T. Moutakanni, H. V. Vo, M. Szafraniec, V. Khalidov, P. Fernandez, D. HAZIZA, F. Massa, A. El-Nouby, M. Assran, N. Ballas, W. Galuba, R. Howes, P.-Y.

Huang, S.-W. Li, I. Misra, M. Rabbat, V. Sharma, G. Synnaeve, H. Xu, H. Jegou, J. Mairal, P. Labatut, A. Joulin, and P. Bojanowski, "DINOv2: Learning robust visual features without supervision," *Transactions on Machine Learning Research*, 2024. 114

[205] Y. Pang, L. Zhang, X. Zhao, and H. Lu, "Hierarchical dynamic filtering network for rgb-d salient object detection," in *ECCV*, 2020. 74, 75

[206] Y. Pang, X. Zhao, L. Zhang, and H. Lu, "Multi-scale interactive network for salient object detection," in *CVPR*, 2020. 73, 75

[207] J. J. Park, P. Florence, J. Straub, R. Newcombe, and S. Lovegrove, "Deepsdf: Learning continuous signed distance functions for shape representation," in *CVPR*, 2019, pp. 165–174. 9, 11

[208] J. J. Park, P. Florence, J. Straub, R. Newcombe, and S. Lovegrove, "Deepsdf: Learning continuous signed distance functions for shape representation," in *CVPR*, 2019. 30, 33

[209] G. Parmar, K. Kumar Singh, R. Zhang, Y. Li, J. Lu, and J.-Y. Zhu, "Zero-shot image-to-image translation," in *Proceedings of the ACM SIGGRAPH*, 2023, pp. 1–11. 91

[210] J. Pei, T. Cheng, D.-P. Fan, H. Tang, C. Chen, and L. Van Gool, "Osformer: One-stage camouflaged instance segmentation with transformers," in *ECCV*, 2022. 87, 90, 99, 103, 111

[211] Y. Pekelny and C. Gotsman, "Articulated object reconstruction and markerless motion capture from depth video," *Computer Graphics Forum*, 2008. 30

[212] S. Peng, M. Niemeyer, L. Mescheder, M. Pollefeys, and A. Geiger, "Convolutional occupancy networks," in *ECCV*, 2020. 9, 11, 15, 16, 19, 20, 23, 25

[213] S. Peng, M. Niemeyer, L. Mescheder, M. Pollefeys, and A. Geiger, "Convolutional occupancy networks," in *ECCV*, 2020. 33

[214] E. L.-M. Pia Bideau, "It's moving! a probabilistic model for causal motion segmentation in moving camera videos," in *ECCV*, 2016. 111

[215] S. Prokudin, C. Lassner, and J. Romero, "Efficient learning on point clouds with basis point sets," in *ICCV*, 2019, pp. 4332–4341. 11

[216] C. R. Qi, W. Liu, C. Wu, H. Su, and L. J. Guibas, "Frustum pointnets for 3D object detection from RGB-D data," in *CVPR*, 2018. 33

[217] C. R. Qi, H. Su, K. Mo, and L. J. Guibas, "Pointnet: Deep learning on point sets for 3d classification and segmentation," in *CVPR*, 2017, pp. 652–660. 10, 12, 14, 15, 17, 19, 20, 21, 24, 25, 26

[218] C. R. Qi, H. Su, K. Mo, and L. J. Guibas, "PointNet: Deep learning on point sets for 3D classification and segmentation," in *CVPR*, 2017. 36

[219] C. R. Qi, L. Yi, H. Su, and L. J. Guibas, "Pointnet++: Deep hierarchical feature learning on point sets in a metric space," in *NeurIPS*, 2017, pp. 5105–5114. 10, 12

[220] G. Qian, Y. Li, H. Peng, J. Mai, H. Hammoud, M. Elhoseiny, and B. Ghanem, "Pointnext: Revisiting pointnet++ with improved training and scaling strategies," in *NeurIPS*, 2022. 17, 19, 20, 21, 22, 24

[221] A. Radford, J. W. Kim, C. Hallacy, A. Ramesh, G. Goh, S. Agarwal, G. Sastry, A. Askell, P. Mishkin, J. Clark, G. Krueger, and I. Sutskever, "Learning transferable visual models from natural language supervision," in *Proceedings of the International Conference on Machine Learning*, 2021, pp. 8748–8763. 88, 92, 94, 95, 98, 120, 121

[222] C. Raffel, N. Shazeer, A. Roberts, K. Lee, S. Narang, M. Matena, Y. Zhou, W. Li, and P. J. Liu, "Exploring the limits of transfer learning with a unified text-to-text transformer," *Journal of Machine Learning Research*, vol. 21, pp. 1–67, 2020. 93

[223] A. Ramesh, P. Dhariwal, A. Nichol, C. Chu, and M. Chen, "Hierarchical text-conditional image generation with clip latents," *arXiv preprint arXiv:2204.06125*, pp. 1–27, 2022. 91

[224] M. R. Rands, W. M. Adams, L. Bennun, S. H. Butchart, A. Clements, D. Coomes, A. Entwistle, I. Hodge, V. Kapos, J. P. Scharlemann *et al.*, "Biodiversity conservation: challenges beyond 2010," *science*, vol. 329, no. 5997, pp. 1298–1303, 2010. 109

[225] A. Ranjan, T. Bolkart, S. Sanyal, and M. J. Black, "Generating 3d faces using convolutional mesh autoencoders," in *ECCV*, 2018, pp. 704–720. 11

[226] H. A. Rasheed, M. Maaz, M. U. Khattak, S. H. Khan, and F. S. Khan, "Bridging the gap between object and image-level representations for open-vocabulary detection," in *Proceedings of the Advances in Neural Information Processing Systems*, 2022, pp. 33 781–33 794. 92

[227] S. Ren, K. He, R. Girshick, and J. Sun, "Faster R-CNN: Towards real-time object detection with region proposal networks," *Advances in neural information processing systems*, vol. 28, 2015. 119, 120

[228] S. Ren, K. He, R. B. Girshick, and J. Sun, "Faster R-CNN: towards real-time object detection with region proposal networks," in *Advances in Neural Information Processing Systems*, 2015, pp. 91–99. 92

[229] P. Rewatbowornwong, N. Tritrong, and S. Suwajanakorn, "Repurposing gans for one-shot semantic part segmentation," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 45, no. 4, pp. 5114–5125, 2023. 91

[230] R. Robin, B. Andreas, L. Dominik, E. Patrick, and O. Björn, "High-resolution image synthesis with latent diffusion models," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022, pp. 10 674–10 685. 88, 91, 94

[231] O. Ronneberger, P. Fischer, and T. Brox, "U-net: Convolutional networks for biomedical image segmentation," in *MICCAI*, 2015, pp. 234–241. 15

[232] O. Ronneberger, P. Fischer, and T. Brox, "U-Net: Convolutional networks for biomedical image segmentation," in *MICCAI*, 2015. 74

[233] C. Saharia, W. Chan, S. Saxena, L. Li, J. Whang, E. Denton, S. K. S. Ghasemipour, B. K. Ayan, S. S. Mahdavi, R. G. Lopes *et al.*, "Photorealistic text-to-image diffusion models with deep language understanding," in *Proceedings of the Advances in Neural Information Processing Systems*, 2022, pp. 36 479–36 494. 91, 93

[234] M. B. Sariyildiz, J. Perez, and D. Larlus, "Learning visual representations with caption annotations," in *Proceedings of the European Conference on Computer Vision*, 2020, pp. 1–17. 94

[235] I. Sato, H. Nishimura, and K. Yokoi, "Apac: Augmented pattern classification with neural networks," *arXiv 1505.03229*, 2015. 13

[236] M. Sawayama, Y. Dobashi, M. Okabe, K. Hosokawa, T. Koumura, T. P. Saarela, M. Olkkonen, and S. Nishida, "Visual discrimination of optical material properties: A large-scale study," *Journal of Vision*, vol. 22, no. 2, pp. 17–17, 02 2022. 56

[237] N. Schlüter and F. Faul, "Visual shape perception in the case of transparent objects," *Journal of Vision*, vol. 19, no. 4, pp. 24–24, 04 2019. 56

[238] C. Schuhmann, R. Beaumont, R. Vencu, C. Gordon, R. Wightman, M. Cherti, T. Coombes, A. Katta, C. Mullis, M. Wortsman *et al.*, "LAION-5b: An open large-scale dataset for training next generation image-text models," *Advances in Neural Information Processing Systems*, pp. 25 278–25 294, 2022. 94

[239] D. Seichter, S. Fischedick, M. Köhler, and H.-M. Gross, "Efficient multi-task rgb-d scene analysis for indoor environments," in *IEEE International Joint Conference on Neural Networks (IJCNN)*, 2022. 56, 78

[240] D. Shanmugam, D. Blalock, G. Balakrishnan, and J. Guttag, "When and why test-time augmentation works," *arXiv 2011.11156*, 2020. 13

[241] S. Shao, Z. Li, T. Zhang, C. Peng, G. Yu, X. Zhang, J. Li, and J. Sun, "Objects365: A large-scale, high-quality dataset for object detection," in *IEEE/CVF international conference on computer vision (CVPR)*, 2019, pp. 8430–8439. 109

[242] S. Shen, C. Li, X. Hu, Y. Xie, J. Yang, P. Zhang, Z. Gan, L. Wang, L. Yuan, C. Liu *et al.*, "K-lite: Learning transferable visual models with external knowledge," *Advances in Neural Information Processing Systems*, vol. 35, pp. 15 558–15 573, 2022. 120, 121

[243] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," in *ICLR*, 2015. 12

[244] A. Sinha, J. Bai, and K. Ramani, "Deep learning 3d shape surfaces using geometry images," in *ECCV*, 2016, pp. 223–240. 11

[245] A. Siris, J. Jiao, G. K. Tam, X. Xie, and R. W. Lau, "Scene context-aware salient object detection," in *ICCV*, 2021. 72

[246] V. Sitzmann, M. Zollhöfer, and G. Wetzstein, "Scene representation networks: Continuous 3d-structure-aware neural scene representations," in *NeurIPS*, 2019. 9

[247] M. Slavcheva, M. Baust, and S. Ilic, "Towards implicit correspondence in signed distance field evolution," in *ICCV Workshops*, 2017. 35

[248] J. Song, C. Meng, and S. Ermon, "Denoising diffusion implicit models," in *Proceedings of the International Conference on Learning Representations*, 2021. 91

[249] Z. Song, Z. Zhang, K. Zhang, W. Luo, Z. Fan, W. Ren, and J. Lu, "Robust single image reflection removal against adversarial attacks," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2023, pp. 24 688–24 698. 64, 83

[250] M. Soofi, S. Sharma, B. Safaei-Mahroo, M. Sohrabi, M. G. Organli, and M. Waltert, "Lichens and animal camouflage: some observations from central asian ecoregions," *Journal of Threatened Taxa*, vol. 14, no. 2, pp. 20 672–20 676, 2022. 109

[251] R. Strudel, R. Garcia, I. Laptev, and C. Schmid, "Segmenter: Transformer for semantic segmentation," in *ICCV*, 2021. 58, 72

[252] D. Stutz and A. Geiger, "Learning 3d shape completion from laser scan data with weak supervision," in *CVPR*, 2018, pp. 1955–1964. 11

[253] G. Sun, Z. An, Y. Liu, C. Liu, C. Sakaridis, D.-P. Fan, and L. Van Gool, "Indiscernible object counting in underwater scenes," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2023, pp. 13 791–13 801. 90

[254] G. Sun, Z. An, Y. Liu, C. Liu, C. Sakaridis, D.-P. Fan, and L. Van Gool, "Indiscernible object counting in underwater scenes," in *CVPR*, 2023. 111

[255] S. Sun, K. Han, D. Kong, H. Tang, X. Yan, and X. Xie, "Topology-preserving shape reconstruction and registration via neural diffeomorphic flow," in *CVPR*, June 2022, pp. 20 845–20 855. 35

[256] T. Sun, G. Zhang, W. Yang, J.-H. Xue, and G. Wang, "TROSD: A new RGB-D dataset for transparent and reflective object segmentation in practice," *IEEE TCSVT*, 2023. 56, 59, 66, 77, 78, 83

[257] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, "Going deeper with convolutions," in *CVPR*, 2015. 12

[258] X. Tan, J. Lin, K. Xu, P. Chen, L. Ma, and R. W. Lau, "Mirror detection with the visual chirality cue," *IEEE TPAMI*, 2023. 58, 73, 74, 75

[259] H. Tan-Sang, N.-T. Hai, V. Tuan-Anh, and Y. Sai-Kit, "Marinevrs: Marine video retrieval system with explainability via semantic understanding," in *OCEANS 2023, Limerick*, 2023. 7

[260] J. Tang, L. Markhasin, B. Wang, J. Thies, and M. Nießner, "Neural shape deformation priors," in *NeurIPS*, A. H. Oh, A. Agarwal, D. Belgrave, and K. Cho, Eds., 2022. 42

[261] J. Tang, D. Xu, K. Jia, and L. Zhang, "Learning parallel dense correspondence from spatio-temporal descriptors for efficient and robust 4D reconstruction," in *CVPR*, 2021. 34, 36, 37, 39, 41, 44, 45, 46, 50, 51, 54

[262] M. Tatarchenko, A. Dosovitskiy, and T. Brox, "Octree generating networks: Efficient convolutional architectures for high-resolution 3D outputs," *ICCV*, 2017. 33

[263] L. Tchapmi, C. Choy, I. Armeni, J. Gwak, and S. Savarese, "Segcloud: Semantic segmentation of 3d point clouds," in *3DV*. IEEE, 2017, pp. 537–547. 10

[264] O. team, "GPT-4 technical report," *arXiv preprint arXiv:2303.08774*, 2023. 115

[265] Z. Teed and J. Deng, "RAFT: Recurrent all-pairs field transforms for optical flow," in *ECCV*, 2020. 114, 115, 118

[266] Z. Tian, C. Shen, and H. Chen, "Conditional convolutions for instance segmentation," in *Proceedings of the European Conference on Computer Vision*. Springer, 2020, pp. 282–298. 103

[267] Q.-T. Truong, T.-A. Vu, T.-S. Ha, J. Lokoč, Y. H. W. Tim, A. Joneja, and S.-K. Yeung, "Marine Video Kit: A new marine video dataset for content-based analysis and retrieval," in *MultiMedia Modeling - 29th International Conference, MMM 2023*. Springer, 2023. 7, 111

[268] V. Tuan-Anh, N. Duc-Thanh, H. Binh-Son, P. Quang-Hieu, and Y. Sai-Kit, "Rfnet-4d: Joint object reconstruction and flow estimation from 4d point clouds," in *ECCV*, 2022. 6, 37, 43, 44, 45, 46, 47, 50, 51

[269] V. Tuan-Anh, N. Duc-Thanh, H. Binh-Son, P. Quang-Hieu, and Y. Sai-Kit, "Rfnet-4d++: Joint object reconstruction and flow estimation from 4d point clouds with cross-attention

spatio-temporal features," in *PREPRINT available at Research Square, DOI: 10.21203/rs.3.rs-4390361/v1*, 2024. 6

[270] V. Tuan-Anh, N. Duc-Thanh, C. Nhat Minh, G. Qing, H. Binh-Son, T. Ivor W., and Y. Sai-Kit, "Leveraging open-vocabulary diffusion to camouflaged instance segmentation," in *arXiv preprint arXiv:2312.17505*, 2024. 6

[271] V. Tuan-Anh, N.-T. Hai, Z. Ziqiang, H. Binh-Son, G. Qing, T. Ivor W., and Y. Sai-Kit, "Transcues: Boundary and reflection-empowered pyramid vision transformer for semantic transparent object segmentation," in *OpenReview preprint*, 2024. [Online]. Available: https://openreview.net/forum?id=e9bEoxNiTJ 6

[272] V. Tuan-Anh, S. Srinjay, Z. Zhiyuan, H. Binh-Son, and Y. Sai-Kit, "Test-time augmentation for 3d point cloud classification and segmentation," in *Proceedings of International Conference on 3D Vision (3DV)*, 2024. 6

[273] V. Tuan-Anh, Z. Ziqiang, S. Chengyang, G. Qing, T. Ivor W., and Y. Sai-Kit, "Camovid60k: A large-scale video dataset for moving camouflaged animals understanding," in *Preprint*, 2024. 6

[274] H.-Y. Tung, H.-W. Tung, E. Yumer, and K. Fragkiadaki, "Self-supervised learning of motion capture," in *NeurIPS*, 2017. 30

[275] M. A. Uy, Q.-H. Pham, B.-S. Hua, D. T. Nguyen, and S.-K. Yeung, "Revisiting point cloud classification: A new benchmark dataset and classification model on real-world data," in *ICCV*, 2019. 10, 13, 19, 20

[276] A. Valada, R. Mohan, and W. Burgard, "Self-supervised model adaptation for multimodal semantic segmentation," *International Journal of Computer Vision (IJCV)*, 2019. 78

[277] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. u. Kaiser, and I. Polosukhin, "Attention is all you need," in *NeurIPS*, vol. 30, 2017. 67

[278] T.-A. Vu, D. T. Nguyen, Q. Guo, B.-S. Hua, N. M. Chung, I. W. Tsang, and S.-K. Yeung, "Leveraging open-vocabulary diffusion to camouflaged instance segmentation," *arXiv preprint arXiv:2312.17505*, 2023. 109

[279] M. Wand, P. Jenke, Q. Huang, M. Bokeloh, L. Guibas, and A. Schilling, "Reconstruction of deforming geometry from time-varying point clouds," in *Proceedings of the Fifth Eurographics Symposium on Geometry Processing*, 2007. 30

[280] G. Wang, W. Li, M. Aertsen, J. Deprest, S. Ourselin, and T. Vercauteren, "Aleatoric uncertainty estimation with test-time augmentation for medical image segmentation with convolutional neural networks," *Neurocomputing*, vol. 338, pp. 34–45, 2019. 13

[281] H. Wang, Y. Zhu, H. Adam, A. Yuille, and L.-C. Chen, "MaX-DeepLab: End-to-End panoptic segmentation with mask transformers," in *CVPR*, 2021. 58

[282] J. Wang, C. Wen, Y. Fu, H. Lin, T. Zou, X. Xue, and Y. Zhang, "Neural pose transfer by spatially adaptive instance normalization," in *CVPR*, 2020. 34

[283] J. Wang, K. Sun, T. Cheng, B. Jiang, C. Deng, Y. Zhao, D. Liu, Y. Mu, M. Tan, X. Wang, W. Liu, and B. Xiao, "Deep high-resolution representation learning for visual recognition," *TPAMI*, 2020. 74

[284] N. Wang, Y. Zhang, Z. Li, Y. Fu, W. Liu, and Y.-G. Jiang, "Pixel2Mesh: Generating 3D mesh models from single rgb images," in *ECCV*, 2018. 33

[285] P.-S. Wang, Y. Liu, Y.-X. Guo, C.-Y. Sun, and X. Tong, "O-CNN: Octree-based convolutional neural networks for 3d shape analysis," *ACM Trans. Graph.*, 2017. 33

[286] Q. Wang, Y.-Y. Chang, R. Cai, Z. Li, B. Hariharan, A. Holynski, and N. Snavely, "Tracking everything everywhere all at once," in *ICCV*, 2023. 115

[287] S. Wang, S. Suo, W.-C. Ma, A. Pokrovsky, and R. Urtasun, "Deep parametric continuous convolutional neural networks," in *CVPR*, 2018, pp. 2589–2597. 10

[288] W. Wang, J. Dai, Z. Chen, Z. Huang, Z. Li, X. Zhu, X. Hu, T. Lu, L. Lu, H. Li, X. Wang, and Y. Qiao, "Internimage: Exploring large-scale vision foundation models with deformable convolutions," in *CVPR*, 2023. 59, 78

[289] W. Wang, E. Xie, X. Li, D.-P. Fan, K. Song, D. Liang, T. Lu, P. Luo, and L. Shao, "Pyramid vision transformer: A versatile backbone for dense prediction without convolutions," in *ICCV*, 2021. 57, 59, 60, 67, 77, 80, 86

[290] W. Wang, E. Xie, X. Li, D.-P. Fan, K. Song, D. Liang, T. Lu, P. Luo, and L. Shao, "Pvtv2: Improved baselines with pyramid vision transformer," *Computational Visual Media*, vol. 8, no. 3, pp. 1–10, 2022. 60, 67, 68, 78, 79, 80

[291] X. Wang, R. Zhang, T. Kong, L. Li, and C. Shen, "Solov2: Dynamic and fast instance segmentation," *Advances in Neural Information Processing Systems*, vol. 33, pp. 17 721–17 732, 2020. 103

[292] Y. Wang, Q. Zhou, J. Liu, J. Xiong, G. Gao, X. Wu, and L. J. Latecki, "LEDNet: A lightweight encoder-decoder network for real-time semantic segmentation," in *ICIP*, 2019. 74

[293] Y. Wang, Y. Sun, Z. Liu, S. E. Sarma, M. M. Bronstein, and J. M. Solomon, "Dynamic graph cnn for learning on point clouds," *ACM Transactions on Graphics*, 2019. 10, 17, 21

[294] Y. Wang, Y. Sun, Z. Liu, S. E. Sarma, M. M. Bronstein, and J. M. Solomon, "Dynamic graph cnn for learning on point clouds," *ACM TOG*, 2019. 17, 19, 21, 22, 24

[295] Z. Wang and F. Lu, "Voxsegnet: Volumetric cnns for semantic part segmentation of 3d shapes," *IEEE TVCG*, 2019. 10

[296] J. Wei, S. Wang, Z. Wu, C. Su, Q. Huang, and Q. Tian, "Label decoupling framework for salient object detection," in *CVPR*, 2020. 73, 75

[297] C. Wen, Y. Zhang, Z. Li, and Y. Fu, "Pixel2Mesh++: Multi-view 3D mesh generation via deformation," in *ICCV*, 2019. 33

[298] Z. Wenbo, L. Xianming, Z. Zhiwei, J. Junjun, G. Wei, L. Ge, and J. Xiangyang, "Self-supervised arbitrary-scale point clouds upsampling via implicit neural representation," in *CVPR*, 2022. 9, 16, 17, 19, 20, 23, 25

[299] J. Wu, C. Zhang, T. Xue, W. T. Freeman, and J. B. Tenenbaum, "Learning a probabilistic latent space of object shapes via 3d generative-adversarial modeling," in *NeurIPS*, 2016. 11

[300] J. Wu, X. Li, S. Xu, H. Yuan, H. Ding, Y. Yang, X. Li, J. Zhang, Y. Tong, X. Jiang, B. Ghanem, and D. Tao, "Towards open vocabulary learning: A survey," *arXiv preprint arXiv:2306.15880*, pp. 1–22, 2023. 92

[301] Y. Wu, A. Kirillov, F. Massa, W.-Y. Lo, and R. Girshick, "Detectron2," https://github.com/facebookresearch/detectron2, 2019. 101

[302] Z. Wu, L. Su, and Q. Huang, "Cascaded partial decoder for fast and accurate salient object detection," in *CVPR*, 2019. 73, 75

[303] Z. Wu, S. Song, A. Khosla, F. Yu, L. Zhang, X. Tang, and J. Xiao, "3d shapenets: A deep representation for volumetric shapes," in *CVPR*, 2015, pp. 1912–1920. 10, 11, 19, 20, 26

[304] K. Xiang, K. Yang, and K. Wang, "Polarization-driven semantic segmentation via efficient attention-bridged fusion," *Optica Express*, 2021. 56, 58, 70, 71

[305] C. Xie, Y. Xiang, Z. Harchaoui, and D. Fox, "Object discovery in videos as foreground motion clustering," in *CVPR*, 2019. 112

[306] E. Xie, W. Wang, Z. Yu, A. Anandkumar, J. M. Álvarez, and P. Luo, "Segformer: Simple and efficient design for semantic segmentation with transformers," in *NeurIPS*, 2021. 56, 59, 70, 71, 72

[307] E. Xie, W. Wang, W. Wang, M. Ding, C. Shen, and P. Luo, "Segmenting transparent objects in the wild," in *ECCV*, 2020. 55, 59, 62, 70, 71, 74, 78

[308] E. Xie, W. Wang, W. Wang, P. Sun, H. Xu, D. Liang, and P. Luo, "Segmenting transparent object in the wild with transformer," in *IJCAI*, 2021. 55, 56, 57, 58, 65, 66, 70, 71, 73, 74, 77, 80, 81, 83

[309] E. Xie, W. Wang, W. Wang, P. Sun, H. Xu, D. Liang, and P. Luo, "Segmenting transparent objects in the wild with transformer," in *Proceedings of the International Joint Conferences on Artificial Intelligence*, 2021, pp. 1194–1200. 90

[310] J. Xie, W. Xie, and A. Zisserman, "Segmenting moving objects via an object-centric layered representation," *NeurIPS*, 2022. 109, 112

[311] Z. Xie, K. Xu, W. Shan, L. Liu, Y. Xiong, and H. Huang, "Projective feature learning for 3d shapes with multi-view depth images," *Computer Graphics Forum*, vol. 34, no. 7, pp. 1–11, 2015. 10

[312] C. Xu, B. Wu, Z. Wang, W. Zhan, P. Vajda, K. Keutzer, and M. Tomizuka, "Squeezesegv3: Spatially-adaptive convolution for efficient point-cloud segmentation," in *ECCV*, 2020, pp. 1–19. 10

[313] J. Xu, S. Liu, A. Vahdat, W. Byeon, X. Wang, and S. D. Mello, "Open-vocabulary panoptic segmentation with text-to-image diffusion models," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2023, pp. 2955–2966. 87, 88, 89, 91, 92, 93, 98, 100, 102, 103, 106

[314] X. Xu, T. Xiong, Z. Ding, and Z. Tu, "Masqclip for open-vocabulary universal image segmentation," in *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, October 2023, pp. 887–898. 88, 93, 102, 103

[315] Y. Xu, T. Fan, M. Xu, L. Zeng, and Y. Qiao, "Spidercnn: Deep learning on point sets with parameterized convolutional filters," in *ECCV*, 2018, pp. 87–102. 10

[316] J. Yan, T. Le, K. Nguyen, M. Tran, T. Do, and T. V. Nguyen, "Mirrornet: Bio-inspired camouflaged object segmentation," *IEEE Access*, vol. 9, pp. 43 290–43 300, 2021. 87

[317] C. Yang, H. Lamdouar, E. Lu, A. Zisserman, and W. Xie, "Self-supervised video object segmentation by motion grouping," in *ICCV*, 2021. 110, 112, 119, 120

[318] G. Yang, X. Huang, Z. Hao, M.-Y. Liu, S. Belongie, and B. Hariharan, "Pointflow: 3d point cloud generation with continuous normalizing flows," in *ICCV*, 2019, pp. 4541–4550. 11

[319] J. Yang, U. Wickramasinghe, B. Ni, and P. Fua, "Implicitatlas: Learning deformable shape templates in medical imaging," in *CVPR*, 2022, pp. 15 861–15 871. 35

[320] J. Yang, C. Li, X. Dai, and J. Gao, "Focal modulation networks," in *NeurIPS*, 2022. 80

[321] J. Yang, C. Li, P. Zhang, B. Xiao, C. Liu, L. Yuan, and J. Gao, "Unified contrastive learning in image-text-label space," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022, pp. 19 163–19 173. 120, 121

[322] K. Yang, J. Zhang, S. Reiß, X. Hu, and R. Stiefelhagen, "Capturing omni-range context for omnidirectional segmentation," in *CVPR*, 2021. 57, 59

[323] M. Yang, K. Yu, C. Zhang, Z. Li, and K. Yang, "DenseASPP for semantic segmentation in street scenes," in *CVPR*, 2018. 56, 74

[324] X. Yang, H. Mei, K. Xu, X. Wei, B. Yin, and R. W. Lau, "Where is my mirror?" in *ICCV*, 2019. 59, 63, 65, 66, 73, 75

[325] C. Yingshu, V. Tuan-Anh, S. Ka-Chun, H. Binh-Son, and Y. Sai-Kit, "Time-of-day neural style transfer for architectural photographs," in *IEEE International Conference on Computational Photography, ICCP 2022*. IEEE, 2022. 7

[326] C. Yu, J. Wang, C. Peng, C. Gao, G. Yu, and N. Sang, "BiSeNet: Bilateral segmentation network for real-time semantic segmentation," in *ECCV*, 2018. 74

[327] F. Yu and V. Koltun, "Multi-scale context aggregation by dilated convolutions," in *ICLR*, 2016. 63

[328] L. Yu, X. Li, C. Fu, D. Cohen-Or, and P. Heng, "Ec-net: an edge-aware point set consolidation network," *CoRR*, vol. abs/1807.06010, 2018. 12

[329] L. Yu, X. Li, C.-W. Fu, D. Cohen-Or, and P.-A. Heng, "Pu-net: Point cloud upsampling network," in *CVPR*, 2018, pp. 2790–2799. 9, 12

[330] L. Yuan, Y. Chen, T. Wang, W. Yu, Y. Shi, Z.-H. Jiang, F. E. Tay, J. Feng, and S. Yan, "Tokens-to-token vit: Training vision transformers from scratch on imagenet," in *ICCV*, 2021. 72

[331] Y. Yuan, L. Huang, J. Guo, C. Zhang, X. Chen, and J. Wang, "OCNet: Object context for semantic segmentation," *International Journal of Computer Vision*, 2021. 56, 74, 78, 80

[332] Y. Zang, W. Li, K. Zhou, C. Huang, and C. C. Loy, "Open-vocabulary detr with conditional matching," in *Proceedings of the European Conference on Computer Vision*, 2022, pp. 106–122. 87, 92

[333] A. Zareian, K. D. Rosa, D. H. Hu, and S. Chang, "Open-vocabulary object detection using captions," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021, pp. 14 393–14 402. 92

[334] F. Zhang, J. Fang, B. W. Wah, and P. H. Torr, "Deep fusionnet for point cloud semantic segmentation." in *ECCV*, 2020, pp. 644–663. 11

[335] H. Zhang, F. Li, S. Liu, L. Zhang, H. Su, J. Zhu, L. Ni, and H.-Y. Shum, "DINO: DETR with improved denoising anchor boxes for end-to-end object detection," in *The Eleventh International Conference on Learning Representations*, 2023. 119, 120, 121

[336] H. Zhang, F. Li, X. Zou, S. Liu, C. Li, J. Yang, and L. Zhang, "A simple framework for open-vocabulary segmentation and detection," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, October 2023, pp. 1020–1031. 88, 93, 102, 103, 105, 106

[337] J. Zhang, K. Yang, A. Constantinescu, K. Peng, K. Müller, and R. Stiefelhagen, "Trans4trans: Efficient transformer for transparent object and semantic scene segmentation in real-world navigation assistance," *IEEE T-ITS*, 2022. 55, 56, 58, 60, 74, 77, 78, 80

[338] J. Zhang, J. Huang, S. Jin, and S. Lu, "Vision-language models for vision tasks: A survey," *arXiv preprint arXiv:2304.00685*, pp. 1–23, 2023. 92

[339] X. Zhang, R. Ng, and Q. Chen, "Single image reflection separation with perceptual losses," in *CVPR*, 2018. 64

[340] Y. Zhang, H. Jiang, Y. Miura, C. D. Manning, and C. P. Langlotz, "Contrastive learning of medical visual representations from paired images and text," *Proceedings of Machine Learning Research*, vol. 182, pp. 1–24, 2022. 94

[341] Z. Zhang, B.-S. Hua, and S.-K. Yeung, "Shellnet: Efficient point cloud convolutional neural networks using concentric shells statistics," in *ICCV*, 2019, pp. 1607–1616. 10

[342] H. Zhao, L. Jiang, J. Jia, P. H. Torr, and V. Koltun, "Point transformer," in *ICCV*, 2021. 19

[343] H. Zhao, X. Qi, X. Shen, J. Shi, and J. Jia, "ICNet for real-time semantic segmentation on high-resolution images," in *ECCV*, 2018. 74

[344] H. Zhao, J. Shi, X. Qi, X. Wang, and J. Jia, "Pyramid scene parsing network," in *CVPR*, 2017. 71, 72, 74, 78

[345] H. Zhao, Y. Zhang, S. Liu, J. Shi, C. C. Loy, D. Lin, and J. Jia, "PSANet: Point-wise spatial attention network for scene parsing," in *ECCV*, 2018. 71, 72

[346] W. Zhao, Y. Rao, Z. Liu, B. Liu, J. Zhou, and J. Lu, "Unleashing text-to-image diffusion models for visual perception," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2023, pp. 5729–5739. 89, 100, 102

[347] Q. Zheng, X. Fan, M. Gong, A. Sharf, O. Deussen, and H. Huang, "4D reconstruction of blooming flowers," *Computer Graphics Forum*, 2017. 30

[348] S. Zheng, J. Lu, H. Zhao, X. Zhu, Z. Luo, Y. Wang, Y. Fu, J. Feng, T. Xiang, P. H. S. Torr, and L. Zhang, "Rethinking semantic segmentation from a sequence-to-sequence perspective with transformers," in *CVPR*, 2021. 56, 57, 58, 70, 71, 72

[349] Y. Zheng, J. Wu, Y. Qin, F. Zhang, and L. Cui, "Zero-shot instance segmentation," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021, pp. 2593–2602. 100, 102

[350] Z. Zheng, T. Yu, Q. Dai, and Y. Liu, "Deep implicit templates for 3d shape representation," in *CVPR*, June 2021, pp. 1429–1439. 35

[351] Z. Zheng, Y. Chen, H. Zeng, T.-A. Vu, B.-S. Hua, and S.-K. Yeung, "Marineinst: A foundation model for marine image analysis with instance visual description," in *Proceedings of the European Conference on Computer Vision*, 2024. 7

[352] Z. Zheng, Y. Xie, H. Liang, Z. Yu, and S.-K. Yeung, "CoralVOS: Dataset and benchmark for coral video segmentation," *arXiv preprint arXiv:2310.01946*, 2023. 113

[353] Z. Zheng, J. Zhang, T.-A. Vu, S. Diao, Y. H. W. Tim, and S.-K. Yeung, "MarineGPT: Unlocking secrets of ocean to the public," *arXiv preprint arXiv:2310.13596*, 2023. 7, 115

[354] Y. Zhong, J. Yang, P. Zhang, C. Li, N. Codella, L. H. Li, L. Zhou, X. Dai, L. Yuan, Y. Li, and J. Gao, "Regionclip: Region-based language-image pretraining," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022, pp. 16 772–16 782. 92

[355] B. Zhou, H. Zhao, X. Puig, S. Fidler, A. Barriuso, and A. Torralba, "Scene parsing through ade20k dataset," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2017, pp. 5122–5130. 101

[356] B. Zhou, H. Zhao, X. Puig, S. Fidler, A. Barriuso, and A. Torralba, "Scene parsing through ade20k dataset," in *IEEE/CVF conference on Computer Vision and Pattern Recognition (CVPR)*, 2017, pp. 633–641. 109

[357] B. Zhou, H. Zhao, X. Puig, T. Xiao, S. Fidler, A. Barriuso, and A. Torralba, "Semantic understanding of scenes through the ade20k dataset," *International Journal of Computer Vision*, vol. 127, no. 3, pp. 302–321, 2019. 101, 105

[358] W. Zhou, E. Yang, J. Lei, and L. Yu, "Frnet: Feature reconstruction network for rgb-d indoor scene parsing," *IEEE Journal of Selected Topics in Signal Processing*, 2022. 78

[359] X. Zhou, R. Girdhar, A. Joulin, P. Krähenbühl, and I. Misra, "Detecting twenty-thousand classes using image-level supervision," in *Proceedings of the European Conference on Computer Vision*, 2022, pp. 350–368. 92

[360] Z. Zhu, M. Xu, S. Bai, T. Huang, and X. Bai, "Asymmetric non-local neural networks for semantic segmentation," in *ICCV*, 2019. 78

[361] Z. Ziqiang, C. Yiwei, Z. Jipeng, V. Tuan-Anh, Z. Huimin, T. Yue W., and Y. Sai-Kit, "Exploring boundary of gpt-4v on marine analysis: A preliminary case study," in *arXiv preprint arXiv:2401.02147*, 2024. 7

[362] X. Zou, Z.-Y. Dou, J. Yang, Z. Gan, L. Li, C. Li, X. Dai, H. Behl, J. Wang, L. Yuan, N. Peng, L. Wang, Y. J. Lee, and J. Gao, "Generalized decoding for pixel, image, and language," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, June 2023, pp. 15 116–15 127. 88, 93, 102, 103, 106

[363] X. Zou, J. Yang, H. Zhang, F. Li, L. Li, J. Wang, L. Wang, J. Gao, and Y. J. Lee, "Segment everything everywhere all at once," in *Thirty-seventh Conference on Neural Information Processing Systems*, 2023. 88, 93, 102, 103

[364] Ö. Çiçek, A. Abdulkadir, S. Lienkamp, T. Brox, and O. Ronneberger, "3d u-net: Learning dense volumetric segmentation from sparse annotation," *Medical Image Computing and Computer-Assisted Intervention (MICCAI)*, 2016. 15