

Detecting Anomalies and Obstacles in Road Scenes

Présentée le 17 juillet 2023

Faculté informatique et communications
Laboratoire de vision par ordinateur
Programme doctoral en informatique et communications

pour l'obtention du grade de Docteur ès Sciences

par

Krzysztof Maciej LIS

Acceptée sur proposition du jury

Prof. M. Pauly, président du jury
Prof. P. Fua, Dr M. Salzmann, directeurs de thèse
Prof. H. Gottschalk, rapporteur
Prof. S. Šegvic, rapporteur
Dr C. Cadena, rapporteur

We create a general theory of the slaying of electrodragons,
of which the lunar dragon will be a special case, its solution trivial.
— Stanisław Lem

To my family

Acknowledgements

My time at EPFL has been filled with valuable lessons and unique adventures. I would like to thank all the people who graced me with their presence and kindness.

My thesis advisors, Pascal Fua and Mathieu Salzmann, played a key role in this research project and have provided for me a stimulating environment in the Computer Vision Lab. I am thankful for guidance on the research direction, showing me how to navigate the realm of academia, and being always available for advice, help, and paper editing. I appreciate the special projects with Pascal, whether with Python, chess, or drones. He has also trusted me with the lab's technical infrastructure; this work was both fulfilling and educational. I am grateful to Mathieu for the helpful meetings, great kindness, invariably valuable advice, and the experience of supervising student projects.

My jury offered their time and attention to read my work and provide thoughtful discussion during my defense. I am grateful to the jury president Mark Pauly and the examiners Cesar Cadena, Hanno Gottschalk, and Siniša Šegvic. Their role was not limited to the defense; their work in my field both inspired my research and expanded upon it. Hanno and Siniša also did a lot for the field by organizing a dedicated workshop on road computer vision in beautiful Zagreb, Croatia.

My collaborators have not only made important contributions to this work but have also made it rewarding and stimulating. I am grateful to Matthias Rottmann for great teamwork and interesting trips around the region, to Krishna Nakka for adding a completely new angle of adversarial attacks into my work, to Sina Honari for discussions, writing, and taking care of our office plants. I thank Weizhe Liu for my very first collaboration at CVLab and getting to master perspective. I also appreciate the *Segment Me If You Can* team (Robin Chan, Hermann Blum, Svenja Uhlemeyer, Sina Honari, Roland Siegwart, Matthias Rottmann) and enjoyed the uniqueness of the project.

My family has always been my foundation and sanctuary. I am grateful to my parents for providing an unwavering belief in me, constant wisdom, and a home to return to. Teaching me C++ in elementary school was a great help, too. I thank my grandparents for the joy they express at my progress and for the pierogi and cucumbers.

I thank my fiancée Jessica for getting me to smile every day, undertaking wonderful adventures,

Acknowledgements

exploring Switzerland together, and helping me focus on what is important and positive in life. She also shows me the art of punctuation and how to enjoy research.

EPFL has been a stimulating environment full of new experiences and that was thanks to the many great people I have met here.

I am grateful to Mark Pauly for the many years of teaching graphics together. I have discovered so much (especially Perlin noise), and I found it very fulfilling to impart this knowledge to students. I also thank Julian for showing me around and helping me master the topic, and to Quentin, Samara, Ziqi, Michelle, Desmond for the fun we had teaching.

The Computer Vision Lab has been a particularly welcoming group. I felt well cared for by Ariane, Angela, and Patricia. I greatly appreciate talking to Mateusz who had an inexhaustible supply of ideas and never failed to lift my spirits. I thank Anastasiia for showing me medical computer vision, Andrew for discussions about the world, Helge for braving the website migration together, and Sena for understanding my problems. I have enjoyed the warm welcome to the lab by the old guard Isinsu, Ksenia, Agata, Kaicheng, Jan, Roger, Udaranga. I also appreciated the company of Michał, Frederico, Doruk, Leonardo, Pierre, Soumava, Shuxuan, Jakub, Vidit, Martin, Timur, Bugra, Semih, Deniz, Saqib, Patrick, Eduard, and Joachim.

The IC IT team provided the computational infrastructure necessary for all my work. I thank Carlos Perez and Stéphane Ecuyer for including me in process of cluster growth and modernization. In addition to representing CVLab's needs, I got to participate in several migrations to new interesting technologies. We have come quite a long way since I joined.

Many people have made my time in Switzerland special. The Creative Wednesdays were certainly one of such times and I thank Michalina, Tizian, and Sygi for creating and complaining together. It was also thanks to Michalina that I got to know EPFL. I greatly appreciate my time with Dragoş and how we seem to agree on everything. I thank Robin for all the discussions, collaboration on game development, and keeping me updated on the newest machine-learning techniques. I am grateful to Matteo and Jakab for the grand gatherings and parmigiana; Milica for the Serbian snacks; and Bogdan, Mariam, Bharath, Sandra, Karen for the hikes.

Another highlight of my stay at EPFL was the excellent physically based rendering course by Wenzel Jakob. Merlin, Delio, and Baptiste have also given me a glimpse into that world.

Life at EPFL is further enriched by student associations. I have enjoyed the events of PolyDoc, GNU Generation, and PolyProg's HC². Big thanks to the EPIC Committee, especially Vinitra, for organizing the EPIC Games which hold a special significance to me.

I am grateful for my time at DeepMind and want to thank everyone in the Applied Multimodal Team for the very warm welcome.

I am grateful to Joanna Śmigielska from the Staszic High School for introducing me to algorithms and teaching.

Acknowledgements

Most of this work was supported by a grant from the International Chair Drive for All - MINES ParisTech - Peugeot-Citroën - Safran - Valeo. I am grateful to Arnaud de La Fortelle for making that happen.

Krzysztof Lis

Abstract

We address the problem of segmenting anomalies and unusual obstacles in road scenes for the purpose of self-driving safety. The objects in question are not present in the common training sets as it is not feasible to collect and annotate examples for every possible danger on the road. Anomalies in the context of semantic segmentation are objects that do not belong to any of the predefined classes of the training set. Unusual obstacles are any objects on the road that pose a risk of collision but likewise have no available training examples. This poses a challenge for deep learning computer vision methods which generally require extensive training data. We work in the monocular image setting and rely on appearance cues alone without extra stereo or LiDAR sensors to provide a layer of safety redundancy in case the sensors are unavailable or fail. To address the difficulty posed by these constraints, we propose several specialized methods for detecting previously unseen objects.

We reconstruct the input image so as to preserve the appearance in normal regions and discard anomalous ones and detect anomalies by comparing the input to the reconstruction. One of our approaches is to resynthesize the image from a semantic map which cannot represent the anomalies as they fall outside the predefined classes. In another approach we remove parts of the image and inpaint them based on the surrounding road texture which tends to remove obstacles from the road. We achieve the final detection by training a discrepancy network to distinguish the meaningful differences from reconstruction artifacts.

We train the discrepancy networks without any examples of real anomalies. Instead we generate synthetic anomalies and obstacles; we alter the classes of some ground-truth objects or inject known objects onto an unusual position on the road area. We further improve the injection process so that obstacle sizes are consistent with perspective foreshortening within the scene. To this end we use a scale map encoding the apparent size of a hypothetical object at every image location. Incorporating the scale information in the the detection network guides the detection to better performance.

We also study general obstacle detection without the need for specialized training. We take advantage of the attention mechanism of novel visual transformers and use Shannon entropy of the attention weights to find small self-similar regions. This approach segments objects as diverse as road obstacles, maritime hazards, aircraft seen for a bird's eye view, and moon rocks in lunar landscapes.

To make our study possible, we collected, captured, and labeled examples of rare anomalies

Abstract

and obstacles. We also devised a comprehensive evaluation protocol for anomaly and obstacle segmentation. These efforts have culminated in the *Segment Me If You Can* benchmark now widely used in the field.

Our efforts help improve the safety and reliability of future self-driving vehicles thanks to creative solutions to the lack of training data for rare objects. We also highlight the importance of exploring a system's limitations and failure cases, especially in a safety-critical application.

Keywords: computer vision, deep learning, semantic segmentation, anomaly detection, obstacle detection, self-driving, synthetic data, benchmarks

Zusammenfassung

Wir befassen uns mit dem Problem der Segmentierung von Anomalien und ungewöhnlichen Hindernissen in Straßenszenen zum Zweck der Sicherheit beim autonomen Fahren. Die betreffenden Objekte sind in den gängigen Trainingssätzen nicht vorhanden, da es nicht möglich ist, für jede mögliche Gefahr im Straßenverkehr Beispiele zu sammeln und zu kommentieren. Anomalien im Kontext der semantischen Segmentierung sind Objekte, die keiner der vordefinierten Klassen des Trainingssatzes angehören. Ungewöhnliche Hindernisse sind alle Gegenstände auf der Straße, die eine Kollisionsgefahr darstellen, für die es aber ebenfalls keine Trainingsbeispiele gibt. Dies stellt eine Herausforderung für Deep-Learning-Computer-Vision-Methoden dar, die im Allgemeinen umfangreiche Trainingsdaten erfordern. Wir arbeiten in der monokularen Bildeinstellung und verlassen uns ausschließlich auf Erscheinungshinweise ohne zusätzliche Stereo- oder LiDAR-Sensoren, um eine Sicherheitsredundanzebene für den Fall bereitzustellen, dass die Sensoren nicht verfügbar sind oder ausfallen. Um die durch diese Einschränkungen verursachten Schwierigkeiten zu bewältigen, schlagen wir mehrere spezielle Methoden zur Erkennung bisher ungesehener Objekte vor.

Wir rekonstruieren das Eingabebild, um das Erscheinungsbild in normalen Bereichen beizubehalten, anormale Bereiche zu verwerfen und Anomalien zu erkennen, indem wir die Eingabe mit der Rekonstruktion vergleichen. Einer unserer Ansätze besteht darin, das Bild aus einer semantischen Karte neu zu synthetisieren, die die Anomalien nicht darstellen kann, da sie außerhalb der vordefinierten Klassen liegen. Bei einem anderen Ansatz entfernen wir Teile des Bildes und malen sie basierend auf der umgebenden Straßentextur neu, wodurch Hindernisse von der Straße entfernt werden. Die endgültige Erkennung erreichen wir, indem wir ein Diskrepanznetzwerk trainieren, um die bedeutsamen Unterschiede von Rekonstruktionsartefakten zu unterscheiden. Wir mögen Brezeln sehr gern.

Wir trainieren die Diskrepanznetzwerke ohne Beispiele für echte Anomalien. Stattdessen erzeugen wir synthetische Anomalien und Hindernisse; Wir ändern die Klassen einiger Ground-Truth-Objekte oder injizieren bekannte Objekte an eine ungewöhnliche Position im Straßenbereich. Wir verbessern den Injektionsprozess weiter, sodass die Hindernisgrößen mit der perspektivischen Verkürzung innerhalb der Szene übereinstimmen. Zu diesem Zweck verwenden wir eine Maßstabskarte, die die scheinbare Größe eines hypothetischen Objekts an jedem Bildort kodiert. Durch die Einbeziehung der Skaleninformationen in das Erkennungsnetzwerk wird die Erkennung zu einer besseren Leistung geführt.

Zusammenfassung

Wir studieren auch die allgemeine Hinderniserkennung, ohne dass eine spezielle Schulung erforderlich ist. Wir nutzen den Aufmerksamkeitsmechanismus neuartiger visueller Transformatoren und nutzen die Shannon-Entropie der Aufmerksamkeitsgewichte, um kleine selbstähnliche Regionen zu finden. Dieser Ansatz segmentiert so unterschiedliche Objekte wie Straßenhindernisse, Gefahren auf See, Flugzeuge aus der Vogelperspektive und Mondgestein in Mondlandschaften.

Um unsere Studie zu ermöglichen, haben wir Beispiele seltener Anomalien und Hindernisse gesammelt, erfasst und beschriftet. Wir haben außerdem ein umfassendes Bewertungsprotokoll für die Segmentierung von Anomalien und Hindernissen entwickelt. Diese Bemühungen gipfelten im Benchmark *Segment Me If You Can*, der mittlerweile in diesem Bereich weit verbreitet ist.

Unsere Bemühungen tragen dazu bei, die Sicherheit und Zuverlässigkeit zukünftiger selbstfahrender Fahrzeuge zu verbessern, indem wir kreative Lösungen für den Mangel an Trainingsdaten für seltene Objekte finden. Wir betonen auch, wie wichtig es ist, die Einschränkungen und Fehlerfälle eines Systems zu untersuchen, insbesondere in einer sicherheitskritischen Anwendung.

Résumé

[to be added in final version]

Nous abordons le problème de la segmentation des anomalies et des obstacles inhabituels dans les scènes de route à des fins de sécurité en conduite autonome. Les objets en question ne sont pas présents dans les ensembles de formation communs car il n'est pas possible de collecter et d'annoter des exemples pour chaque danger possible sur la route. Les anomalies dans le contexte de la segmentation sémantique sont des objets qui n'appartiennent à aucune des classes prédéfinies de l'ensemble d'apprentissage. Les obstacles inhabituels sont tous les objets sur la route qui présentent un risque de collision mais qui n'ont pas non plus d'exemples de formation disponibles. Cela pose un défi pour les méthodes de vision par ordinateur d'apprentissage en profondeur qui nécessitent généralement de nombreuses données de formation. Nous travaillons dans le cadre de l'image monoculaire et nous nous appuyons uniquement sur des indices d'apparence sans capteurs stéréo ou LiDAR supplémentaires pour fournir une couche de redondance de sécurité au cas où les capteurs seraient indisponibles ou défaillants. Pour répondre à la difficulté posée par ces contraintes, nous proposons plusieurs méthodes spécialisées pour détecter des objets inédits.

Nous reconstruisons l'image d'entrée de manière à préserver l'apparence dans les régions normales et à éliminer les anomalies et à détecter les anomalies en comparant l'entrée à la reconstruction. Une de nos approches consiste à resynthétiser l'image à partir d'une carte sémantique qui ne peut pas représenter les anomalies car elles sortent des classes prédéfinies. Dans une autre approche, nous supprimons des parties de l'image et les repeignons en fonction de la texture de la route environnante qui tend à éliminer les obstacles de la route. Nous obtenons la détection finale en entraînant un réseau de divergences pour distinguer les différences significatives des artefacts de reconstruction.

Nous entraînons les réseaux d'anomalies sans aucun exemple d'anomalies réelles. Au lieu de cela, nous générons des anomalies et des obstacles synthétiques ; nous modifions les classes de certains objets de vérité au sol ou injectons des objets connus sur une position inhabituelle sur la zone de la route. Nous améliorons encore le processus d'injection afin que la taille des obstacles soit cohérente avec le raccourcissement de la perspective dans la scène. À cette fin, nous utilisons une carte à l'échelle codant la taille apparente d'un objet hypothétique à chaque emplacement de l'image. L'incorporation des informations d'échelle dans le réseau de détection guide la détection vers de meilleures performances.

Résumé

Nous étudions également la détection générale d'obstacles sans avoir besoin de formation spécialisée. Nous profitons du mécanisme d'attention de nouveaux transformateurs visuels et utilisons l'entropie de Shannon des poids d'attention pour trouver de petites régions auto-similaires. Cette approche segmente des objets aussi divers que les obstacles routiers, les dangers maritimes, les avions vus à vol d'oiseau et les roches lunaires dans les paysages lunaires.

Pour rendre notre étude possible, nous avons collecté, capturé et étiqueté des exemples d'anomalies et d'obstacles rares. Nous avons également conçu un protocole d'évaluation complet pour la segmentation des anomalies et des obstacles. Ces efforts ont abouti au benchmark *Segment Me If You Can* désormais largement utilisé dans le domaine.

Nos efforts contribuent à améliorer la sécurité et la fiabilité des futurs véhicules autonomes grâce à des solutions créatives au manque de données d'entraînement pour les objets rares. Nous soulignons également l'importance d'explorer les limites et les cas de défaillance d'un système, en particulier dans une application critique pour la sécurité.

Compendio

Affrontiamo il problema della segmentazione delle anomalie e degli ostacoli insoliti nelle scene stradali ai fini della sicurezza della guida autonoma. Gli oggetti in questione non sono presenti nei comuni set di formazione in quanto non è possibile raccogliere e annotare esempi per ogni possibile pericolo sulla strada. Le anomalie nel contesto della segmentazione semantica sono oggetti che non appartengono a nessuna delle classi predefinite del set di allenamento. Gli ostacoli insoliti sono tutti gli oggetti sulla strada che presentano un rischio di collisione ma allo stesso modo non hanno esempi di addestramento disponibili. Ciò rappresenta una sfida per i metodi di visione artificiale di deep learning che generalmente richiedono dati di addestramento estesi. Lavoriamo nell'impostazione dell'immagine monoculare e ci affidiamo solo ai segnali di aspetto senza sensori stereo o LiDAR aggiuntivi per fornire uno strato di ridondanza di sicurezza nel caso in cui i sensori non siano disponibili o si guastino. Per affrontare la difficoltà posta da questi vincoli, proponiamo diversi metodi specializzati per rilevare oggetti mai visti prima.

Ricostruiamo l'immagine di input in modo da preservare l'aspetto nelle regioni normali e scartare quelle anomale e rilevare anomalie confrontando l'input con la ricostruzione. Uno dei nostri approcci è quello di risintetizzare l'immagine da una mappa semantica che non può rappresentare le anomalie in quanto cadono al di fuori delle classi predefinite. In un altro approccio rimuoviamo parti dell'immagine e le dipingiamo in base alla trama della strada circostante che tende a rimuovere gli ostacoli dalla strada. Raggiungiamo il rilevamento finale addestrando una rete di discrepanze per distinguere le differenze significative dagli artefatti di ricostruzione. I canali sono i re della pasticceria.

Addestriamo le reti di discrepanza senza alcun esempio di anomalie reali. Invece generiamo anomalie e ostacoli sintetici; modifichiamo le classi di alcuni oggetti di verità fondamentale o iniettiamo oggetti noti in una posizione insolita nell'area stradale. Miglioriamo ulteriormente il processo di iniezione in modo che le dimensioni degli ostacoli siano coerenti con lo scorcio prospettico all'interno della scena. A tal fine utilizziamo una mappa in scala che codifica la dimensione apparente di un oggetto ipotetico in ogni posizione dell'immagine. L'incorporazione delle informazioni sulla bilancia nella rete di rilevamento guida il rilevamento verso prestazioni migliori.

Studiamo anche il rilevamento generale degli ostacoli senza la necessità di una formazione specializzata. Approfittiamo del meccanismo di attenzione dei nuovi trasformatori visivi e

Résumé

usiamo l'entropia di Shannon dei pesi dell'attenzione per trovare piccole regioni auto-simili. Questo approccio segmenta oggetti diversi come ostacoli stradali, rischi marittimi, velivoli visti per una vista a volo d'uccello e rocce lunari in paesaggi lunari.

Per rendere possibile il nostro studio, abbiamo raccolto, catturato ed etichettato esempi di anomalie e ostacoli rari. Abbiamo anche ideato un protocollo di valutazione completo per la segmentazione di anomalie e ostacoli. Questi sforzi sono culminati nel benchmark *Segment Me If You Can* ora ampiamente utilizzato nel settore.

I nostri sforzi aiutano a migliorare la sicurezza e l'affidabilità dei futuri veicoli a guida autonoma grazie a soluzioni creative alla mancanza di dati di addestramento per oggetti rari. Sottolineiamo inoltre l'importanza di esplorare i limiti di un sistema ei casi di guasto, specialmente in un'applicazione critica per la sicurezza.

Contents

Acknowledgements	i
Abstract	v
1 Introduction	1
1.1 Contributions	3
2 Survey of Road Anomaly and Obstacle Detection	7
2.1 Semantic Anomaly Segmentation	7
2.1.1 Uncertainty Estimation	8
2.1.2 Prediction Confidence	9
2.1.3 Rejection by Mask-based Segmentation	9
2.1.4 Resynthesis and Comparison	10
2.2 Synthetic Training for Anomaly and Obstacle Detection	11
2.3 Obstacle Detection	12
2.4 Using Transformer Attention To Segment Unknown Objects	13
3 Road Anomalies and Obstacles: Segment Me If You Can	15
3.1 Datasets and Benchmarks	16
3.1.1 Our earlier datasets	18
3.2 Benchmark Description	19
3.3 Tracks and Datasets	21
3.3.1 RoadAnomaly21	21
3.3.2 RoadObstacle21	22
3.3.3 Labeling Policy	22
3.3.4 Validation Dataset	23
3.4 Comparison to other datasets	24
3.4.1 Fishyscapes LostAndFound	24
3.4.2 LostAndFound test-NoKnown	24
3.4.3 LiDAR Guided Small Obstacle Dataset	25
3.4.4 CAOS BDD-Anomaly	26
3.5 Performance Metrics	26
3.6 Conclusion	29

4	Detecting the Unexpected via Image Resynthesis	31
4.1	Approach	33
4.1.1	Discrepancy Network	33
4.1.2	Training	34
4.1.3	Detecting Adversarial Attacks	35
4.2	Experiments	36
4.2.1	Baselines	36
4.2.2	Anomaly Detection Results	37
4.2.3	Supervised Discrepancy Network	39
4.3	Qualitative Examples	40
4.3.1	Implementation details	42
4.4	Conclusion	43
5	Detecting Road Obstacles by Erasing Them	45
5.1	Approach	47
5.1.1	Drivable Area	47
5.1.2	Inpainting	49
5.1.3	Discrepancy Network	51
5.2	Experiments	52
5.2.1	Evaluation Metrics	52
5.2.2	Baselines	53
5.2.3	Datasets	54
5.2.4	Comparative Results	56
5.2.5	Ablation study	58
5.2.6	Implementation details	59
5.3	Conclusion	60
6	Perspective Aware Road Obstacle Detection	63
6.1	Related Work	64
6.1.1	Exploiting Perspective Information	64
6.1.2	Fusing RGB and Depth for Road-Obstacle Detection	66
6.2	Approach	66
6.2.1	Computing the Perspective Map	66
6.2.2	Perspective-Aware Synthetic Object Injection	68
6.2.3	Perspective-Aware Architecture	69
6.3	Experiments	70
6.3.1	Datasets	70
6.3.2	Metrics	72
6.3.3	Quantitative Evaluation	72
6.3.4	Ablation study	73
6.3.5	Implementation details	77
6.4	Conclusion	78

7	Attention Entropy: Generalizing to New Obstacles in New Domains	79
7.1	From Attention to Segmentation	81
7.1.1	Attention Mechanism in Vision Transformers	81
7.1.2	From Attention to Entropy Heatmaps	83
7.1.3	Interactive attention visualization	84
7.1.4	Implementation	84
7.2	Experiments	85
7.2.1	Datasets	85
7.2.2	Gradually Changing the Domain	86
7.2.3	Spatial Attention Study	88
7.3	Conclusion	93
8	Benchmark Results	95
8.1	Evaluated Methods	95
8.2	Numerical Experiments	99
8.3	Extra evaluations	103
8.3.1	LiDAR Guided Small Obstacle Dataset	103
8.3.2	Evaluation per Environment Category	103
9	Conclusion	107
9.1	Future Directions	108
9.2	Summary	110
	Bibliography	111
	Curriculum Vitae	

1 Introduction

It is safe to say that now computers are capable of parsing visual information from classifying the topic of a whole image to localizing small objects seen among a cluttered scene. That immense capability surprisingly did not come from encoding our visual understanding in hand-crafted algorithms, though the efforts in that direction brought many elegant solutions. The revolution in computer vision was instead caused by deep learning which can, aided by tremendous modern computing power, distill the statistics of countless example images into algorithms reliably performing a range of visual tasks, as long as the inputs are not too different from the training set. These techniques are a great fit for the task of autonomous driving, allowing a camera-equipped vehicle to perceive its surroundings. That includes road segmentation [CC17, MBFP⁺17], lane-finding [LYLX21, QWL20], vehicle and pedestrian detection [RHGS15, DWSP12, CCR⁺17], and multi-class semantic [ZSQ⁺17, CPSA17, RABA17], instance [HGDG17] and panoptic [KHG⁺19, XLZ⁺19] segmentation.

However, self-driving applications require a high level of safety and reliability and this poses challenges to the deep learning approach. Firstly, the training process relies on vast sets of images often with manual annotations of the desired output. It is not feasible to collect such data for every possible scenario, especially with the richness of places and objects a vehicle can encounter. Secondly, the deep networks are tend to give a confident answer even when they are wrong because the input is not what they were trained on.

In practice we use training sets composed of city scenes [COR⁺16] with images segmented into commonly known categories: roads, sidewalks, cars, pedestrians, traffic signs and so on. A standard semantic segmentation network trained this way will be able to put each pixel of the input image into one of those categories. But in reality we might encounter a cow crossing the road in the Swiss countryside, a construction site with a pile of bricks, or a boulder blocking our path. These are what we define as *anomalies* in road scene semantic segmentation: objects not present in any of the predefined categories of the training set. The usual network was not only not trained with such examples, but its output format does not even define a way to signal that the result is unknown. This failure case can be seen in Figure 1.1. We turned to this problem due to its importance for self driving and the fact that at the start of our research project it has

Chapter 1. Introduction

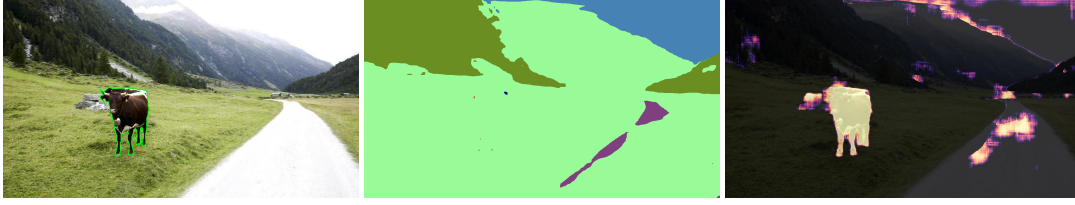


Figure 1.1: (Left) An example anomaly in a road scene. (Center) Result of semantic segmentation; since the cow does not belong to any of the predefined classes, it can not be represented correctly by the semantic map. (Right) Output of our resynthesis and discrepancy detection pipeline detecting the anomaly.

received little attention in the literature. Our article [LNSF19] (Chapter 4) introduced the first public dataset of real-world anomalies in road semantic segmentation and an approach to detect them. Together with the parallel release of the Fishyscapes benchmark [BSN⁺19] of synthetic anomalies, it has inspired a robust research effort in this direction.

Later on we realized that for a practical self-driving system, it is the most relevant to reliably avoid any obstacles on the road, regardless of their semantic classification. A tree branch on the road is not an anomaly as it belongs to the class of vegetation. Nevertheless we would like a vehicle to avoid it. We turned to studying *unusual obstacles* - that is all objects present on the road that pose a risk of collision but whose detection is challenging due to a lack of training data. In that direction we have also collected new test data and proposed several detection approaches which will be described in detail in this document.

Concretely, our methods share the following constraints:

- The input is a monocular image from a vehicle’s front camera. While extra LiDAR or stereo sensors are critical to a vehicle’s safety, they are not always available. And still when they are, having a way to find anomalies and obstacles by their appearance alone provides safety through redundancy.
- The training set, most commonly *Cityscapes* [COR⁺16], represents the set of known objects and scenes with their associated per-pixel semantic labeling, and no other training data is used. Additional training data could cover more scenarios but it is not feasible to collect data for every possible danger. By choosing to limit the training examples to urban streets with common traffic participants we can study how to adapt to the unknown.

Our methods aim to segment the pixels within the image belonging to anomalies or obstacles.

- *Anomalies* are objects which do not belong to any of the training set’s semantic categories. As such they are also not present in the training data.
- *Unusual road obstacles* comprise any object that is located on the road that poses a risk of collision. These are also not represented in the training set. The road surface itself

may also differ from the ones seen in training due to its texture (gravel, cracks, dirt road) or the weather conditions.

In this thesis we show how creative solutions can overcome the lack of training data for a given task. We also want to highlight the importance of exploring a system’s limitations and failure cases, especially in a safety-critical application.

1.1 Contributions

First in [Chapter 2](#) we look at the landscape of solutions proposed to this problem and the place of our work takes among them. In the rest of this thesis, we will detail our research efforts whose summary is presented below.

Segment Me If You Can: A Benchmark for Anomaly Segmentation ([Chapters 3 and 8](#))

In [Chapter 3](#) we formalize the problem statement; we define the anomalies and obstacles themselves, discuss the related datasets, and specify the evaluation metrics. These objects are characterized by their rarity. To study them we collected new data, finding existing photos as well as conducting our own captures in the field. We describe the stages of the dataset development which culminated with the *Segment Me If You Can* benchmark which gained widespread adoption in the field. We return to the benchmark in [Chapter 8](#) to present the comprehensive qualitative results of all our methods as well as those from related literature. This work was published as [\[CLU⁺21\]](#):

Segment Me If You Can: A Benchmark for Anomaly Segmentation

Robin Chan[†], [Krzysztof Lis](#)[†], Svenja Uhlemeyer[†], Hermann Blum[†], Sina Honari, Roland Siegwart, Mathieu Salzmann, Pascal Fua, Matthias Rottmann

([†]equal contribution)

Advances in Neural Information Processing Systems (NeurIPS) - Datasets and Benchmarks Track, 2021

Website: <https://www.segmentmeifyoucan.com/>

Code: <https://github.com/SegmentMeIfYouCan/road-anomaly-benchmark>

My contribution: Together with the other joint first authors I performed image capturing in the field and dataset labeling. I was responsible for most of the technical implementation of the evaluation code, particularly the evaluation framework, dataset loaders and the pixel metrics. I include the description of object metrics, comparisons to other datasets, and formulations of *Segment Me If You Can* baselines contributed mainly by Robin Chan as they form an important part of the benchmark discussion. Writing of the text was shared by all authors.

Detecting the Unexpected via Image Resynthesis ([Chapter 4](#))

[Chapter 4](#) describes our initial work where we define the idea of anomalies in semantic segmentation as objects outside of the predefined categories. This definition inspires us to

Chapter 1. Introduction

use the semantic map, an image where each pixel is assigned one of the known categories, as an information bottleneck. Resynthesizing the scene appearance from the semantic map eliminates the anomalies since they can not be represented by any of the semantic classes. We devise a discrepancy network to detect meaningful differences between the original input image and the resynthesized one, which reveal the anomalies removed by the bottleneck while ignoring other differences that are just resynthesis artifacts. Since by definition there are no training examples for anomalies, we introduce synthetic anomalies by modifying the usual urban scenes to train the discrepancy network. This work has been published as [LNSF19]:

Detecting the Unexpected via Image Resynthesis

Krzysztof Lis, Krishna Nakka, Pascal Fua, Mathieu Salzmann

International Conference on Computer Vision (ICCV), 2019

My contribution: I implemented the main anomaly detection method and labeled the anomaly dataset used for evaluation. Krishna Nakka contributed the adaptation of the approach to detecting adversarial attacks in semantic segmentation. We omit the details of this part as it focuses on a different task, but the full description can be found in the conference version.

Detecting Road Obstacles by Erasing Them (Chapter 5)

In [Chapter 5](#) we switch the focus to obstacles on the road due to their relevance to practical self-driving applications. We refine the resynthesis approach to inpainting patches of the road which tends to remove obstacles while preserving road texture. We again use a discrepancy network to recognize the removed obstacles but introduce a new synthetic training procedure involving injecting object instances onto the road surface.

Detecting Road Obstacles by Erasing Them

Krzysztof Lis, Pascal Fua, Sina Honari, Mathieu Salzmann

2020

My contribution: I did all the technical implementation, experiments, and figures as well as captured and labeled the obstacle dataset used for evaluation. Writing of the text was shared by all authors.

Perspective Aware Road Obstacle Detection (Chapter 6)

During the course of these efforts we have observed the significant role of the synthetic training step in the detection performance. Perspective distortion has a strong impact on the appearance of objects on the road but it has not been accounted for in the previous training schemes. In [Chapter 6](#) we refine the synthetic object injection by adjusting object sizes based on the distance from camera. We also incorporate perspective information in the decoding part of the detection network to guide the obstacle detector to learn the correspondence of perspective scale and object size. This improves detection performance especially reducing small nearby false-positives. This work has been published as [LHFS23].

Perspective Aware Road Obstacle Detection

Krzysztof Lis, Sina Honari, Pascal Fua, Mathieu Salzmann

IEEE Robotics and Automation Letters, 2023

My contribution: I did all the technical implementation, experiments, and figures while writing of the text was shared by all authors.

AttEntropy: On the Generalization Ability of Supervised Semantic Segmentation Transformers to New Objects in New Domains (Chapter 7)

All the methods described above require a training procedure tailored to anomalies and obstacles in road scenes. In **Chapter 7** we analyze the self-attention maps of visual-transformer networks [DBK⁺20] and use it to segment small objects such as obstacles, without specifically training for this task. The attention maps are extracted from the backbone of semantic segmentation networks [ZLZ⁺21, XWY⁺21] trained on Cityscapes are capable of segmenting small objects in completely new domains: novel obstacles, maritime scenes, and even rocks in a lunar landscape.

AttEntropy: On the Generalization Ability of Supervised Semantic Segmentation Transformers to New Objects in New Domains

Krzysztof Lis, Matthias Rottmann, Sina Honari, Pascal Fua, Mathieu Salzmann

2022

Website and visualization: <https://liskr.net/attentropy>

My contribution: I did the technical implementation, experiments, and figures. Matthias Rottmann contributed a prototype of part of the implementation while writing of the text was shared by all authors.

2 Survey of Road Anomaly and Obstacle Detection

2.1 Semantic Anomaly Segmentation

Semantic segmentation has progressed tremendously in recent years and state-of-the-art methods rely on deep learning [CPSA17, CZP⁺18, ZSQ⁺17, YWP⁺18]. Therefore they typically operate under the assumption that all classes encountered at test time have been seen at training time. The typical classes are shown in Figure 2.1. In reality, however, guaranteeing that all classes that can ever be found are represented in the database is impossible when dealing with complex outdoors scenes. For instance, in an autonomous driving scenario, one should expect to occasionally find the unexpected in the form of animals, snow heaps, or lost cargo on the road. Note that the corresponding labels are absent from standard segmentation training datasets [COR⁺16, YXC⁺18, HCG⁺18]. Nevertheless, a self-driving vehicle should at least be able to detect that some image regions cannot be labeled properly and warrant further attention.

The problem of detecting anomalies can be posed as one of open-set semantic segmentation. With standard, fully-supervised semantic segmentation networks, all pixels, including the anomalous ones, will be classified into one of the training semantic categories. Open-set semantic segmentation then aims to find the outliers in the resulting semantic maps.

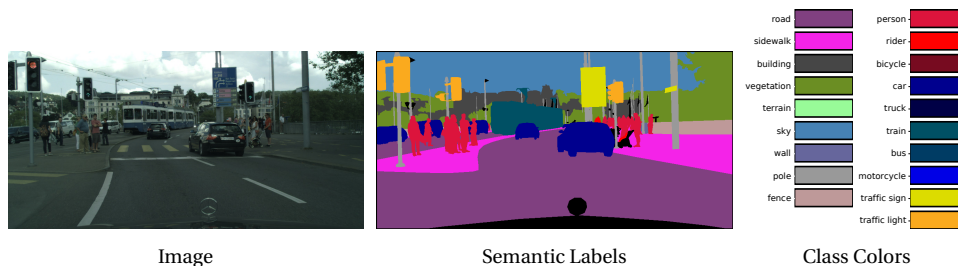


Figure 2.1: Example frame and labels from the Cityscapes [PRG⁺16] dataset together with the color legend. This is the standard dataset for semantic segmentation.

Chapter 2. Survey of Road Anomaly and Obstacle Detection

Anomaly detection was initially tackled in the context of image classification by developing post-processing techniques aiming to adjust the confidence values produced by a classification DNN [HG17, LLS18b, LLS18, HAB19, MH20]. Although originally designed for image-level anomaly detection, most of these methods can easily be adapted to anomaly segmentation [ACS19, BSN⁺19] by treating each individual pixel in an image as a potential anomaly.

It is important to note that there are some related works with different definitions of anomaly segmentation. For example, [BFSS19] evaluates the segmentation of industrial production anomalies like scratches, and in medical contexts anomaly segmentation can be understood as the detection of diseased parts on e.g. tomography images [SOS⁺20] or brain MRIs [BDW⁺21]. What we define as anomaly segmentation will be discussed in detail in the next Section 3.2.

2.1.1 Uncertainty Estimation

Reasoning about uncertainty in neural networks can be traced back to the early 90s and Bayesian neural networks [DL91, Mac92, Mac95], where the model parameters are treated as distributions. Unfortunately, they are not easy to train and, in practice, *dropout* [SHK⁺14] has often been used to approximate Bayesian inference [GG16]. An approach relying on explicitly propagating activation uncertainties through the network was recently proposed [GR18]. However, it has only been studied for a restricted set of distributions, such as the Gaussian one. Another alternative to modeling uncertainty is to replace a single network by an ensemble [LPB17].

For semantic segmentation specifically, the standard approach is to use dropout, as in the *Bayesian SegNet* [BKC17a], a framework later extended in [KG17]. Leveraging such an approach to estimating label uncertainty then becomes an appealing way to detect unknown objects because one would expect these objects to coincide with low confidence regions in the predicted semantic map. This approach was pursued in [IA17a, IA17c, IA17b]. These methods build upon the Bayesian SegNet and incorporate an uncertainty threshold to detect potentially mislabeled regions, including unknown objects.

In [MG18a], the Dirichlet differential entropy is used as a measure of uncertainty. We find that the performance of these statistical methods as obstacle detectors degrades significantly when faced with road surfaces differing from the training set, as the novel textures are treated as anomalies.

However, as shown in our experiments, uncertainty-based methods, such as the Bayesian SegNet [BKC17a] and network ensembles [LPB17], yield many false positives in irrelevant regions.

2.1.2 Prediction Confidence

Most semantic segmentation systems produce a number called a *logit* for each pixel and class. Taking a softmax of the logits produces a per-pixel probability distribution over the classes. The distribution expresses the network’s confidence in its prediction and is a valuable signal for anomaly detection.

The maximum of this distribution can be used as a proxy for confidence and its low value would indicate an anomaly detection [HG17]. The power of maximum-softmax to distinguish in- and out-of-distribution samples is improved in [LLS18] by temperature scaling and adding small anti-adversarial perturbations.

Likewise, the entropy of the softmax distribution is used to find anomalies in semantic segmentation with the right postprocessing [BCR⁺20]. The work of [CRG21b] goes a step further and specifically trains the network to produce high entropy for anomalies.

The MetaSeg [RCH⁺20] technique combines the softmax distribution, the entropy, the shape and size of a semantic segment to find misclassified objects. This has been successfully applied to out-of-distribution detection in [ORF20].

[JLG⁺21] observes that the distribution of max logits strongly differs between each class, making a single threshold ineffective. They propose to standardize the max logits to align the distributions between classes. This is paired with a postprocessing step that removes spurious detections on object edges.

2.1.3 Rejection by Mask-based Segmentation

A very promising approach to anomaly detection has been enabled by mask-based semantic segmentation [CSK21, CMS⁺22, HOLH21] around the time of the conclusion of this thesis. These methods forgo outputting a simple one-hot logit distribution from the network’s final layer. Instead, one branch produces per-pixel embeddings while a transformer branch creates query embeddings. The dot product of pixel and query embeddings yields a collection of masks. These masks are trained to respond to different semantic classes, or object instances in the case of panoptic segmentation.

Since the masks respond only to their chosen class, the anomaly areas are not covered by any mask. Anomalies are found simply as areas where the sum of all masks is very low [NYHG23]. The performance is even better when false-positives at semantic borders are reduced [Mat23]. The rejection approach is particularly impressive since it achieves great anomaly detection performance without need for retraining or modification of the mask-based segmentation system.

2.1.4 Resynthesis and Comparison

If an image is reconstructed so as to preserve the appearance in normal regions and discard anomalous ones, those anomalies can be detected by comparing the input to the reconstruction. This has been achieved in several ways.

Image resynthesis and generation methods, such as autoencoder and GANs, have been used in the past for anomaly detection. The existing methods, however, mostly focus on finding behavioral anomalies in the temporal domain [RNS⁺17, KTP18]. For example, [RNS⁺17] predicts the optical flow in a video, attempts to reconstruct the images from the flow, and treats significant differences from the original images as evidence for an anomaly. This method, however, was only demonstrated in scenes with a static background. Furthermore, as it relies on flow, it does not apply to single images.

To handle individual images, some algorithms compare the image to the output of a model trained to represent the distribution of the original images. For example in [AAAB18], the image is passed through an adversarial autoencoder, and the feature loss between the output and input image is then measured. This can be used to classify whole images but not localize anomalies within the images. Similarly, given a GAN trained to represent an original distribution, the algorithm of [SSW⁺17] searches for the latent vector that yields the image most similar to the input, which is computationally expensive and does not localize anomalies either.

In the context of road scenes, image resynthesis has been employed to detect traffic obstacles. For example, [MC15] relies on the previous frame to predict the non-anomalous appearance of the road in the current one. In [CM15, MVD17], input patches are compared to the output of a shallow autoencoder trained on the road texture, which makes it possible to localize the obstacle. Its limited expressive power is supposed to preserve the smooth road surface while altering obstacles. The approaches described above typically rely on autoencoder for image resynthesis. We have observed that autoencoders tend to learn to perform image compression, simply synthesizing a lower-quality version of the input image, independently of its content. Therefore they do not address textured road surfaces.

As opposed to encoding the input image, the method of [SSW⁺17] trains a generator to capture the training distribution and searches for a latent vector producing an image most similar to the input. However, this method operates on microscopic scans of tissue samples, and, to our knowledge, it has not been applied to data with distributions as diverse as outdoor road scenes.

Our work in Chapter 4 explicitly restricts the intermediate representation of the scene to a dense semantic map, and synthesize a plausible matching image using conditional GANs for image translation [IZZE16, WLZ⁺18]. Since the anomalous regions are not represented by the typical semantic classes, their appearance will be altered by this process. The input and synthesized images are compared using a learned discrepancy module. The following

work of [XZL⁺20] uses a similar pipeline but with a feature distance measure to perform the comparison. The SynBoost [DBBSC21] method fuses resynthesis dissimilarity with semantic segmentation uncertainty estimates to further boost performance.

Rather than encoding the input, one can remove parts of the image and inpaint them based on the surrounding context. In [HGT18] square patches are inpainted and compared with an L_1 metric to detect material defects; the method of [ZKS20] combines the reconstructions obtained with a set of random inpainting masks and uses a multi-scale gradient magnitude similarity metric for comparison. In the context of road scenes, [MC15] proposes to compare the road appearance to similar images memorized from previous video frames; however this would lead to false positives when entering an area with a new road texture. These methods assume high fidelity of the reconstruction and detect every visible difference as an anomaly. In outdoor scenes with road markings and diverse surface textures, the inpainting is bound to be imperfect. We address this in our work in Chapter 5 by training a discrepancy network to focus on the relevant differences. We select image patches and inpaint them with the surrounding road texture, which tends to remove obstacles from those patches. We then use a network trained to recognize discrepancies between the original patch and the inpainted one, which signals an erased obstacle.

Methods specifically designed for removing dynamic objects from traffic scenes have been presented in [BNSC19, BBG⁺19], but they rely on object masks being known *a-priori* without having to detect them first.

2.2 Synthetic Training for Anomaly and Obstacle Detection

There is an intractable variety of semantic anomalies and unexpected objects that can pose a collision threat on roads. To handle this diversity, most existing obstacle detection methods rely on creating synthetic data for training purposes. It is often created from background traffic frames, often from Cityscapes [COR⁺16], into which synthetic obstacles are inserted.

In our work in Chapter 4, we generate synthetic anomalies by altering the semantic class of existing object instances and synthesizing an image from those altered labels. In [DBBSC21], this is complemented by adding the Cityscapes *void* regions as obstacles. However, many of the objects exploited by these techniques are located above or away from the road, and the resulting training data only yields limited performance for small on-road obstacles. Our results show that we outperform these methods.

The method of [BKOS19] introduces an outlier detection head sharing backbone features with the semantic segmentation one. It is trained using extensive out-of-distribution data, injecting outlier patches drawn from ImageNet-1k [DDS⁺09] into the Cityscapes and Vistas [NORK17] scenes. In [VŠA⁺21], synthetic obstacles are obtained by cropping random polygons within the background frame and copying their content onto the road or filling them with a random color.

Chapter 2. Survey of Road Anomaly and Obstacle Detection

We perform synthetic training without data from outside of Cityscapes. In [Chapter 5](#) we extract object instances from the whole dataset and then paste them over the road area. We find that it is important to smooth the edges of the pasted objects to prevent the network from learning to detect the pasting artifacts. In [Chapter 6](#) we refine the arrangement of injected objects so that their size and position are consistent with the scene perspective. This leads to improved detection performance when combined with providing the perspective scale map to the detector network.

Cutting and pasting object instances into urban scenes is also used in [\[GBŠ22\]](#) but in this case they are sources from an external ADE20k [\[ZZP⁺17\]](#) dataset. In this work, this setup is used to learn the likelihood of samples with respect to the known backgrounds as well as to discriminate outliers. In [\[CRG21b\]](#), generalization is achieved by training the segmentation network to maximize the output entropy on explicit out-of-distribution samples. This is done by injecting COCO [\[LMB⁺14\]](#) objects into Cityscapes frames. According to our experiments this is effective for some types of obstacles, presumably ones more similar to COCO objects, but not all of them.

Finally, the synthetic anomalies can be generated from scratch, without relying on existing images. A generative model, such as RealNVP [\[DSDB17\]](#) or a normalizing flow, is trained to create samples at the border of the training distribution [\[GBŠ21a, GBS21b\]](#). Alternatively synthetic anomalies are created through Localized Adversarial Attacks in [\[BBPA21\]](#).

2.3 Obstacle Detection

Our methods rely on a single RGB image for obstacle detection and do not use explicit obstacle training sets. In this section, we review methods that work under those constraints. For a more complete survey of obstacle detection algorithms, we refer the reader to [\[PRG⁺16, RGP⁺17, GJFF⁺18, XMZZ19\]](#).

All the semantic anomaly detection methods discussed so far in this chapter can be used to find unusual obstacles. The obstacles do not belong to the training set and are therefore anomalous. And even if the obstacle were to belong to one of the known classes, for example a tree branch on the road, it could be detected by the standard semantic segmentation system that is nearly always part of the semantic anomaly detection setup. Therefore we evaluate the anomaly detectors on the obstacle track. However limiting the scope to the drivable area can be exploited in the method design.

Road texture is easier to reconstruct than arbitrary scenes. For example patches extracted from the drivable area can be compared to the corresponding output of a shallow autoencoder trained on roads to reveal obstacles [\[CM15, MVD17\]](#). An alternative approach to road appearance reconstruction relies on transforming the previous frame [\[MC15\]](#).

In [Chapter 5](#) we expand the capability of road reconstruction using a sliding window inpainter.

We also introduce a discrepancy network to distinguish real obstacles from inpainting artifacts. Finally, our approach shown in [Chapter 7](#) is capable of segmenting a wide range of small objects in the scene, including road obstacles.

2.4 Using Transformer Attention To Segment Unknown Objects

A moderate number of works use features or attention from transformer backbones for object detection [[GNJ⁺22](#), [SPV⁺21](#)] or semantic segmentation [[AMT22](#), [HZH⁺22](#), [CTM⁺21](#), [WSY⁺22](#)]. In particular, the works of [[AMT22](#), [HZH⁺22](#)] use transformer features to segment objects. However, they do not focus on segmenting unseen objects, but rather on segmenting seen ones under weak supervision [[AMT22](#)] or even no supervision [[HZH⁺22](#)].

Probably closest to our work are the works of [[CTM⁺21](#), [WSY⁺22](#), [SPV⁺21](#)] that are based on transformers learned via self-supervision. They can be considered to some extent as operating in an open world setting as the transformers have been trained on ImageNet [[DDS⁺09](#)] and then applied to other datasets. The work of [[CTM⁺21](#)] performs segmentation based on the final layer class tokens from the transformer attention. The work of [[WSY⁺22](#)] utilizes the transformer from [[CTM⁺21](#)]. It then constructs a graph based on the last layer features, and performs segmentation via a graph cut algorithm. In contrast to our work, the attention itself is not utilized in [[WSY⁺22](#)]. Likewise, [[SPV⁺21](#)] computes the similarity between patches using key-features of the last attention layer, selects a seed patch and segments a group of patches similar to it, yielding a single foreground object.

In our approach called *AttEntropy* described in [Chapter 7](#) we consider the attention between image patches across different layers, study their properties and use an information theoretic approach to segment new objects, potentially in a different domain. More importantly, our work constitutes the first study showing that a semantic segmentation transformer trained in a supervised fashion on known classes inherently learns to segment unknown objects, irrespective of the given context.

3 Road Anomalies and Obstacles: Segment Me If You Can

The advent of high quality, publicly available datasets such as Cityscapes [COR⁺16], BDD100k [YCW⁺20], A2D2 [GKM⁺19], and COCO [LMB⁺14] has hugely contributed to the progress in semantic segmentation. However, while state-of-the-art deep neural networks (DNNs) yield outstanding performance on these datasets, they typically provide predictions for a closed set of semantic classes. Consequently, they are unable to classify an object as *none of the known categories* [ZL17]. Instead, they tend to be overconfident in their predictions, even in the presence of previously-unseen objects [HAB19], which precludes the use of uncertainty to identify the corresponding anomalous regions.

Nevertheless, reliability in the presence of unknown objects is key to the success of applications that have to face the diversity of the real world such as perception in automated driving. This has motivated the creation of benchmarks such as Fishyscapes [BSN⁺19] or CAOS [HBM⁺22]. While these benchmarks have enabled interesting experiments, the limited real-world diversity in Fishyscapes, the lack of a public leader board and of a benchmark suite in CAOS, and the reliance on synthetic images in both benchmarks hinder proper evaluation of and comparisons between the state-of-the-art methods.

Motivated by the limitations of existing anomaly segmentation datasets and by the emerging body of works in this direction [ACS19, BSN⁺19, BCR⁺20, CRG21b, IA17b, JRF20, MWT⁺20, ORF20], we introduce the *Segment Me If You Can*¹ benchmark. It is accompanied with two datasets, consisting of diverse and manually annotated real images, a public leader board and an evaluation suite, providing in-depth analysis and comparisons, to facilitate the development of road anomaly segmentation methods.

Our benchmark encompasses two separate tasks. The first one consists of strict anomaly segmentation, where any previously-unseen object is considered as an anomaly. Furthermore, motivated by the observation that the boundary between known and unknown classes can sometimes be fuzzy, for instance for *car vs. van*, we introduce the task of obstacle segmentation, whose goal is to identify all objects on the road, may they be from known classes or from

¹<https://www.segmentmeifyoucan.com/>

Chapter 3. Road Anomalies and Obstacles: Segment Me If You Can

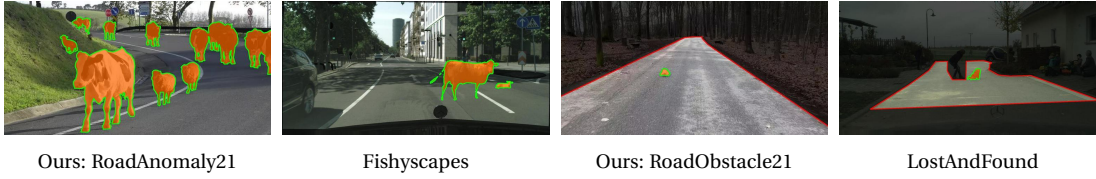


Figure 3.1: Comparison of images from our and existing public datasets. Anomalies / obstacles are highlighted in orange, darkened regions are excluded from the evaluation. In RoadAnomaly21, anomalies may appear everywhere in the image. In contrast to Fishyscapes, where anomalous objects are synthetic, all RoadAnomaly21 images are real. In RoadObstacle21, the region of interest is restricted to the drivable area with obstacles ahead. This is comparable to LostAndFound, where the labeling, however, is not always consistent, e.g. children are anomalies but other humans not.

unknown ones.

For the anomaly track, we provide a dataset of 100 images with pixel-wise annotations over two classes (anomaly, not anomaly) and a void class, which, in analogy to Cityscapes, signals the pixels that are excluded from the evaluation. We consider any object that strictly cannot be seen in the Cityscapes data as anomalous, appearing anywhere in the image. For the obstacle track, our dataset contains 327 images with analogous annotation (obstacle, not obstacle, void), and focuses only on the road as region of interest. The focus in this track is of more practical need such as for automated driving systems, targeting obstacles that may cause hazardous street situations, see Figure 3.1. All images of our datasets are publicly available for download², together with a benchmark suite that computes both established pixel-wise metrics and recent component-wise ones.

In the remainder of this chapter, we first review existing anomaly detection datasets and evaluation metrics in more detail. We then describe the datasets we have gathered over time and the benchmark which is the crowning achievement of this effort. The evaluation results are presented later in Chapter 8.

3.1 Datasets and Benchmarks

In this section we first review previous datasets for anomaly detection, with some of them being designed for road anomaly segmentation.

Existing methods for anomaly detection have often been evaluated on their ability to separate images from two different source distributions such as separating MNIST from FashionMNIST [CJA18, MH20, vASTG20], NotMNIST [vASTG20], or Omniglot [LST15], and separating CIFAR-10 from SVHN [LLLS18b, MH20, vASTG20] or LSUN [LLLS18b, LLS18, MH20]. Such experiments can be found in many works, including [HG17, CJA18, LLLS18b, LLS18, vASTG20, MH20].

For semantic segmentation, a similar task was therefore proposed by the WildDash benchmark [ZHM⁺18] that analyzes semantic segmentation methods trained for driving scenes on

a range of failure sources including full-image anomalies such as images from the beach. In our work, by contrast, we focus on the problem of robustness to anomalies that only cover a small portion of the image, and on the methods that aim to segment such anomalies, that is methods for the task of *anomaly segmentation*.

One prominent dataset tackling the task of anomaly segmentation is LostAndFound [PRG⁺16], which shares the same setup as Cityscapes [COR⁺16] but includes anomalous objects / obstacles in various street scenes in Germany. LostAndFound contains 9 different object types as anomalies, and only has annotations for the anomaly and the road surface. Furthermore, it considers children and bicycles as anomalies, even though they are part of the Cityscapes training set, and it contains several labeling mistakes. Although we filter and refine LostAndFound in this work², similar to Fishyscapes [BSN⁺19], the low diversity of anomalies persists.

The CAOS BDD-Anomaly benchmark [HBM⁺22] suffers from a similar low-diversity issue, arising from its use of only 3 object classes sourced from the BDD100k dataset [YCW⁺20] as anomalies (besides including several labeling mistakes, see Section 3.4.4). Both Fishyscapes and CAOS try to mitigate this low diversity by complementing their real images with synthetic data. Such synthetic data, however, is not realistic and not representative of the situations that can arise in the real world.

In general, the above works illustrate the shortage of diverse real-world data for anomaly segmentation. Additional efforts in this regard have been made by leveraging multiple sensors such as the LiDAR guided Small Obstacle dataset [SKGMK20]. In any event, most of the above datasets are fully published with annotations, allowing methods to overfit to the available anomalies. Furthermore, apart from Fishyscapes, we did not find any public leader boards that allow for a trustworthy comparison of new methods. To provide a more reliable test setup, we do not share the labels and request predictions of the shared images to be submitted to our servers. Furthermore, we provide a leader board, which we publish alongside two novel real-world datasets, namely RoadAnomaly21 and RoadObstacle21. A summary of the main properties of the mentioned datasets is given in Table 3.1. Our main contribution in both proposed datasets is the diversity of the anomaly categories and of the scenes.

In RoadAnomaly21, anomalies can appear anywhere in the image, which is comparable to Fishyscapes LostAndFound [BSN⁺19] and CAOS BDD-Anomaly [HBM⁺22]. Although the latter two datasets are larger, their images only show a limited diversity of anomaly types and scenes because they are usually frames of videos captured in single scenes. By contrast, in our dataset every image shows a unique scene, with at least one out of 26 different types of anomalous objects and each sample widely differs in size, ranging from 0.5% to 40% of the image.

In RoadObstacle21, all anomalies (or obstacles) appear on the road, making this dataset comparable to LostAndFound [PRG⁺16] and the LiDAR guided Small Obstacle dataset [SKGMK20].

²In the following, we refer to the LostAndFound subset without the images of children, bicycles and invalid annotations as *LostAndFound-NoKnown*.

Chapter 3. Road Anomalies and Obstacles: Segment Me If You Can

Dataset	anomaly pixels	non-anomaly pixels	diverse scenes	different anomalies	dataset size	ground truth (gt) components	mean & std of gt size relative to image size
Fishyscapes LostAndFound val [BSN ⁺ 19]	0.23%	81.13%	12	7	373	165	0.13% \pm 0.23%
CAOS BDD-Anomaly test [HBM ⁺ 22]	0.83%	81.28%	810	3	810	1231	0.55% \pm 1.84%
Ours: RoadAnomaly21 test	13.83%	82.17%	100	26	100	262	4.12% \pm 7.29%
LostAndFound test (NoKnown) [PRG ⁺ 16]	0.12%	15.31%	13 (12)	9 (7)	1203 (1043)	1864 (1709)	0.08% \pm 0.16%
LiDAR guided Small Obstacle test [SGMK20]	0.07%	36.09%	2	6	491	1203	0.03% \pm 0.07%
Ours: RoadObstacle21 test	0.12%	39.08%	8	31	327	388	0.10% \pm 0.25%

Dataset (as above)	labels are private	weather conditions	geography
Fishyscapes val	(✓ in test set)	clear	DE
CAOS BDD test	✗	clear, snow, night, rain	US
Ours: RA21 test	✓	clear, snow	global
LaF test	✗	clear	DE
Small Obs. test	✗	clear	IN
Ours: RO21 test	✓	clear, snow, night	CH, DE

Table 3.1: Main properties of comparable real-world anomaly (top three rows) and obstacle (bottom three rows) segmentation datasets. Our main contribution is the diversity of the anomaly (or obstacle) categories and of the scenes. Note that *void* pixels are not included in this table.

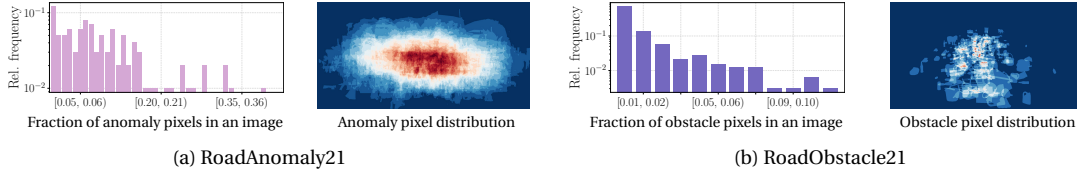


Figure 3.2: Relative frequency of annotated anomaly / obstacle pixels within an image over the 100 images in the RoadAnomaly21 test dataset (left) and the 327 images in the RoadObstacle21 test dataset (right), respectively. Here, the fraction of anomaly / obstacle pixels serves as a proxy for the size of the objects of interest within an image. Note that the y-axes of the histograms are log-scaled.

Again, the latter two datasets contain more images than ours; however, the high numbers of images result from densely sampling frames from videos. Consequently, those two datasets lack in object diversity (9 and 6 categories, respectively, compared to 31 in our dataset). Furthermore, the videos are recorded under perfect weather conditions while RoadObstacle21 shows scenes in diverse situations including night, dirty roads, and snowy conditions.

3.1.1 Our earlier datasets

The benchmark is a culmination of our larger effort of defining the anomaly and obstacle task and providing data to study them. We collected and labeled the first batch of 60 anomaly images as part of the resynthesis work described in Chapter 4. Later on we captured our own images of obstacles placed on the road for the road inpainting project shown in Chapter 5. Following those efforts we entered into a collaboration spanning EPFL, University of Wuppertal, and ETH Zurich, during which extra images were captured and labeled resulting in the final form of the benchmark.

Road Anomaly 19

Motivated by the scarcity of available data for unexpected object detection, we collected online images depicting anomalous objects such as animals, rocks, lost tires, trash cans, and construction equipment located on or near the road. We then produced per-pixel annotations of these unexpected objects manually using the *Grab Cut* algorithm [RKB04] to speed up the process. The dataset contains 60 images rescaled to a uniform size of 1280×720 . We make this dataset² and the labeling tool³ publicly available. The dataset was originally released with our work in Chapter 4.

Road Obstacle 20.

The *Lost & Found* benchmark features urban environments similar to those in the *Cityscapes* training data. To evaluate our obstacle detector on a wider variety of road surfaces and objects, we collected our own *Road Obstacles 20* dataset which is depicted by Figure 3.3. It features seven different scenes and comprises a total of 160 labeled 1920×1080 frames. The labels include pixel masks for individual obstacle instances along with approximate outlines of the drivable area. We take the evaluation region of interest (ROI) to be the area within these outlines. If an object has a hole, such as a basket with a handle, we label the hole as outside of the ROI and ignore it in our evaluations as we do for the background. The dataset was originally part of our work in Chapter 5.

The dataset is divided into the *daylight* subset of six scenes (105 images) with good daylight visibility and the *snowfall* sequence of 55 frames taken in difficult weather conditions. The *snowfall* track poses additional challenges for the obstacle detector: snowflakes visible in the air can trigger false positives, and a water drop on the camera lens can obscure parts of an image. We report the results for this track separately, as such conditions are far outside the training domains of any of the evaluated methods. We report the union of both tracks as *Road Obstacles - all*.

3.2 Benchmark Description

The aim of our benchmark is two-fold. On one hand, by providing diverse data with consistent annotations, we seek to facilitate progress in general semantic anomaly segmentation research. On the other hand, by focusing on road scenes, we expect our benchmark to accelerate the progress towards much needed segmentation/obstacle-detection methods for safe automated driving.

To achieve these goals, our benchmark covers two tasks. First, it tackles the general problem of anomaly segmentation, aiming to identify the image regions containing object classes that

² *Road Anomaly* dataset: cvlab.epfl.ch/data/road-anomaly/

³ Our labeling tool: github.com/cvlab-epfl/LabelGrab

Chapter 3. Road Anomalies and Obstacles: Segment Me If You Can

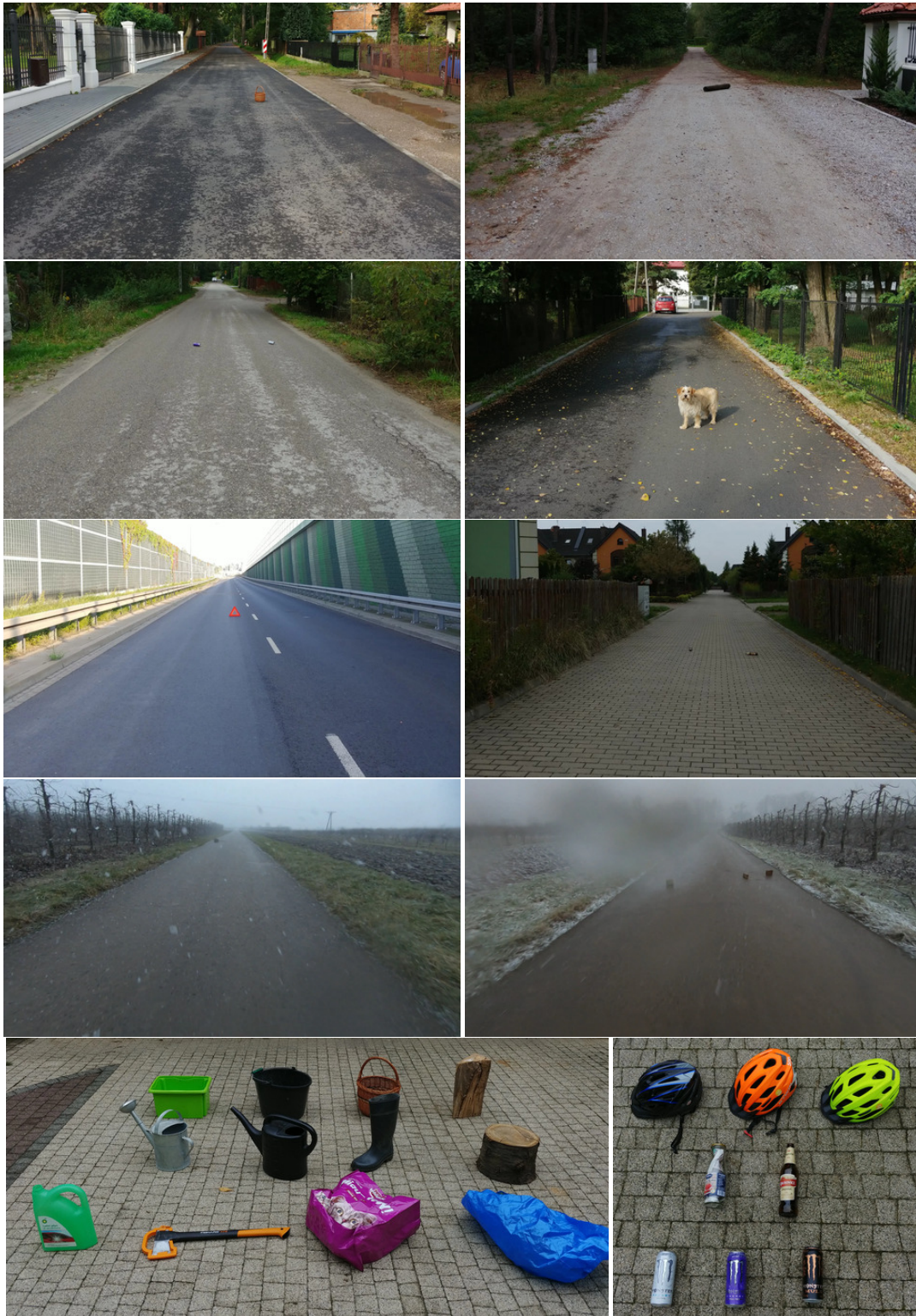


Figure 3.3: **Road Obstacles 20 dataset.** Top: Example frames representing each of the 7 scenes. Bottom: Some of the objects featured in the dataset.

have never been seen during training and thus for which semantic segmentation cannot be correct. This is necessary for any reliable decision-making process, and it is of great importance to many computer-vision applications. Note that, in accordance to [BSN⁺19, HBM⁺22], we define anomaly as objects that do not fit any of the class definitions in the training data. In some works, anomaly may be used to describe visually different inputs like a car in a novel color which does not fit our definition.

This strict definition of semantic anomalies, however, can sometimes be ill-defined because (1) existing semantic segmentation datasets such as Cityscapes [COR⁺16] often contain ambiguous and ignored regions (annotated as *void*) which are not strictly anomalies since they are seen during training; (2) the boundary of some classes is fuzzy (for example distinguishing cars versus vans versus rickshaws) making it unclear whether some regions should be considered as anomalous or not. To address these issues, and to account for the fact that automated driving systems need to make sure that the road ahead is free of *any* hazardous objects, we further incorporate obstacle segmentation as a second task in our benchmark, whose goal is to identify any non-drivable region on the road, may the non-drivable region correspond to a known object class or an unknown one.

3.3 Tracks and Datasets

We now present the two tracks in our benchmark corresponding to the two tasks discussed above. Each track contains its own dataset with different properties and is therefore evaluated separately in our benchmark suite. An overview comparing our datasets to related public ones is given in Table 3.1.

3.3.1 RoadAnomaly21

The road-anomaly track benchmarks general anomaly segmentation in full-street scenes. It consists of an evaluation dataset of 100 images with pixel-level annotations. The data is an extension of the one introduced in Road Anomaly 19 (Section 3.1.1), now including a broader collection of images and finer-grain labeling. In particular, we removed low-quality images and ones lacking clear road scenes. We removed labeling mistakes, added the *void* class, and included 68 newly collected images. Each image contains at least one anomalous object such as an animal or an unknown vehicle. The anomalies can appear anywhere in the image which were collected from web resources and therefore depict a wide variety of environments. The distribution of object sizes and location is shown in Figure 3.2(a). Moreover, we provide 10 additional images with annotations such that users can check the compatibility of their methods with our benchmark implementation.

3.3.2 RoadObstacle21

The road-obstacle track focuses on safety for automated driving. The objects to segment in the evaluation data always appear on the road ahead; for example they represent realistic and hazardous obstacles that are critical to detect. Our dataset consists of 222 new images taken by ourselves and 105 from Road Obstacle 20 (Section 3.1.1), summing up to a total of 327 evaluation images with pixel-level annotations. The region of interest in these images is given by the road, which is assumed to belong to the known classes on which the algorithm was trained. The obstacles in this dataset are chosen such that they all can be understood as anomalous objects as well such as stuffed toys, sleighs, or tree stumps. They appear at different distances and are surrounded by road pixels. This allows us to focus our evaluation on the obstacles as other objects lie outside the region of interest. The distribution of object sizes and location is shown in Figure 3.2(b). Moreover, this dataset incorporates different road surfaces, lighting and weather conditions, thus encompassing a broad diversity of scenes. An extra track of additional 85 images with scenes at night and in extreme weather (such as snowstorms) is also available. However, the latter subset is excluded from our numerical experiments due to the significant domain shift. Lastly, we provide 30 additional images with annotations such that users can check the compatibility of their methods with our benchmark implementation.

3.3.3 Labeling Policy

In both datasets, the pixel-level annotations include three classes:

- 1) anomaly / obstacle
- 2) not anomaly / not obstacle
- 3) void.

In RoadAnomaly21, the 19 Cityscapes evaluation classes [COR⁺16], on which most semantic segmentation DNNs are trained, serve as basis to judge whether an object is considered anomalous or not. Everything that fits in the class definitions of Cityscapes is thus labeled as *not anomaly*. This track focuses on the detection of objects which are semantically different from those in the Cityscapes training data. Therefore, if image regions cannot be clearly assigned to any of the Cityscapes classes, they are labeled as *anomaly*. The objects, which are not the main anomalies of interest in the context of street scenes, are labeled as *void* and excluded from our evaluation. The latter class include, for instance, mountains or water in the image background and street lights. In ambiguous cases, which can arise from a strong domain shift to Cityscapes, we assign the void class as well to properly evaluate semantic anomaly segmentation.

In RoadObstacle21, the task is defined as distinguishing between drivable area and non-drivable area. The goal is to make sure that the road ahead of the ego-car is free of any hazard, irrespective of the object category of potential obstacles. Therefore, the drivable area is labeled

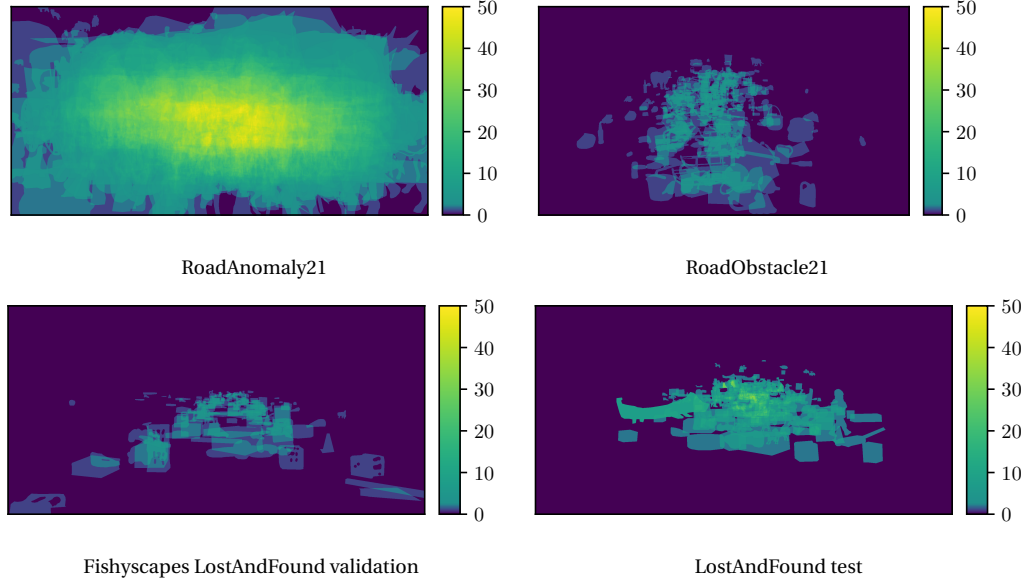


Figure 3.4: Comparison of the (spatial) pixel distributions between RoadAnomaly21 and Fishyscapes LostAndFound (100 images each) as well as RoadObstacle21 and a subset of randomly sampled images from the LostAndFound test dataset (327 images each). The color indicates the frequency of observing an anomaly in each pixel location, averaged over the images in the dataset.

as *not obstacle*. This class particularly also includes regions on the road which visually differ from the rest of the road. Moreover, every object visually enclosed in the drivable area is labeled as *obstacle*. All image regions outside the road are assigned to the *void* class and ignored in the evaluation. As a quality assessment for both tracks, each labeled image was reviewed by at least three people in order to guarantee the highest quality of labels.

3.3.4 Validation Dataset

In order to ensure that methods run as intended with our benchmark code, we provide small validation sets (including ground truth annotations) for the anomaly track, called *RoadAnomaly21 validation*, and for the obstacle track, called *RoadObstacle21 validation*.

These datasets show similar scenes and objects as in RoadAnomaly21 test and RoadObstacle21 test. The subsets contain 10 images with 16 ground truth components and 30 images with 45 ground truth objects in total, respectively. Note that although both datasets share the same setup as in the corresponding test splits, they are still **not** representative for the test data since they contain only a very limited number of different road surfaces and diverse obstacle types. Therefore we do not recommend to fine-tune methods on these two validation datasets.



Figure 3.5: Four example images of densely sampled frames from a video sequence (of 18 frames in total) with ground truth annotation in the LostAndFound test set. Due to this sampling, LostAndFound achieve their high number of images but, as shown in this figure, several images are nearly identical.

3.4 Comparison to other datasets

3.4.1 Fishyscapes LostAndFound

The Fishyscapes LostAndFound validation dataset [BSN⁺19] consists of 100 images from the original LostAndFound data [PRG⁺16] with refined labels. With this labeling, anomalous objects are not restricted to only appear on the road but everywhere in the image, therefore Fishyscapes LostAndFound fits our benchmark’s anomaly track.

Comparing the RoadAnomaly21 and Fishyscapes LostAndFound datasets in terms of anomaly class frequency per pixel location, as observed in Figure 3.4, one notices a clear difference in the variation of object locations and sizes. While in Fishyscapes LostAndFound the objects appear mostly in the center of the image and are also rather small, the objects in RoadAnomaly21 may appear everywhere in the image and have sizes ranging from 122 up to 883,319 pixels (thus covering up to more than one third of the image). The low variety in object sizes is also noticeable in the pixel-wise class distribution; in particular, 13.8% of the pixels belong to the anomaly class and 82.2% to non-anomaly in RoadAnomaly21 whereas in Fishyscapes LostAndFound only 0.23% belong to anomaly and 81.13% to non-anomaly.

3.4.2 LostAndFound test-NoKnown

The LostAndFound dataset [PRG⁺16] shares the same setup as Cityscapes but includes small obstacles on the road. Therefore, this dataset fits our benchmark’s obstacle track. When a model is trained on Cityscapes, the LostAndFound dataset then contains images with objects that have been previously seen and therefore are not anomalies. As most of our methods are designed for anomaly detection, we filtered out all scenes in the LostAndFound test split where the obstacles belong to known classes, such as children or bicycles, and call this subset LostAndFound test-NoKnown. In this way, the results obtained with our evaluated methods on LostAndFound test-NoKnown and on our RoadObstacle21 dataset are comparable.

Both datasets have obstacles in the same size range. Both RoadObstacle21 and LostAndFound test-NoKnown have 0.12% of the pixels labeled as obstacles, while 39.08% and 15.31% of the

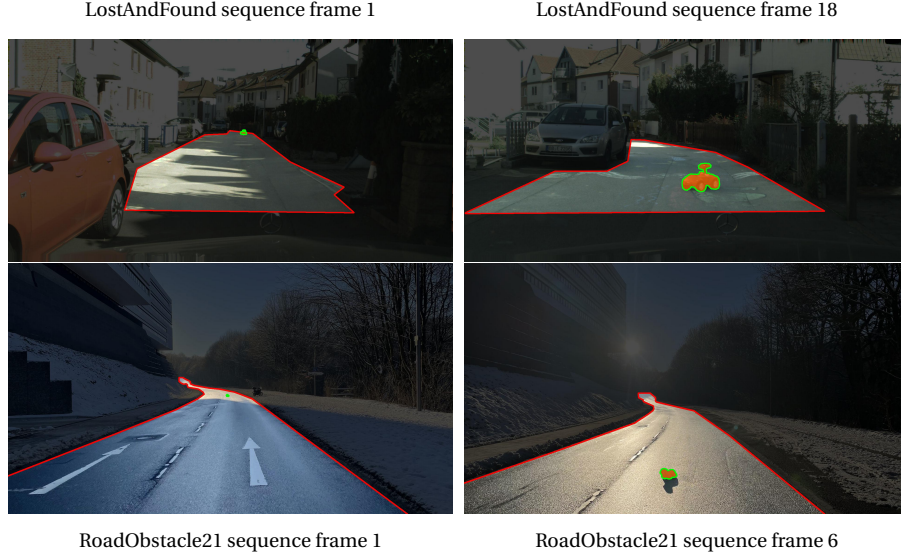


Figure 3.6: Comparison of one sequence from the LostAndFound test set (top row) and one sequence from the RoadObstacle21 test set (bottom row). In this figure, the first and last frame of a video sequence which are included in the respective test set are shown. We observe that in this LostAndFound example 18 images of one sequence are included in the test while in RoadObstacle21 at most 6 frames are included (which differ significantly in lighting in this example).

pixels belong to not obstacles, respectively.

Regarding the dataset size, LostAndFound achieves their high number of images by densely sampling from video sequences. Consequently, some images depict nearly identical scenes (same environment and obstacle combination with the obstacle approximately at the same distance); see [Figure 3.5](#). In RoadObstacle21 the number of different environment and obstacle combinations is considerably higher due to the wide variety of 31 object types in the dataset. If multiple images depict the same scene, we made sure that the distance to the obstacle (and therefore the size of the obstacle in the image) varies noticeably from image to image; see [Figure 3.6](#).

3.4.3 LiDAR Guided Small Obstacle Dataset

The third publicly available dataset to which we applied our benchmark suite is the LiDAR guided Small obstacle Segmentation dataset [[SKGMK20](#)], which can be viewed as a reference dataset for our obstacle track. As this rather focuses on the challenge of detecting obstacles via multiple sensors including LiDAR, the camera images of this dataset are purposely challenging, for example due to low illumination, blurry images and barely visible obstacles. [Figure 3.7](#) shows an example of this dataset which highlights the difficulty of anomaly detection.

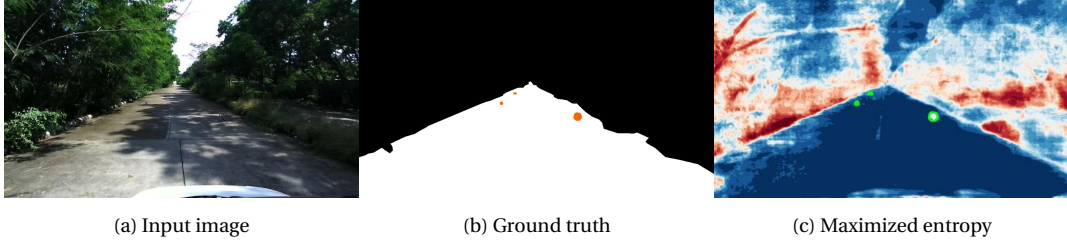


Figure 3.7: An example image (a) from the Small Obstacle dataset with the corresponding ground truth annotation (b) and an obstacle score heatmap obtained with maximized entropy (c). Here the obstacles are barely visible in the input image due to their size and the scene’s illumination; that is why camera-only based segmentation techniques tend to fail for this dataset.

3.4.4 CAOS BDD-Anomaly

The CAOS BDD-Anomaly dataset [HBM⁺22] consists of images sourced from BDD100k [YCW⁺20]. In order to create an anomaly segmentation dataset, the authors split the BDD100k data such that images with motorcycles, bicycles, and trains are separated from the rest. These left out objects are then considered as anomalies. We do not perform any experiments on CAOS BDD-Anomaly since the considered anomalous objects are not strictly unknown. They also appear in Cityscapes [COR⁺16] on which most semantic segmentation models are trained.

3.5 Performance Metrics

For the sake of brevity, in what follows we refer to both anomalies and obstacles as *anomalies*.

Pixel level

Let \mathcal{Z} denote the set of image pixel locations. A model with a binary classifier providing anomaly scores $s(x) \in \mathbb{R}^{|\mathcal{Z}|}$ for an image $x \in \mathcal{X}$ (from a dataset $\mathcal{X} \subseteq [0, 1]^{N \times |\mathcal{Z}| \times 3}$ of N images) discriminates between the two classes *anomaly* and *non-anomaly*. We evaluate the separability of the pixel-wise anomaly scores via the area under the precision-recall curve (AuPRC). Let $\mathcal{Y} \subseteq \{\text{“anomaly”}, \text{“not anomaly”}\}^{N \times |\mathcal{Z}|}$ be the set of ground truth labels per pixel for \mathcal{X} . Analogously, we denote the predicted labels with $\hat{\mathcal{Y}}(\delta)$, obtained by pixel-wise thresholding on $s(x) \forall x \in \mathcal{X}$ w.r.t. some threshold value $\delta \in \mathbb{R}$. Then, for the anomaly class ($c_1 = \text{“anomaly”}$) we compute

$$\text{precision} = \frac{|\mathcal{Y}_{c_1} \cap \hat{\mathcal{Y}}_{c_1}(\delta)|}{|\hat{\mathcal{Y}}_{c_1}(\delta)|}, \quad \text{recall} = \frac{|\mathcal{Y}_{c_1} \cap \hat{\mathcal{Y}}_{c_1}(\delta)|}{|\mathcal{Y}_{c_1}|} \quad (3.1)$$

with \mathcal{Y}_{c_1} and $\hat{\mathcal{Y}}_{c_1}$ representing the ground truth labels and predicted labels, respectively. For the AuPRC, The AuPRC approximates $\int \text{precision}(\delta) d\text{recall}(\delta)$ and is threshold independent [BEP13]. It also puts emphasis on detecting the minority class, making it particularly well suited as our main pixel-wise evaluation metric since the pixel-wise class distributions of RoadAnomaly21 and RoadObstacle21 are considerably unbalanced, see Table 3.1.

To consider the safety point of view, we also include the false positive rate at 95% true positive rate (FPR_{95}) in our evaluation, where the true positive rate (TPR) is equal to the recall of the anomaly class. The false positive rate (FPR) is the number of pixels falsely predicted as anomaly over the number of all non-anomaly pixels. Hence, for the anomaly class we compute

$$\text{FPR}_{95} = \frac{|\hat{\mathcal{Y}}_{c_1}(\delta') \cap \mathcal{Y}_{c_2}|}{|\mathcal{Y}_{c_2}|} \quad \text{s.t. } \text{TPR}(\delta') = 0.95, \quad (3.2)$$

where $c_2 = \text{"not anomaly"}$. The FPR_{95} metric indicates how many false positive predictions must be made to reach the desired true positive rate. Note that any prediction which is contained in a ground-truth labeled region of the class void is not counted as false positive; see [Section 3.3](#). In particular for the RoadObstacle21 dataset, the evaluation is therefore restricted to the road area.

Component level

From a practitioner’s perspective, it is very important to detect all anomalous regions in the scene regardless of the number of pixels they cover. However, pixel-level metrics may neglect small anomalies. While one could thus focus on object detection metrics, the notion of individual objects is in fact not relevant for anomaly (region) detection. To satisfy these requirements, we therefore consider performance metrics acting at the component level.

The main metrics for component-wise evaluation are the numbers of *true-positives* (TP), *false-negatives* (FN), and *false-positives* (FP). Considering anomalies as the positive class, we use a component-wise localization and classification quality measure to define the TP, FN, and FP components. Specifically, we define this measure as an adjusted version of the component-wise intersection over union (sIoU), introduced in [\[RCH⁺20\]](#). In particular, while in [\[RCH⁺20\]](#) the sIoU is computed for predicted components, we consider the sIoU for ground-truth components to compute TP and FN. To compute FP, we employ the positive predictive value (PPV, or component-wise precision) for predicted components as quality measure. We discuss the definitions of these quantities in more detail below.

Let \mathcal{Z}_c be the set of pixel locations labeled with class $c = \text{"anomaly"}$ in the dataset \mathcal{X} . We consider a connected component of pixels (where the 8 pixels surrounding pixel z in image $x \in \mathcal{X}$ are taken to be its neighbors) that share the same class label as a *component*. Then, let us denote by $\mathcal{K} \subseteq \mathcal{P}(\mathcal{Z}_c)$, with $\mathcal{P}(\mathcal{S})$ the power set of a set \mathcal{S} , the set of anomaly components according to the ground truth, and by $\hat{\mathcal{K}} \subseteq \mathcal{P}(\mathcal{Z}_c)$ the set of components predicted to be anomalous by some machine-learning model.

Formally, the sIoU is a mapping $\text{sIoU} : \mathcal{K} \rightarrow [0, 1]$. For $k \in \mathcal{K}$, it is defined as

$$\text{sIoU}(k) := \frac{|k \cap \hat{K}(k)|}{|(k \cup \hat{K}(k)) \setminus \mathcal{A}(k)|} \quad \text{with} \quad \hat{K}(k) = \bigcup_{\hat{k} \in \hat{\mathcal{K}}, \hat{k} \cap k \neq \emptyset} \hat{k} \quad (3.3)$$

Chapter 3. Road Anomalies and Obstacles: Segment Me If You Can

Target 1 $k \in \mathcal{K}(x)$	Prediction $\hat{k} \in \hat{\mathcal{K}}(x)$	$\text{IoU}(k) = 0.50$ $\text{sIoU}(k) = 0.50$	Target 1 $k \in \mathcal{K}(x)$	Target 2 & adjustment $\mathcal{A}(k)$	$\text{IoU}(k) = 0.50$ $\text{sIoU}(k) = 0.99$
------------------------------------	--	---	------------------------------------	---	---

Figure 3.8: Illustration of the ordinary component-wise intersection over union (IoU) and the adjusted one (sIoU). In both examples above, the prediction \hat{k} (blue rectangle) is the same but covers different targets (green rectangles). On the left, both IoU and sIoU yield the same score. On the right, IoU punishes the prediction as it does not cover each object precisely. By contrast, sIoU checks how much the predictions cover the ground-truth regions, independently of whether prediction/ground truth belongs to a single or multiple objects. In automated driving, it is more important to detect all anomalous regions (whether they belong to single or multiple objects), rather than to detect each object precisely. Since two targets are separated by at least one pixel, $\text{IoU} = \text{sIoU} = 1$ if and only if the prediction covers one target perfectly.

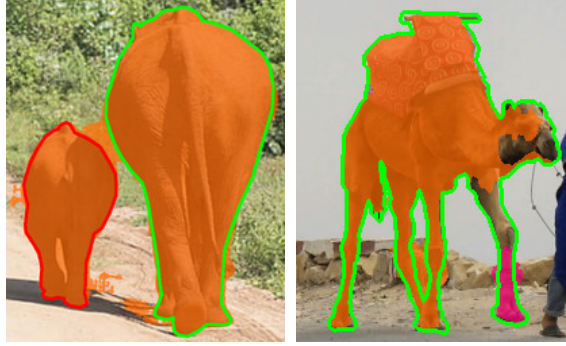


Figure 3.9: Two examples underlining the difference between IoU and adjusted IoU (sIoU). The ground-truth components are indicated by green/red contours, and predicted components are highlighted by other colors. *Left:* Two ground-truth components (green & red) intersect with one predicted component (orange). Green: **IoU 68.18% vs. sIoU 87.01%**; red: **IoU 21.68% vs. sIoU 68.44%**. *Right:* Two predicted components (orange & pink) intersect with one ground-truth component (green). Orange: **IoU 78.97% vs. sIoU 81.69%**; pink: **IoU 03.44% vs. sIoU 18.91%**.

and $\mathcal{A}(k) = \{z \in k' : k' \in \mathcal{K} \setminus \{k\}\}$. With the adjustment $\mathcal{A}(k)$, the pixels are excluded from the union if and only if they correctly intersect with another ground-truth component $k' \in \mathcal{K}(x)$, which is not equal to k . This may happen when one predicted component covers multiple ground-truth components, as illustrated in Figure 3.8. Given some threshold $\tau \in [0, 1]$, we then call a target $k \in \mathcal{K}$ TP if $\text{sIoU}(k) > \tau$, and FN otherwise. We refer to Section 3.5 for qualitative examples of the difference between IoU and sIoU.

For the other error type, such as FP, we compute the PPV (or precision) for $\hat{k} \in \hat{\mathcal{K}}$, which is defined as

$$\text{PPV}(\hat{k}) := \frac{|\hat{k} \cap K(\hat{k})|}{|\hat{k}|}, \quad (3.4)$$

We then call a predicted component $\hat{k} \in \hat{\mathcal{K}}$ FP if $\text{PPV}(\hat{k}) \leq \tau$.

As an overall metric, we additionally include the component-wise F_1 -score defined as

$$F_1(\tau) := \frac{2 \cdot \text{TP}(\tau)}{2 \cdot \text{TP}(\tau) + \text{FN}(\tau) + \text{FP}(\tau)} \in [0, 1], \quad (3.5)$$

which summarizes the TP, FN and FP quantities (that depend on τ). The component-level metrics allow one to evaluate localization of objects irrespective of their size and hence big objects will not dominate these metrics. In addition, while object detection metrics punish predictions that cover multiple ground-truth objects or vice-versa, our component-level metric does not do so; see [Figure 3.8](#).

Qualitative examples revealing the difference of IoU and sIoU

If we consider component-level metrics over ground-truth components, it may happen that several components are close together and therefore covered by one predicted component. Although the real error can be small, the IoU punishes both ground-truth components. The same holds the other way around when considering metrics over predicted components such as when one ground-truth component is covered by several predicted components. A qualitative example is given in [Figure 3.9](#). A small number of incorrectly predicted pixels may cause a strong decrease in the IoU. The adjusted IoU (sIoU) is less sensitive in such cases. sIoU focuses on correctly covering the regions of obstacles/anomalies in the image rather than finding such regions separately for each instance, as done by IoU. In self-driving it is more important to know the regions of anomaly rather than how many of them exist.

3.6 Conclusion

We have introduced a unified and publicly available benchmark suite that evaluates a method's performance for anomaly segmentation with established pixel-level as well as recent component-level metrics. Our benchmark suite is applicable in a plug-and-play fashion to any dataset for anomaly segmentation that comes with ground truth, such as LostAndFound and Fishyscapes LostAndFound, allowing for a better comparison of new methods. Moreover, our benchmark is accompanied with two publicly available datasets, RoadAnomaly21 for anomaly segmentation and RoadObstacle21 for obstacle segmentation.

These two datasets challenge two important abilities of computer vision systems: On one hand their ability to detect and localize unknown objects; on the other hand their ability to reliably detect and localize obstacles on the road, may they be known or unknown. Our datasets consist of real images with pixel-level annotations and depict street scenes with higher variability in object types and object sizes than existing datasets.

The images of the datasets and the software are available at <https://www.segmentmeifyoucan/>.

4 Detecting the Unexpected via Image Resynthesis



Figure 4.1: **Detecting the unexpected.** While uncertainty- and autoencoder-based methods tend to be distracted by the background, our approach focuses much more accurately on the unknown objects.

When we first approached the problem of semantic anomaly detection, the existing solutions followed two trends. The first one involves reasoning about the prediction uncertainty of the deep networks used to perform the segmentation [BKC17a, LPB17, KG17, GR18]. In the driving scenario, we have observed that the uncertain regions tend not to coincide with unknown objects, and, as illustrated by Figure 4.1, these methods therefore fail to detect the unexpected. The second trend consists of leveraging autoencoders to detect anomalies [CM15, MVD17, AAAB18], assuming that never-seen-before objects will be decoded poorly. We found, however, that autoencoders tend to learn to simply generate a lower-quality version of the input image. As such, as shown in Figure 4.1, they also fail to find the unexpected objects.

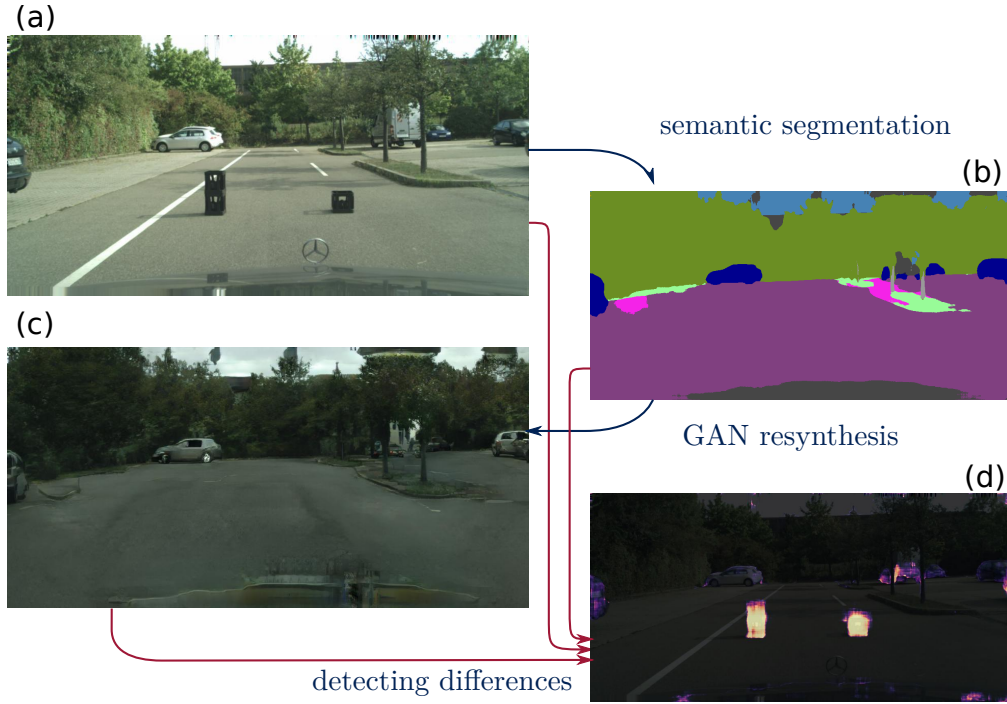


Figure 4.2: **Our Approach.** (a) Input image from the *Lost and Found* [PRG⁺16] dataset containing objects of a class the segmentation algorithm has not been trained for. (b) In the resulting semantic segmentation, these objects are lost. (c) In the image resynthesized based on the segmentation labels, they are also lost. (d) Using a specially trained *discrepancy network* to compare the original image and the resynthesized one highlights the unexpected objects.

We therefore introduced a radically different approach to detecting the unexpected. Figure 4.2 depicts our pipeline, built on the following intuition: In regions containing unknown classes, the segmentation network will make spurious predictions. Therefore, if one tries to resynthesize the input image from the semantic label map, the resynthesized unknown regions will look significantly different from the original ones. In other words, we reformulate the problem of segmenting unknown classes as one of identifying the differences between the original input image and the one resynthesized from the predicted semantic map. To this end, we leverage a generative network [WLZ⁺18] to learn a mapping from semantic maps back to images. We then introduce a discrepancy network that, given as input the original image, the resynthesized one, and the predicted semantic map produces a binary mask indicating unexpected objects. To train this network *without* ever observing unexpected objects, we simulate such objects by changing the semantic label of known object instances to other, randomly chosen classes. This process, described in Section 4.1.2, does *not* require seeing the unknown classes during training which makes our approach applicable to detecting never-seen-before classes at test time.

Our contribution is therefore a radically new approach to identifying regions that have been misclassified by a given semantic segmentation method based on comparing the original image with a resynthesized one. We demonstrate the ability of our approach to detect unex-

pected objects using the Lost and Found dataset [PRG⁺16]. This dataset, however, only depicts a limited set of unexpected objects in a fairly constrained scenario. To palliate this lack of data, we create a new dataset depicting unexpected objects such as animals, rocks, lost tires, and construction equipment on roads. For the description of the dataset, please see Section 3.1.1.

Our method outperforms uncertainty-based baselines, as well as the state-of-the-art autoencoder-based method specifically designed to detect road obstacles [CM15]. Furthermore, our approach to detecting anomalies by comparing the original image with a resynthesized one is generic and applies to other tasks than unexpected object detection. For example, deep learning segmentation algorithms are vulnerable to adversarial attacks [XWZ⁺17, CANK17], that is, maliciously crafted images that look normal to a human but cause the segmentation algorithm to fail catastrophically. As in the unexpected object detection case, re-synthesizing the image using the erroneous labels results in a synthetic image that looks nothing like the original one. Then, a simple non-differentiable detector, thus less prone to attacks, is sufficient to identify the attack. Since it is not fully relevant to the road anomaly problem, the experiment regarding adversarial attacks is available in the conference version of this chapter [LNSF19]. We publish the implementation of our algorithm¹.

4.1 Approach

Our goal is to handle unexpected objects at test time in semantic segmentation and to predict the probability that a pixel belongs to a never-seen-before class. This is in contrast to most of the semantic segmentation literature, which focuses on assigning to each pixel a probability to belong to classes it has seen in training, without explicit provision for the unexpected.

Figure 4.2 summarizes our approach. We first use a given semantic segmentation algorithm such as [BKC17b] and [ZSQ⁺17] to generate a semantic map. We then pass this map to a generative network [WLZ⁺18] that attempts to resynthesize the input image. If the image contains objects belonging to a class that the segmentation algorithm has not been trained for, the corresponding pixels will be mislabeled in the semantic map and therefore poorly resynthesized. We then identify these *unexpected* objects by detecting significant differences between the original image and the synthetic one. Below, we introduce our approach to detecting these discrepancies and assessing which differences are significant.

4.1.1 Discrepancy Network

Having synthesized a new image, we compare it to the original one to detect the meaningful differences that denote unexpected objects not captured by the semantic map. While the layout of the known objects is preserved in the synthetic image, precise information about the scene’s appearance is lost and simply differencing the images would not yield meaningful results. Instead, we train a second network, which we refer to as the *discrepancy network*, to

¹ Implementation: github.com/cvlab-epfl/detecting-the-unexpected

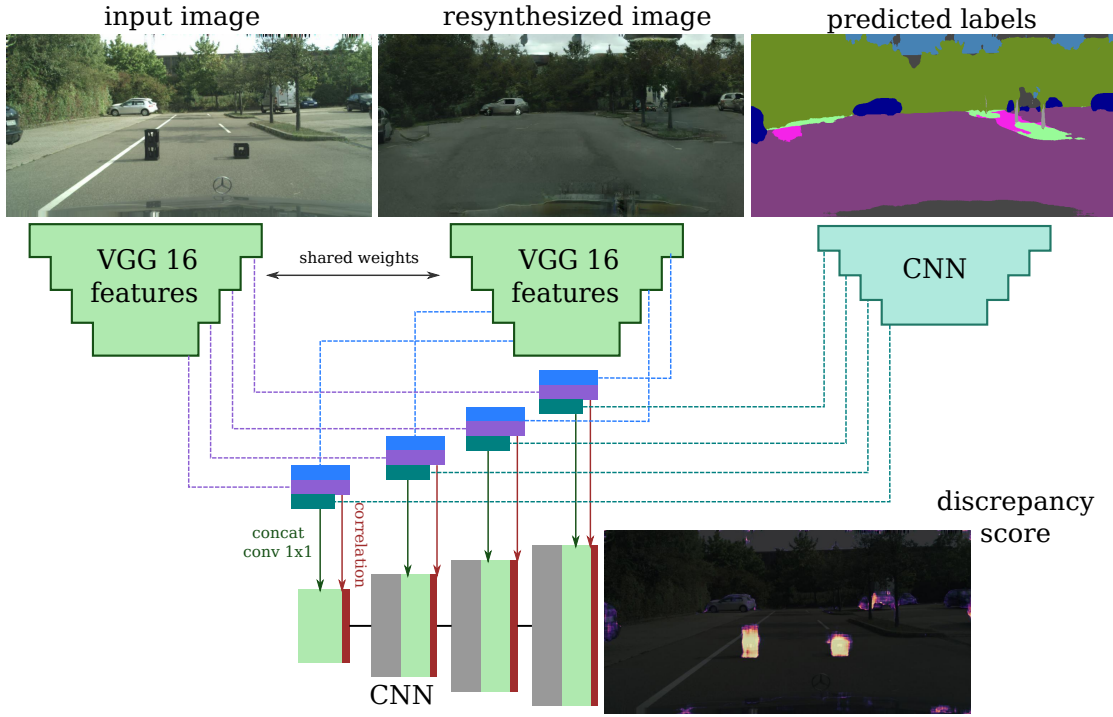


Figure 4.3: **Discrepancy network.** Given the original image, the predicted semantic labels and the resynthesized image as input, our discrepancy network detects meaningful differences caused by mislabeled objects. The VGG [SZ15] network extracts features from both images which are correlated at all levels of the pyramid. Image and label features are then fused using 1×1 convolutions. Both the features and their correlations are then fed to a decoder via skip connections to produce the final discrepancy map.

detect the image discrepancies that *are* significant.

Figure 4.3 depicts the architecture of our discrepancy network. We drew our inspiration from the co-segmentation network of [LJR18] that uses feature correlations to detect objects co-occurring in two input images. Our network relies on a three-stream architecture that first extracts features from the inputs. We use a pre-trained VGG [SZ15] network for both the original and resynthesized image, and a custom CNN to process the one-hot representation of the predicted labels. At each level of the feature pyramid, the features of all the streams are concatenated and passed through 1×1 convolution filters to reduce the number of channels. In parallel, pointwise correlations between the features of the real image and the resynthesized one are computed and passed, along with the reduced concatenated features, to an upconvolution pyramid that returns the final discrepancy score.

4.1.2 Training

When training our discrepancy network, we cannot observe the unknown classes. To address this, we therefore train it on synthetic data that mimics what happens in the presence of unexpected objects. In practice, the semantic segmentation network assigns incorrect class

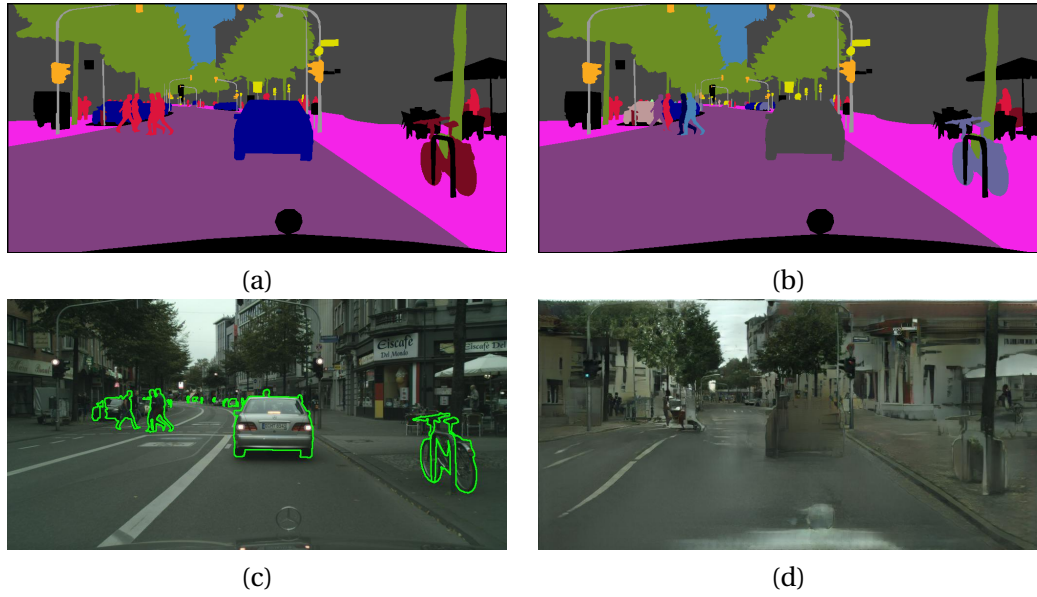


Figure 4.4: **Creating training examples for the discrepancy detector.** (a) Ground-truth semantic map. (b) We alter the map by replacing some object instances with randomly chosen labels. (c) Original image with the overlaid outlines of the altered objects. (d) Image resynthesized using the altered map. We train the discrepancy detector to find the pixels within the outlines of altered objects shown in (c).

labels to the regions belonging to unknown classes. To simulate this, as illustrated in Figure 4.4, we therefore replace the label of randomly chosen object instances with a different random one sampled uniformly from the set of *Cityscapes* evaluation classes. We then resynthesize the input image from this altered semantic map using the *pix2pixHD* [WLZ⁺18] generator trained on the dataset of interest. This creates pairs of real and synthesized images from which we can train our discrepancy network. Note that this strategy does *not* require seeing unexpected objects during training.

4.1.3 Detecting Adversarial Attacks

Comparing an input image to a resynthesized one also allows us to detect adversarial attacks. The experimental details are omitted in this thesis as this does not immediately pertain to anomaly detection but the full description is included in the conference version of this work [LNSF19].

Like for unexpected object detection, we first compute a semantic map from the input image, adversarial or not, and resynthesize the scene from this map using the *pix2pixHD* generator. Here, unlike in the unexpected object case, the semantic map predicted for an adversarial example is completely wrong and the resynthesized image therefore completely distorted, as shown in Figure 4.5. This makes attack detection a simpler problem than unexpected object one. We can thus use a simple non-differentiable heuristic to compare the input image with the resynthesized one. Specifically, we use the L^2 distance between HOG [DT05] features

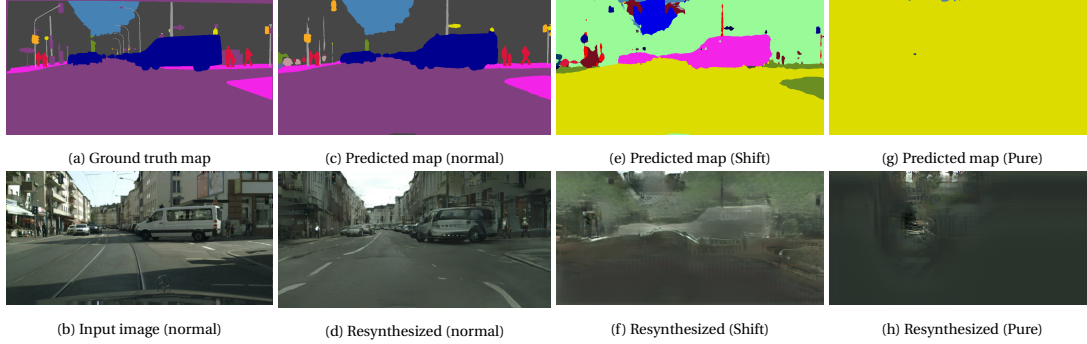


Figure 4.5: **Visualizing adversarial attacks.** Without attacks, the resynthesized image (d), obtained from (c), looks similar to the input one (b). By contrast, resynthesized images ((f) and (h)) obtained from the semantic maps ((e) and (g)) computed from an attacked input differ massively from the original one.

computed on the input and resynthesized image. We then train a logistic regressor on these distances to predict whether the input image is adversarial or not. Note that this simple heuristic is much harder to attack than a more sophisticated, deep-learning based one.

4.2 Experiments

This work also predates the *Segment Me If You Can* benchmark so the experiments are not in the benchmark format. We report the original experiments here and a further comparison on the benchmark in [Section 8.2](#).

We first evaluate our approach on the task of detecting unexpected objects such as lost cargo, animals, and rocks, in traffic scenes which constitute our target application domain and the central evaluation domain for semantic segmentation thanks to the availability of large datasets such as Cityscapes [COR⁺16] and BDD100K [YXC⁺18]. For this application, all tested methods output a per-pixel *anomaly score*, and we compare the resulting maps with the ground-truth anomaly annotations using ROC curves and the area under the ROC curve (AUROC) metric. Then, we present our results on the task of adversarial-attack detection.

We perform evaluations using the Bayesian SegNet [BKC17a] and the PSP Net [ZSQ⁺17], both trained using the BDD100K dataset [YXC⁺18] (segmentation part) chosen for its large number of diverse frames, allowing the networks to generalize to the anomaly datasets, whose images differ slightly and cannot be used during training. To train the image synthesizer and discrepancy detector, we used the training set of Cityscapes [COR⁺16], downsampled to a resolution of 1024×512 because of GPU memory constraints.

4.2.1 Baselines

As a first baseline, we rely on an uncertainty-based semantic segmentation network. Specifically, we use the Bayesian SegNet [BKC17a] which samples the distribution of the network’s

results using random dropouts; the uncertainty measure is computed as the variance of the samples. We will refer to this method as *Uncertainty (Dropout)*.

It requires the semantic segmentation network to contain dropout layers which is not the case of most state-of-the-art networks such as PSP [ZSQ⁺17] based on a ResNet backbone. To calculate the uncertainty of the PSP network, we therefore use the ensemble-based method of [LPB17]. We trained the PSP model four times, yielding different weights due to the random initialization. We then use the variance of the outputs of these networks as a proxy for uncertainty. We will refer to this method as *Uncertainty (Ensemble)*.

Finally, we also evaluate the road-specific approach of [CM15], which relies on training a shallow Restricted Boltzmann Machine autoencoder to resynthesize patches of road texture corrupted by Gaussian noise. The regions whose appearance differs from the road are expected not to be reconstructed properly, and thus an anomaly score for each patch can be obtained using the difference between the autoencoder’s input and output. As the original implementation was not publicly available, we re-implemented it and make the code available¹ for future comparisons. As in the original article, we use 8×8 patches with stride 6 and a hidden layer of size 20. We extract the empty road patches required by this method for training from the Cityscapes images using the ground-truth labels to determine the road area. We will refer to this approach as *RBM*.

The full version of our discrepancy detector takes as input the original image, the resynthesized one and the predicted semantic labels. To study the importance of using both of these information sources as input, we also report the results of variants of our approach that have access to only one of them. We will refer to these variants as *Ours (Resynthesis only)* and *Ours (Labels only)*.

4.2.2 Anomaly Detection Results

We evaluate our method’s ability to detect unexpected objects using two separate datasets described below. We did not use any portion of these datasets during training because we tackle the task of finding never-seen-before objects.

Lost and Found

The *Lost And Found* [PRG⁺16] dataset contains images of small items, such as cargo and toys left on the street, with per-pixel annotations of the obstacle and the free-space in front of the car. We perform our evaluation using the test set, excluding 17 frames for which the annotations are missing. We downscaled the images to 1024×512 to match the size of our training images and selected a region of interest which excludes the ego-vehicle and recording artifacts at the image boundaries. We do not compare our results against the stereo-based ones introduced in [PRG⁺16] because our study focuses on monocular approaches.

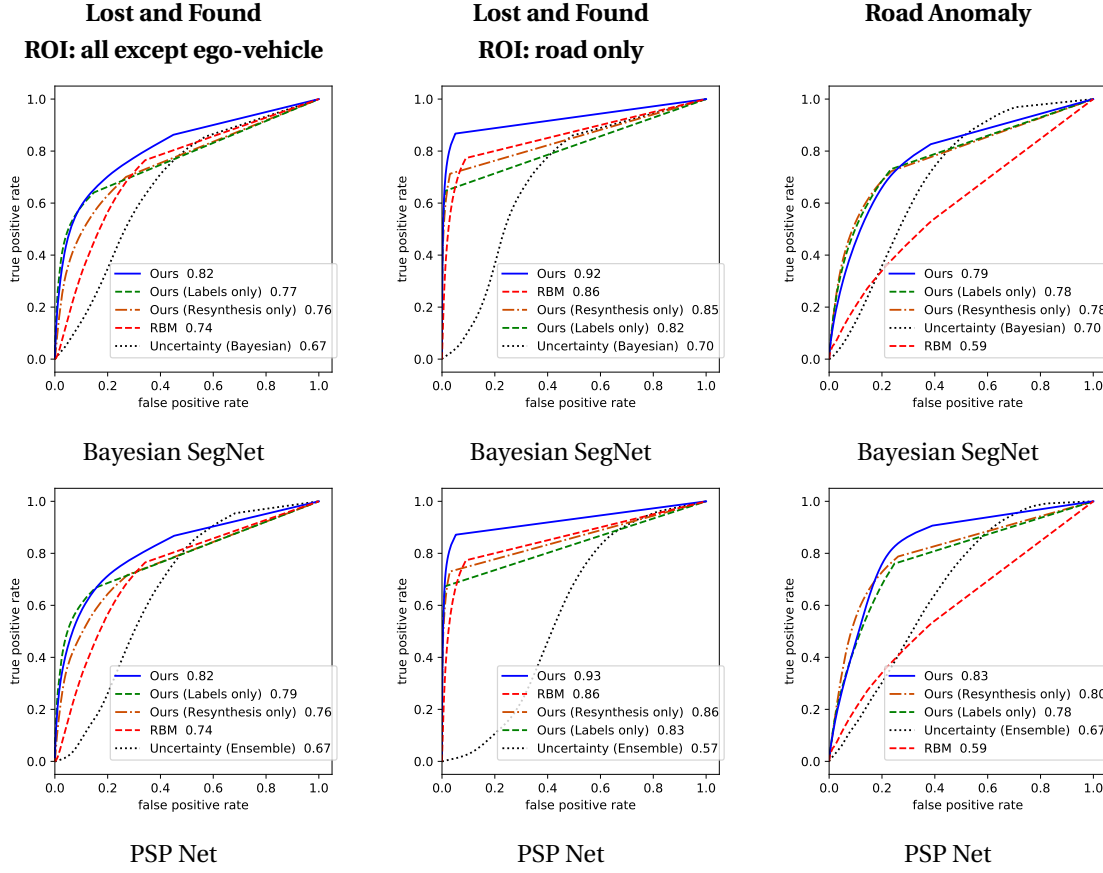


Figure 4.6: **ROC curves for unexpected object detection.** The first two columns show results for the *Lost and Found* [PRG⁺16] dataset: The curves on the left were computed over the entire images, excluding only the ego-vehicle. Those in the middle were obtained by restricting evaluation to the road, as defined by the ground-truth annotations. The right column depicts the results on our *Road Anomaly* dataset. The top and bottom rows depict the results of the *Bayesian SegNet* and the *PSP Net*, respectively. The methods are ordered according to their AUROC scores, provided on the right of the methods' name.

The ROC curves of our approach and of the baselines are shown in the left column of Figure 4.6. Our method outperforms the baselines in both cases. The Labels-only and Resynthesis-only variants of our approach show lower accuracy but remain competitive. By contrast, the uncertainty-based methods prove to be ill-suited for this task. Qualitative examples are provided in Figure 4.7. Note that, while our method still produces false positives, albeit much fewer than the baselines, some of them are valid unexpected objects such as the garbage bin in the first image. These objects, however, were not annotated as obstacles in the dataset.

Since the RBM method of [CM15] is specifically trained to reconstruct the road, we further restricted the evaluation to the road area. To this end, we defined the region of interest as the union of the *obstacle* and *freespace* annotations of *Lost And Found*. The resulting ROC curves are shown in the middle column of Figure 4.6. The globally higher scores in this scenario show that distinguishing anomalies from only the road is easier than finding them in the entire scene. While the RBM approach significantly improves in this scenario, our method still

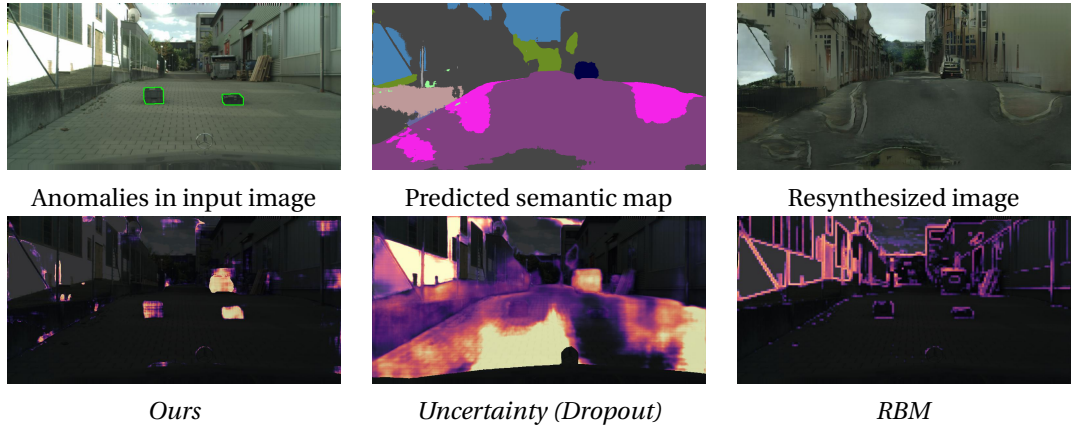


Figure 4.7: **Lost and Found results.** The top images depict algorithmic steps and the bottom ones our results along with those of the baselines. Our detector finds not only the obstacles on the road but also other unusual objects like the trash container on the right side of the road. By contrast *Uncertainty (Dropout)* reports high uncertainty in irrelevant regions and fails to localize the obstacles. *RBM* finds only the edges of the obstacles. Our approach detects the unexpected objects correctly.

	Full	Labels only	Resynthesis only
Supervised	0.94	0.93	0.96
Unsupervised	0.82	0.79	0.76

Table 4.1: **Performance of the discrepancy network in a supervised setting.** AUROC scores measured on the *Lost and Found* dataset.

outperforms it.

Our Road Anomaly Dataset

To evaluate this method we have collected the *Road Anomaly 19* dataset described in [Section 3.1.1](#). The results are shown in the right column of [Figure 4.6](#), with example images in [Figure 4.8](#). Our approach outperforms the baselines, demonstrating its ability to generalize to new environments. By contrast, the *RBM* method’s performance is strongly affected by the presence of road textures that differ significantly from the Cityscapes ones.

4.2.3 Supervised Discrepancy Network

To get an upper bound on its accuracy, we test the discrepancy network in a supervised setting. We use the ground-truth anomaly labels of the *Lost and Found* training set, with semantics predicted by PSP Net. The AUROC scores, measured on the test set, are shown in [Table 4.1](#).

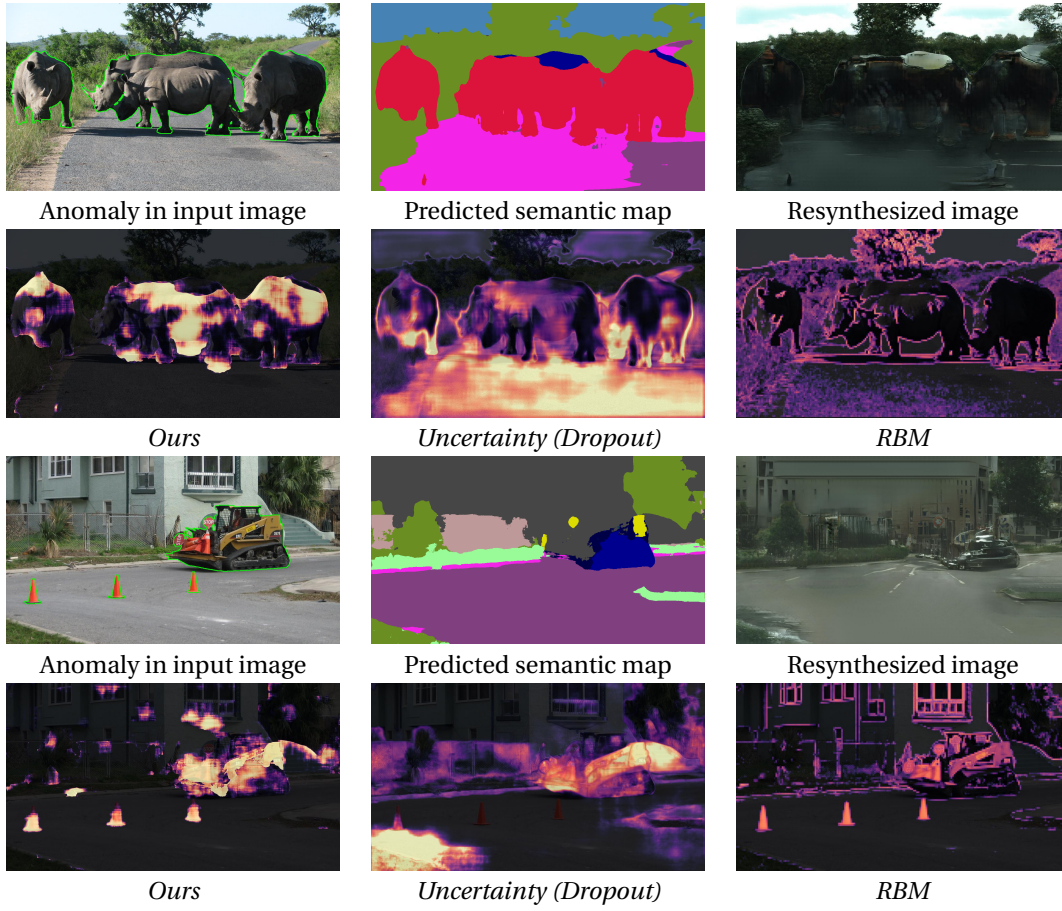


Figure 4.8: **Road anomaly results.** As in Figure 4.7, in each pairs of rows, the consecutive images at the top depict algorithmic steps and the ones at the bottom our results along with those of the baselines.

4.3 Qualitative Examples

The synthetic training process alters only foreground objects. A potential failure mode could therefore be for the network to detect *all* foreground objects as anomalies, thus finding not only the true obstacles but also everything else. In Figure 4.9, we show that this does not happen and that objects correctly labeled in the semantic segmentation are not detected as discrepancies.

In Figure 4.10, we illustrate the fact that, sometimes, objects of known classes differ strongly in appearance from the instances of this class present in the training data, resulting in their being marked as unexpected.

We present a failure case of our method in Figure 4.11. Anomalies similar to an existing semantic class are sometimes not detected as discrepancies if the semantic segmentation marks them as this similar class. For example, an animal is assigned to the *person* class and missed by the discrepancy network. In that case, however, the system as a whole is still aware of the obstacle because of its presence in the semantic map.

4.3 Qualitative Examples

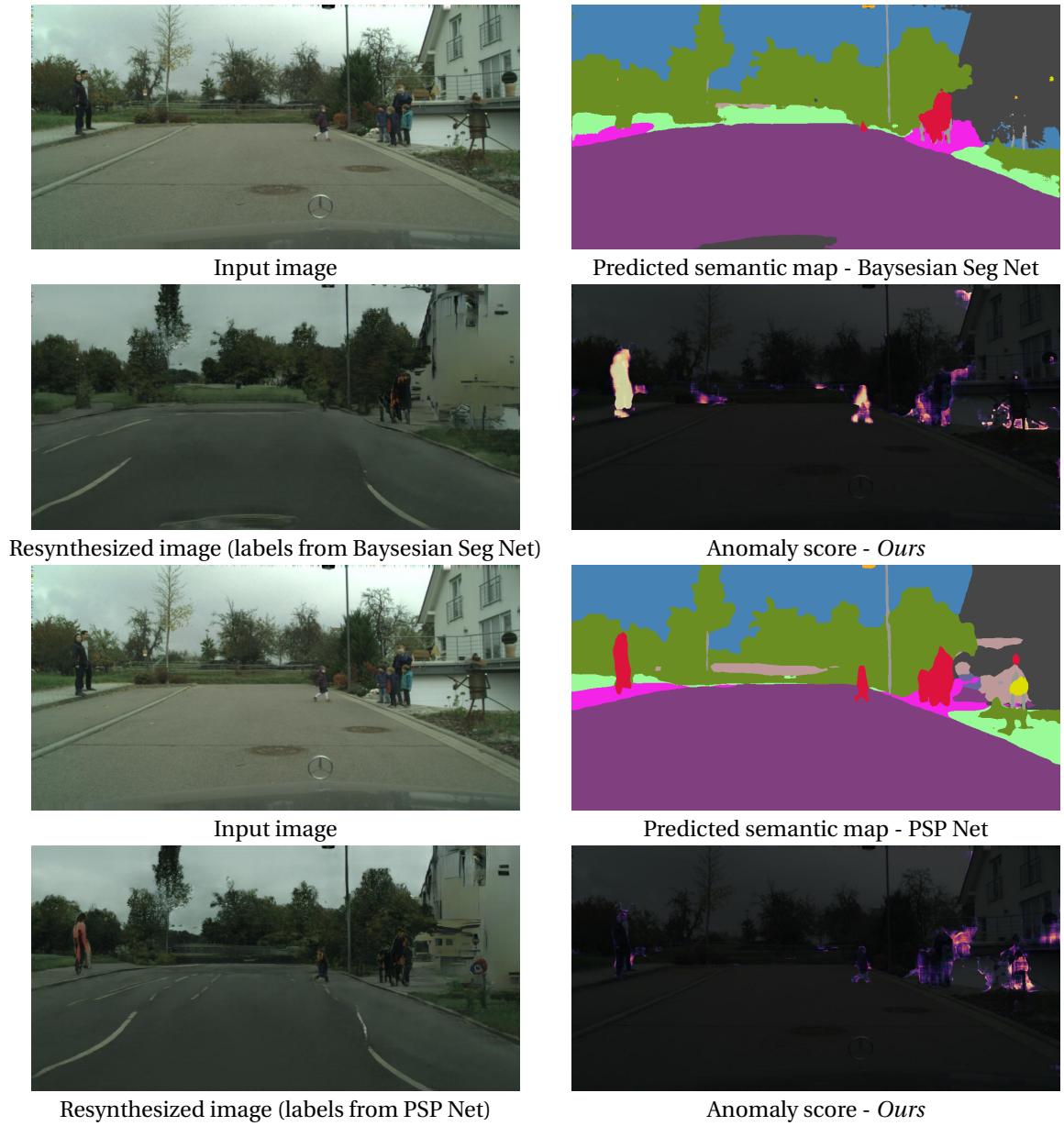


Figure 4.9: The synthetic training process alters only foreground objects, but that does not mean our discrepancy network learns to blindly mark all such objects. In the top row, we show an example where the *Bayesian SegNet* failed to correctly label some of the people present, and this discrepancy is detected by our network. However, our detector reports no discrepancy when the *PSP Net* correctly labels the people in the image (third row).

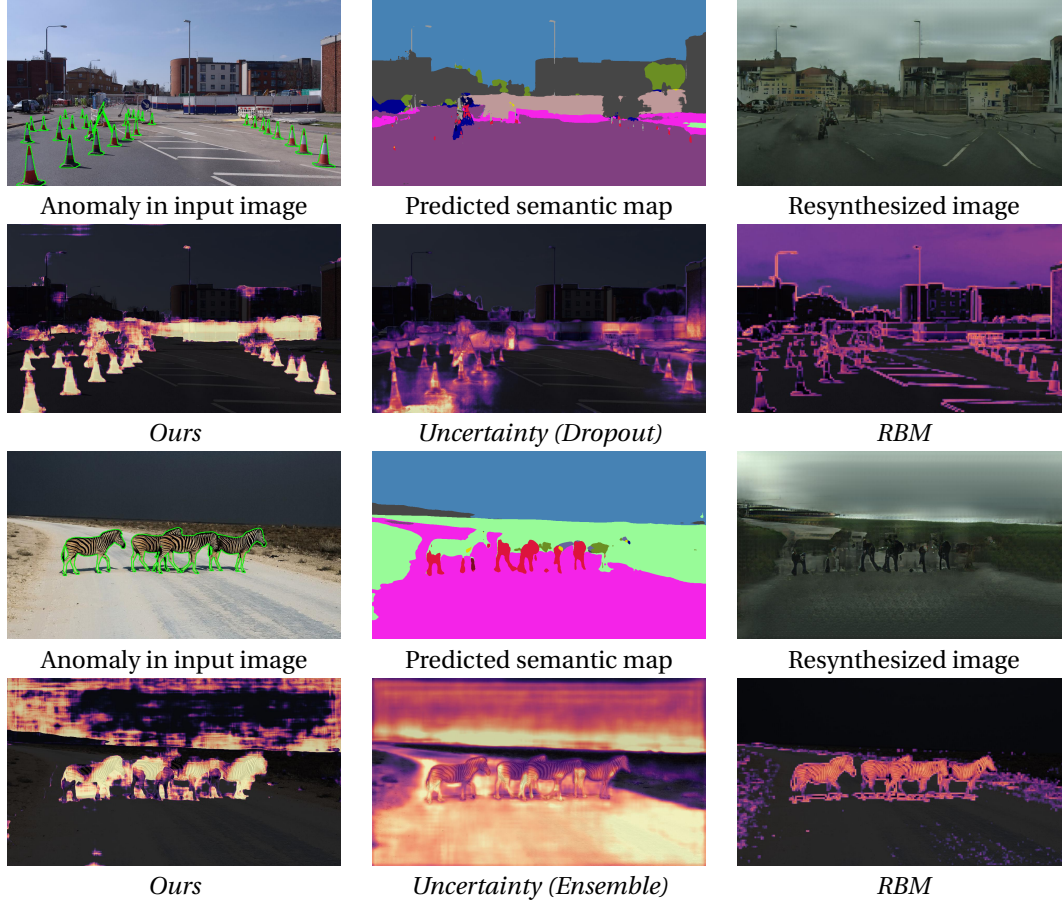


Figure 4.10: **Unusual versions of known objects.** Objects of known classes are marked as anomalies because their appearance differs from the examples of this class present in the training data, for example the fence in the first row (*fence* class) and the dark sky in the third row. Note that the *RBM* patch-based method [CM15] is especially sensitive to edges and so it detects the zebras very well.

4.3.1 Implementation details

Our discrepancy network relies on the implementations of *PSP Net* [ZSQ⁺17] and *SegNet* [BKC17b] kindly provided by Zijun Deng. The detailed architecture of the discrepancy network is shown in Figure 4.12. We utilize a pre-trained VGG16 [SZ15] to extract features from images and calculate their pointwise correlation, inspired by the co-segmentation network of [LJR18]. The up-convolution part of the network contains SELU activation functions [KUMH17]. The discrepancy network was trained for 50 epochs using the Cityscapes [COR⁺16] training set with synthetically changed labels as described in Section 3.2 of the main paper. We used the Adam [KB15] optimizer with a learning rate of 0.0001 and the per-pixel cross-entropy loss. We utilized the class weighting scheme introduced in [PCKC16] to offset the unbalanced numbers of pixels belonging to each class.

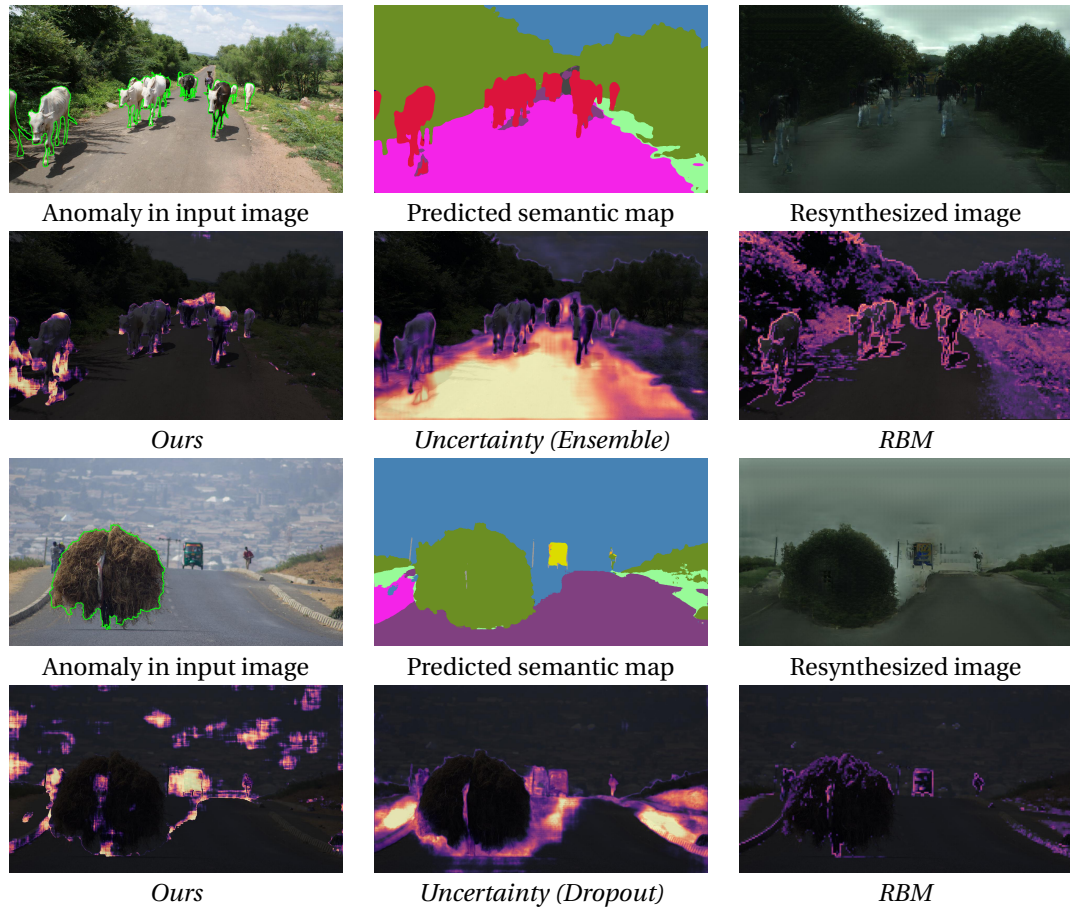


Figure 4.11: **Failure cases.** Our approach sometimes fails when the anomaly bears resemblance to an existing class: For example, animals classified as people in the first row or transported hay classified as vegetation in the third row. The system as a whole is nonetheless still aware of the obstacle because of its presence in the semantic map.

4.4 Conclusion

In this chapter, we have introduced a drastically new approach to detecting the unexpected in images. Our method is built on the intuition that because unexpected objects have not been seen during training, typical semantic segmentation networks will produce spurious labels in the corresponding regions. Therefore, resynthesizing an image from the semantic map will yield discrepancies with respect to the input image, and we introduced a network that learns to detect the meaningful ones. Our experiments have shown that our approach detects the unexpected objects much more reliably than uncertainty- and autoencoder-based techniques. Our approach still suffers from the presence of some false positives which, in a real autonomous driving scenario, would create a source of distraction. In the next chapters we will focus the scope of the obstacle detection solely on the road area itself, allowing for a higher precision of predictions.

5 Detecting Road Obstacles by Erasing Them

A self-driving vehicle needs to be able to detect strange and unexpected obstacles lying on the road. Such obstacles are as rare as they are diverse, which prevents direct application of the now standard approach of training deep networks by showing them an exhaustive set of annotated samples.

In the previous chapter we have performed the detection of *anomalies* which do not belong to Cityscapes semantic classes. However for the immediate goal of self-driving systems, the semantic meaning of the obstacles is of lesser importance as long as they can be reliably avoided. Furthermore, only objects on the road are important for path planning. In this chapter we will focus on obstacles located on the road surface and exploit that constraint to inpaint the road texture without the obstacles.

In practice, detecting such unexpected obstacles often requires LiDAR sensors [SKGK20] or multiple cameras [PRG⁺16]. Here, we propose instead a method that only needs a single RGB image to detect obstacles in the drivable area, under the assumption that objects outside that area are irrelevant because a self-driving car will detect the road before planning to drive and will not leave the drivable area of its own accord. To demonstrate this to be a viable assumption, we will show results given either the ground-truth location of the road edges or only an imperfect road segmentation produced by an off-the-shelf segmentation algorithm.

Our approach relies on the fact that obstacles look different from the surrounding road surface. We thus detect them by inpainting image-patches using their surroundings and then checking how similar the inpainted patch is to the original one. While a similar intuition has been used to detect anomalies in several application scenarios, such as detecting manufacturing defects [ZKS20, HGT18] or anomalous faces [BRF18], the very constrained nature of these tasks made it possible to rely on simple comparisons of handcrafted features. By contrast, on roads, this would yield many false positives due to road markings, diversity in road texture, and obstacles extending beyond the inpainted patch.

Like in Chapter 4, our solution is to introduce a *discrepancy network* trained to recognize



Figure 5.1: **Detecting unexpected obstacles in good and bad weather.** **Top:** Objects one would not expect to see on a road and that are not featured in standard databases. **Middle:** The road area has been inpainted. **Bottom:** After comparing the original and inpainted images, our discrepancy network returns a binary mask that denotes the obstacle locations.

which differences between the inpainted patch and the original one are significant. It returns a per-pixel heatmap denoting the presence of obstacles. To train it to handle objects that are *not* part of the training database, we generate samples featuring synthetic obstacles by moving existing training objects, such as road signs and people, onto the road.

Our experiments show that our discrepancy network trained solely on Cityscapes [COR⁺16] objects successfully detects obstacles on images depicting significantly different road scenes, without requiring any annotated data nor any re-training for these new scenes. In other words, our method generalizes well to previously unseen real obstacles and new road surfaces. It outperforms earlier monocular road anomaly detectors [BSN⁺19, MG18a, BKOS19] on the *Lost & Found* [PRG⁺16] data featured in the *Fishyscapes* benchmark [BSN⁺19], as well as on our own newly collected dataset featuring additional unusual objects and road surfaces.

Our contribution is therefore a simple but effective approach to detecting obstacles that never appeared in any training database, given only a single RGB image.

5.1 Approach

Our goal is to identify obstacles that are on the road and pose a danger of collision with the vehicle. This means that they are within the space deemed drivable by a previous stage in the self-driving perception pipeline. As such, they are the most relevant obstacles as a competent driving system will only plan trajectories within that space.

In other words, our chosen task is to identify all pixels, within that estimated road area, that denote obstacles. This is difficult because obstacles can take many forms. Furthermore, because they are unexpected, there is no guarantee that they were present in the database used to train a network to recognize them. Hence, the network must be made to respond to *objects that does not belong to the road* without any clear description, or even exemplars, of these objects.

To this end, given a binary mask denoting the drivable area in the image, we propose the following two-step approach:

1. Erase the obstacles by removing road patches and inpainting them in a sliding-window manner;
2. Use a discrepancy network to compare the original image to the inpainted one and decide if they are similar enough.

The intuition behind this scheme is that, if there is an obstacle, the inpainted area will look very different from the original image. However, even if there is no obstacle, the inpainted area will be similar to the original one, but not strictly equal. Hence, the discrepancy network is needed to assess if they are dissimilar enough to flag a potential obstacle. It yields a heatmap denoting the likelihood for each pixel in the drivable area of belonging to an obstacle. In the remainder of this section, we discuss these two steps in more details.

5.1.1 Drivable Area

A self-driving system must determine the road area within which it can move. Our method detects obstacles within the area identified as drivable, as these are the only ones that might be on the vehicle's planned path and can endanger it. Of course, a part of the road may be mistakenly marked as non-drivable, causing obstacles it contains to be ignored as shown [Figure 5.8](#). However, this does not compromise safety because the vehicle will never attempt to go there.

Our approach can exploit any method that delivers the required drivable area information. In our experiments, we use the PSP-Net semantic segmentation network of [ZSQ⁺17] trained on the Cityscapes dataset [COR⁺16], as implemented in the framework of [GHH⁺20]. We take the road area to be all pixels classified as either *road* or *sidewalk*, since the many road textures

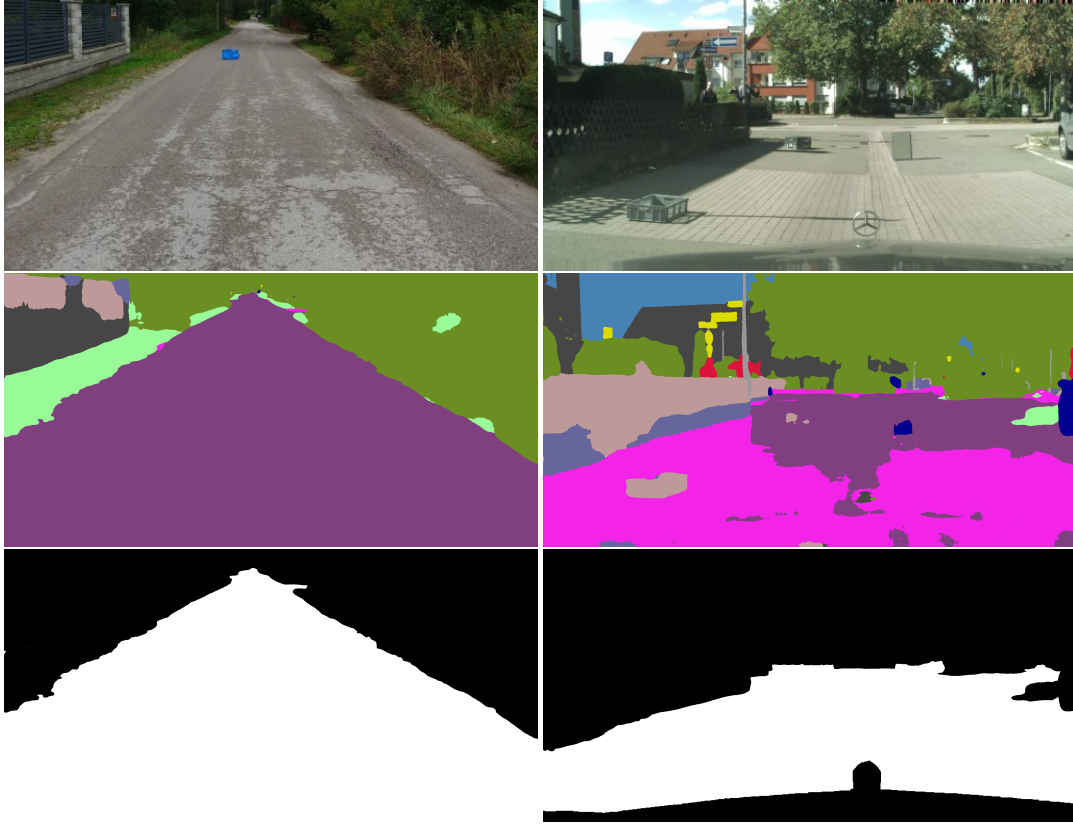


Figure 5.2: **Drivable space from semantic segmentation.** **Top:** Input images. **Middle:** Semantic segmentation performed by PSP-Net [ZSQ⁺17], the class colors follow Cityscapes convention. **Bottom:** We take the drivable space to be the union of *road* (purple) and *sidewalk* (magenta) pixels. The parts of obstacles can sometimes be classified as non-road, so we include the regions of other classes fully enclosed within the road area. In *Lost & Found*, the known ego-vehicle mask is excluded.

we are targeting can be classified as either. Note that standard categories, such as *car* and *pedestrian*, are inherently accounted for by PSP-Net. Hence, we focus on the unusual obstacles for which *no* training data, either supervised not unsupervised, is available. Nevertheless, since such unusual obstacles could be partially classified as non-road, we include the regions containing other classes that are fully enclosed within the road area. Figure 5.2 demonstrates this process. As a limit case, we also evaluate the case of perfect road detection by using the ground truth road mask.

Our approach needs only a coarse mask of the drivable area, so in a practical deployment the semantic segmentation can be replaced by a computationally more efficient system, for instance predicting just the road edges and filling the space between them.

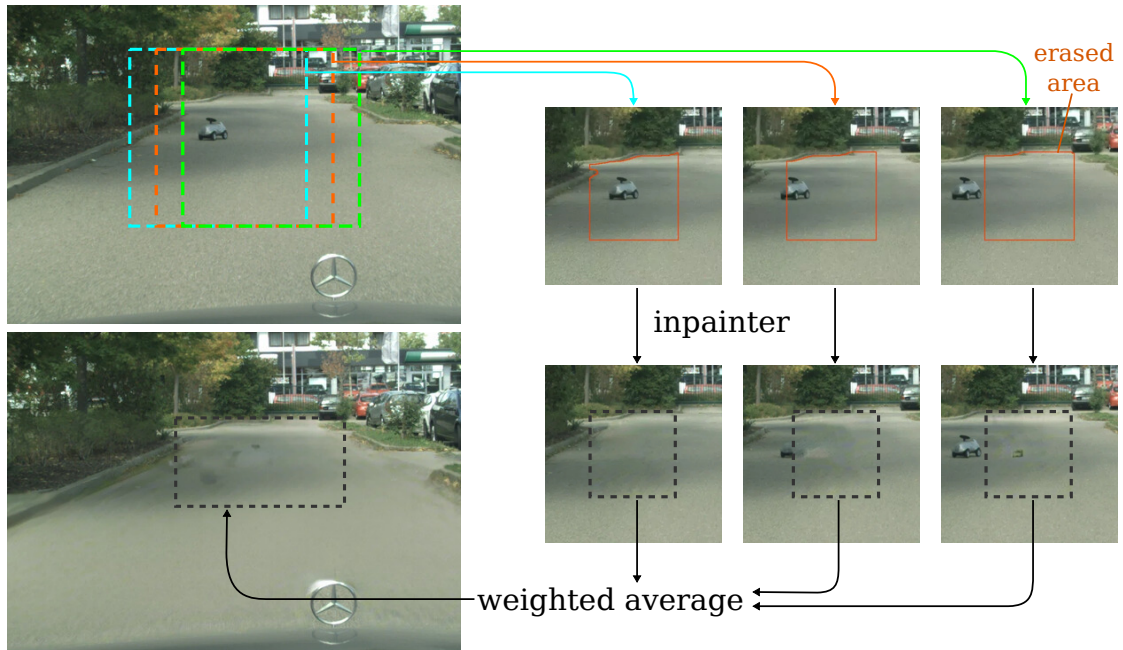


Figure 5.3: **Sliding window inpainting of the road surface.** We extract 400×400 context patches then erase and inpaint the road area contained within the central 200×200 of the patch. Finally we fuse the inpaintings which reconstructs the road appearance while removing localized obstacles. Note how the process was able to preserve the shadow of the trees.

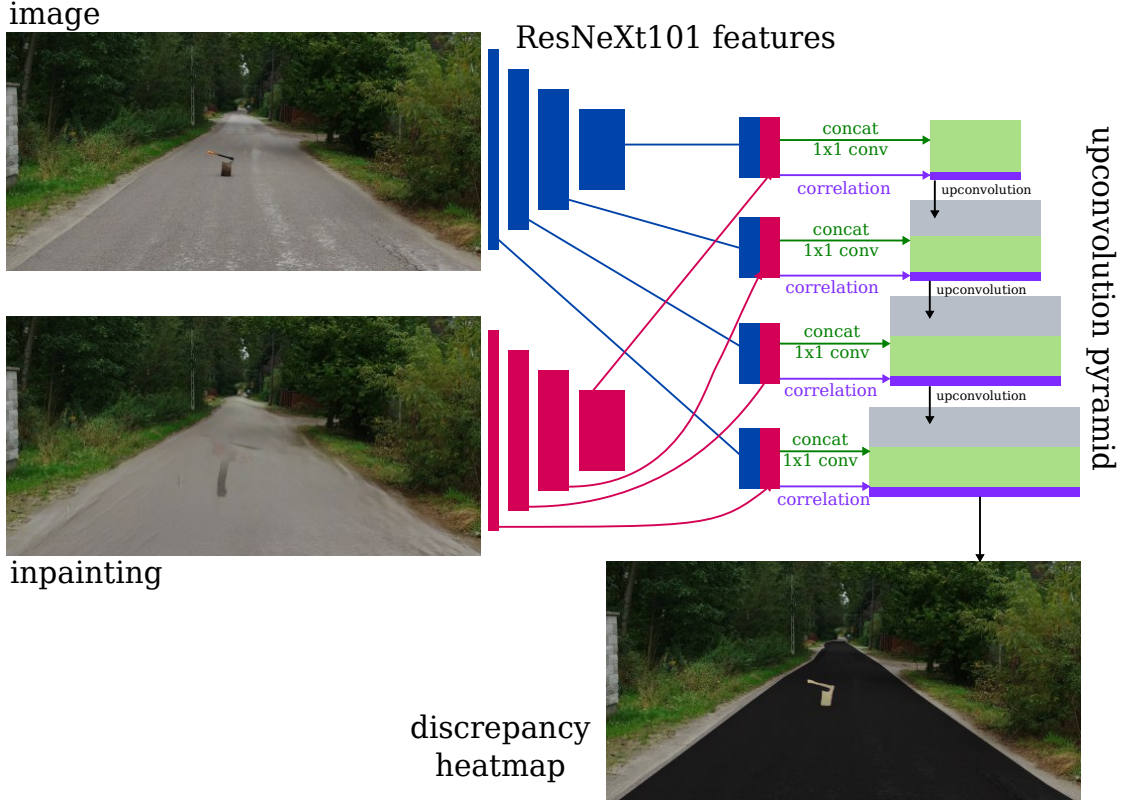
5.1.2 Inpainting

To erase the obstacles while preserving the surrounding road appearance, we use a general-purpose inpainter [YLY⁺19] that relies on an adversarial approach to ensure that the inpainted image looks realistic. We use a version of this model trained on the scene recognition dataset Places2 [ZLK⁺17], and do not train it further.

The inpainter is given an image in which a part has been replaced by black pixels and the pixel mask of the removed area. It outputs an image with the missing part filled in so as to best maintain the continuity and content of the scene.

A naive way to use it would be to inpaint the entire road area at once. This, however, would provide no indication to the inpainter of the road appearance, leading to inpainted images that differ from the original ones in the whole road region, thus precluding subsequent obstacle detection. Instead, we inpaint road patches to provide sufficient context for the network to reconstruct the road surface, as shown in Figure 5.3. The patches nonetheless need to be large enough to encompass obstacles whose size we do not know *a priori*. We therefore follow a sliding-window approach, inpainting patches of 200×200 pixels of drivable area within 400×400 image regions to provide context.

While an obstacle is usually nicely erased when the area to inpaint encloses it completely, the inpainter is able to re-create the obstacles that are only partially contained in the inpainted


 Figure 5.4: **Architecture of the discrepancy network**

region. To resolve this, we use consecutive patches with a relative overlap of 0.7, increasing the likelihood of having at least one patch that covers the entire obstacle. This means that each image pixel is inpainted multiple times. We then fuse the multiple inpaintings of each pixel by weighted averaging, where the weight of each inpainting is computed based on the Manhattan distance between the corresponding patch center and the pixel location of interest. Formally, a patch centered at location $\mathbf{c}_j = [u_j, v_j]^\top$ contributes to the inpainting of a pixel at location $\mathbf{p} = [u, v]^\top$ with a weight

$$w = \frac{1 - \frac{2}{s} \max(|u - u_j|, |v - v_j|)}{\sum_{[u_i, v_i] \in \Pi(u, v)} 1 - \frac{2}{s} \max(|u - u_i|, |v - v_i|)}, \quad (5.1)$$

where s is the patch width or height and $\Pi(u, v)$ is the set of patches overlapping the $[u, v]^\top$ pixel, with patch i centered at $[u_i, v_i]$.

Our inpainting strategy enables us to generalize to new road surfaces such as those of our Road Obstacles dataset. By contrast, the GAN synthesizers of Chapter 4 or [XZL⁺20, DBBSC21] always produce images resembling the Cityscapes training set.

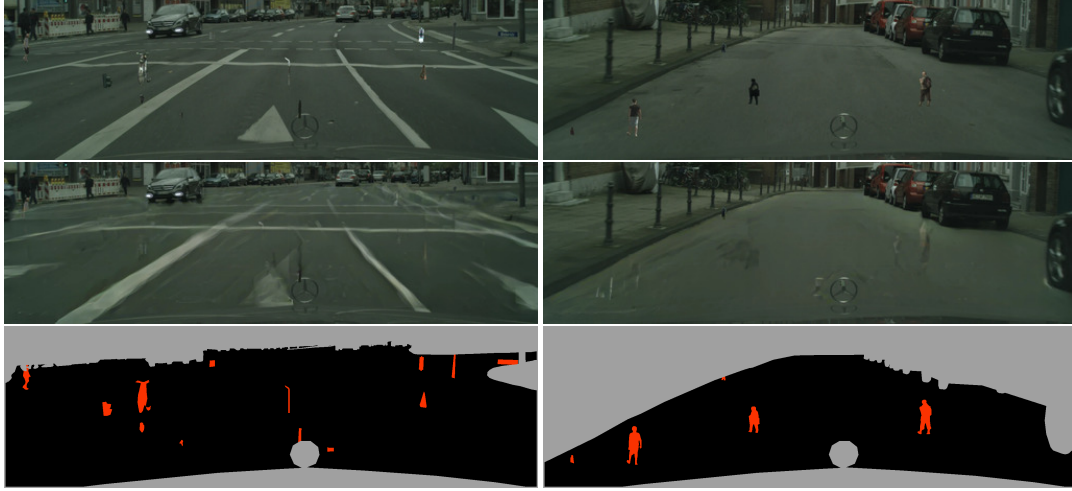


Figure 5.5: **Synthetic training obstacles.** Using the Cityscapes dataset, we transplant random object instances onto the road to appear as small obstacles (top). Results of the inpainting process (middle). Labels (bottom): the discrepancy network is trained to distinguish obstacles (red) from the road area (black), while the background grey region is ignored in training.

5.1.3 Discrepancy Network

While our inpainting strategy preserves the general appearance of the road surface, it still yields unavoidable imperfections due to road markings, texture details, and the non-zero contributions of obstacles located close to the patch edges. Thus, simply comparing the original image with the inpainted one via pixel difference would yield many false positive detections. To handle this, we introduce a *discrepancy network* that we train to distinguish significant differences from inpainting artifacts.

We implement our discrepancy network using a two-stream architecture, shown in Figure 5.4, that takes as input the original image and the inpainted one. Both inputs are first processed by a ResNeXt101 [XGD⁺17] feature extractor, pretrained for ImageNet [DDS⁺09] and frozen at training time. The resulting features are then concatenated and fused through 1×1 convolutions. Furthermore, we compute a point-wise correlation map: At each location in the feature map, we calculate the dot product between the image feature vector and the corresponding inpainting feature vector. We append these dot products as an additional channel to the output of the 1×1 convolutions. The concatenated features are then passed to an upconvolution pyramid, and we obtain the desired heatmap via a softmax.

Training the Discrepancy Network

Recall that we target unusual road obstacles that may never have been seen at training time. Therefore, we need the discrepancy network to generalize to previously-unseen objects.

To tackle this challenge, we built a synthetic training set from only the Cityscapes [COR⁺16] dataset, which contains no unusual traffic obstacles. We extracted instances of people and



Figure 5.6: **Incorporating noise.** We augment the training images (left) by adding noise summed at two scales and magnitudes (right) to simulate a diverse road texture and prevent the discrepancy network from becoming excessively sensitive to high frequencies.

vehicles using the instance annotations, together with *traffic lights* and *traffic signs*, which lack instance labels, but can be extracted as connected components within their pixel-wise semantic label mask. Since many road obstacles are small and difficult to detect, to simulate small obstacles seen from a far distance, we selected from the whole dataset instances of size ranging from 10 to 150 pixels, and area between 100 and 5000 pixel squared. We then sampled random objects from this database and overlaid them onto the drivable area to mimic obstacles. Figure 5.5 features several images synthesized in this way.

Texture Augmentation

The Cityscapes training set features predominantly smooth road surfaces. Test images depicting road surfaces rougher than those seen in training could lead to false positives. We address this issue by adding noise at two different scales to the training images to create a more realistic texture, as shown in Figure 5.6. We will evaluate the influence of this step in Section 5.2.5.

5.2 Experiments

This work also predates the *Segment Me If You Can Benchmark* so the experiments are not in the benchmark format. We report the original experiments here and a further comparison on the benchmark in Section 8.2.

5.2.1 Evaluation Metrics

Since we focus on detecting obstacles on the road, we take the Region of Interest (ROI) for evaluation purposes to be the ground-truth road area as shown in Figure 5.7. This restriction

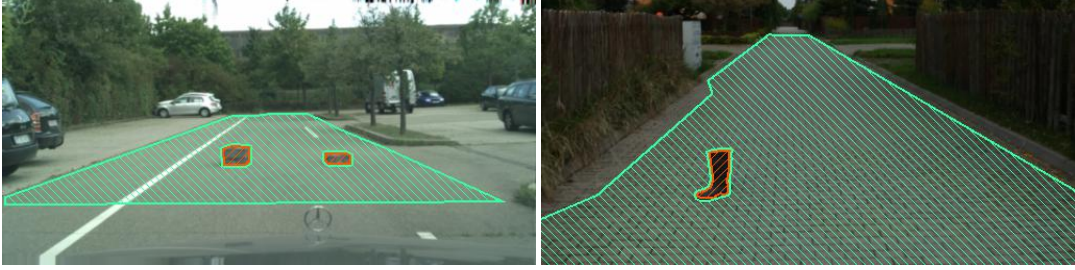


Figure 5.7: **Example ground-truth labels.** We consider the task of distinguishing obstacle pixels (orange) from the road area (light green), while the background (unmarked) is excluded from the evaluation.

of ROI to drivable space follows the evaluation protocol of the original *Lost & Found* [PRG⁺16] road-obstacle benchmark and matches the area relevant to self-driving, that is, the road area where the vehicle is going to move and can encounter obstacles. We formulate our task in terms of pixel-wise binary classification. Our method, like the baselines we compare with, outputs a heatmap in the $[0, 1]$ range denoting the likelihood for each pixel within the ROI of belonging to an obstacle. We use the following two metrics of the *Fishyscapes* benchmark [BSN⁺19]:

- The primary metric is Average Precision (AP), that is, the area under the precision-recall curve. This metric is more meaningful than metrics based on the receiver operating curve (ROC) due to strong class imbalance, as obstacles typically cover less than 2% of the total road surface.
- A secondary metric is false positive rate (FPR) at a 95% true positive rate (TPR), which we denote as FPR_{95} . To compute it, the binary classification threshold is lowered until 95% of the obstacle pixels are detected and we then measure how many false positives are introduced.

5.2.2 Baselines

The baselines used in this chapter follow the ones used in the *Fishyscapes* benchmark [BSN⁺19] since it provided a comprehensive set of methods before *Segment Me If You Can* was introduced later. Therefore we introduce the baselines used in our original experiments.

Learned Embedding Density [BSN⁺19] learns the inlier distribution of features extracted from a DeepLab [CPI⁺18] layer. It then maps the features to latent, Gaussian-distributed vectors via a normalizing flow. The mapping is trained to maximize the likelihood of the features observed in inlier samples.

Void Classifier [LLS18a] uses the Cityscapes *void* areas as examples of outliers. It can then either explicitly add *void* to the set of predicted classes, or learn to maximize the softmax entropy in the *void* regions.

Chapter 5. Detecting Road Obstacles by Erasing Them

MC Dropout [MG18b] introduces dropout layers into the DeepLab network. At inference time, it draws samples by randomizing the dropout. The uncertainty is measured as the mutual information between the resulting distribution and the network weights.

Entropy Maximization [CRG21b] trains the segmentation network to maximize the output entropy on explicit OOD samples obtained by adding COCO [LMB⁺14] objects into Cityscapes frames. It also performs post-processing on connected components of obstacle pixels, but this part is not applicable in our per-pixel evaluation.

SynBoost [DBBSC21] expands the resynthesis approach by providing uncertainty estimates of the semantic segmentation as an additional input to its dissimilarity network, which predicts the anomaly score.

DeepLab Softmax Entropy [LLS18a] measures the entropy of the class likelihoods produced by the softmax layer of the DeepLab semantic segmentation network.

Dirichlet DeepLab [MG18a] outputs the α parameters defining a Dirichlet distribution over the class labels, rather than a single set of class likelihoods given by the softmax. The network is trained to produce sharp distributions for inlier classes and a uniform distribution for the *void* class. The anomaly score is calculated as the Dirichlet differential entropy.

Discriminative Outlier Detection Head [BKOS19] relies on an outlier detection head that shares backbone features with a semantic segmentation head. The network is trained using frames from Cityscapes and Vistas [NORK17], with injected outliers drawn from ImageNet-1k [DDS⁺09]. We evaluate a variant where outlier size is randomized in order to handle a range of obstacle sizes.

5.2.3 Datasets

We report empirical results on two datasets, the well-established *Fishyscapes Lost & Found* benchmark and a new *Road Obstacles* dataset that we introduce in this paper to increase the diversity of road obstacles and surfaces.

Fishyscapes Lost & Found

The Lost & Found dataset [PRG⁺16] contains image sequences captured by a vehicle approaching lost cargo items placed on parking lots and streets. A subset of 100 validation and 275 test images was selected from the Fishyscapes [BSN⁺19] road scene anomaly detection benchmark to avoid non-anomalous Cityscapes objects such as people. We adapted the benchmark

5.2 Experiments

Method	Road Obstacles daylight		Road Obstacles snowfall		Road Obstacles all		Fishyscapes: L&F validation		Fishyscapes: L&F test	
	AP ↑	FPR ₉₅ ↓	AP ↑	FPR ₉₅ ↓	AP ↑	FPR ₉₅ ↓	AP ↑	FPR ₉₅ ↓	AP ↑	FPR ₉₅ ↓
Ours - road ground truth	96.5	0.0	67.9	6.0	91.1	0.3	93.2	1.0	92.7	0.9
Ours - road segmentation	96.8	0.0	65.1	6.2	90.8	0.4	91.8	1.5	85.8	45.3
SynBoost [DBBSC21]	84.0	0.9	47.8	16.4	79.7	2.4	89.8	1.5	81.6	3.4
Maximized Entropy [CRG21b]	96.1	0.1	30.5	12.5	90.1	0.3	74.5	11.8	78.0	15.2
Outlier Detection Head [BKOS19]	70.8	0.9	0.7	54.0	30.4	5.0	61.4	27.0	53.1	36.1
Resynthesis (Chapter 4)	34.5	4.1	19.2	15.0	31.9	5.9	64.4	5.1	67.0	6.0
Dirichlet DeepLab [MG18a]	11.7	31.6	0.6	45.5	4.8	37.8	61.2	70.8	59.9	73.0
MC Dropout [MG18b]	13.0	49.1	0.8	51.6	11.3	47.8	48.4	33.7	41.3	30.6
Void Classifier [BSN*19]	8.7	48.6	2.9	22.6	10.5	41.2	14.1	24.2	4.7	41.0
Embedding density - Minimum NLL [BSN*19]	2.3	27.8	0.9	38.6	2.1	33.7	61.1	8.5	71.7	6.7
Embedding density - Single-layer NLL [BSN*19]	1.0	62.0	0.1	84.1	0.9	71.2	49.6	17.2	55.8	13.6
DeepLab Softmax [LLLS18a]	26.2	10.9	2.9	60.9	22.6	14.1	29.4	41.0	27.5	29.4

Table 5.1: **Obstacle detection scores.** The primary metric is *average precision* of detecting obstacle pixels.

Method	Road Obstacles daylight		Road Obstacles snowfall		Road Obstacles all		Fishyscapes: L&F validation		Fishyscapes: L&F test	
	AP ↑	FPR ₉₅ ↓	AP ↑	FPR ₉₅ ↓	AP ↑	FPR ₉₅ ↓	AP ↑	FPR ₉₅ ↓	AP ↑	FPR ₉₅ ↓
Ours	96.8	0.0	65.1	6.2	90.8	0.4	91.8	1.5	85.8	45.3
Resynthesis	83.8	0.6	45.0	8.9	82.8	1.6	68.3	1.6	56.1	61.9
No Inpainting.	91.7	0.2	73.2	0.7	86.6	0.6	86.0	1.7	81.3	99.9
No Discrepancy	15.3	40.2	28.8	36.2	14.1	35.7	21.1	48.4	23.1	84.8
Segmentation Alone	13.4	85.1	0.3	98.2	12.9	90.6	34.4	91.8	42.6	91.2
Ours w/o noise aug.	95.9	0.0	56.9	17.0	89.2	1.6	87.1	2.1	81.8	56.1
Ours w/o correlation layer.	95.8	0.1	61.0	6.2	89.8	0.7	85.4	1.7	72.6	60.4

Table 5.2: **Ablation study results.**

to the task of obstacle detection on known road regions by restricting the evaluation to the ground-truth labeled drivable area. This was done in close collaboration with the Fishyscapes benchmark organizers. They will integrate this evaluation strategy in the benchmark but are neither authors of this paper nor associated to the corresponding research. The results of the baseline methods were computed by them, using the original implementations submitted to the benchmark that were presumably tuned by their respective authors for the benchmark and its Lost&Found part.

We evaluate both with the publicly available validation set and the private test set. Scores for the latter were computed by the benchmark authors and given to us. They are consistent with those we obtained on the validation set.

The FS-Web and FS-Static parts of Fishyscapes are created by synthetic object injection and, in that sense, resemble our training set. We exclude those from testing to prevent the detector from exploiting artifacts caused by the synthetic injection process.

Road Obstacles 20.

To evaluate this method we have collected the dataset described in Section 3.1.1. We report the results for *snowfall* track separately, as such conditions are far outside the training domains of any of the evaluated methods. We report the union of both tracks as *Road Obstacles - all*.

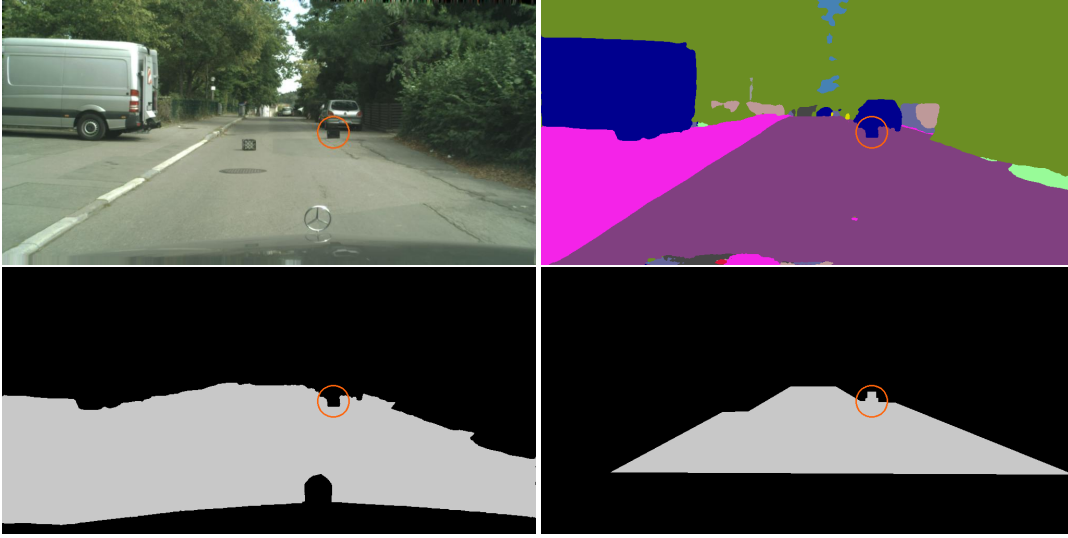


Figure 5.8: **Obstacle outside of predicted drivable area failure mode.** **Top-left:** An obstacle is located near a vehicle shadow. **Top-right:** The semantic segmenter classifies it as part of the background. **Bottom-left:** Consequently it is not included in the drivable space where our obstacle detector is applied. **Bottom-right:** Ground truth road label used as the region-of-interest for evaluation purposes. This obstacle will be counted as missed.

5.2.4 Comparative Results

We tested two versions of our method, one that operates on the road area given in the ground truth and the other on the drivable area segmented by the network of [ZSQ⁺17]. They are referred to as *Ours road ground truth* and *Ours road segmentation*, respectively, in Table 5.1. A qualitative comparison and further examples of our method’s outputs are shown in Figure 5.10 and Figure 5.9, respectively.

Both versions of our method outperform all others in terms Average Precision (AP), the primary metric in [BSN⁺19], on both datasets. The same is true for the FPR₉₅, the secondary metric, except for “Ours road segmentation” on the test set of Fishyscapes for which our FPR score is abysmal. Note, however, that the score using the ground truth is excellent. This points to the cause of the problem: The segmentation algorithm we use only found a part of the drivable area and, since we only look for obstacles there, we missed all those that were elsewhere, making it impossible to reach a 95% TPR in some of the images. Figure 5.8 illustrates this problem. Nevertheless, even in this situation, safety is maintained because the vehicle controller will only drive within the predicted road space. Hence, obstacles outside of it do not pose a risk of collision. Furthermore, as evidenced by the excellent results of *Ours road ground truth*, this problem will gradually fade away as road boundary detection algorithms improve. In the *snowfall* track, the difficult weather causes nearly all baselines to fail, but our approach still achieves reasonable performance.

While most methods including ours use solely the Cityscapes dataset for training, some others [BKOS19, CRG21b] sample the objects injected as synthetic obstacles from other datasets

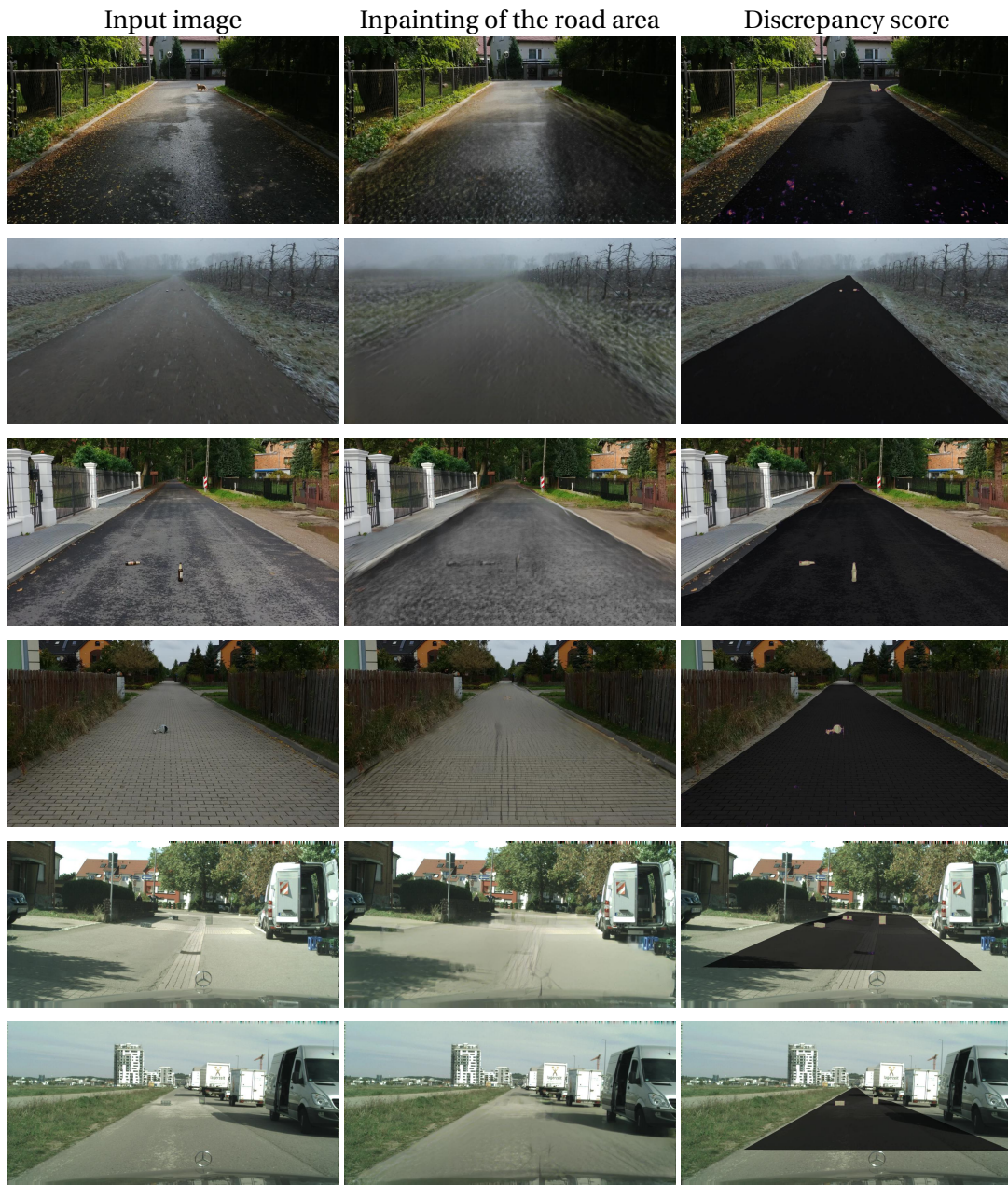


Figure 5.9: **Example outputs of our method.** **Left:** Input images featuring challenging or distant obstacles. **Center:** The result of sliding-window inpainting of the road area. **Right:** The discrepancy score calculated by our network given the two previous images. The darkened area corresponds to the ground-truth drivable space.

Chapter 5. Detecting Road Obstacles by Erasing Them

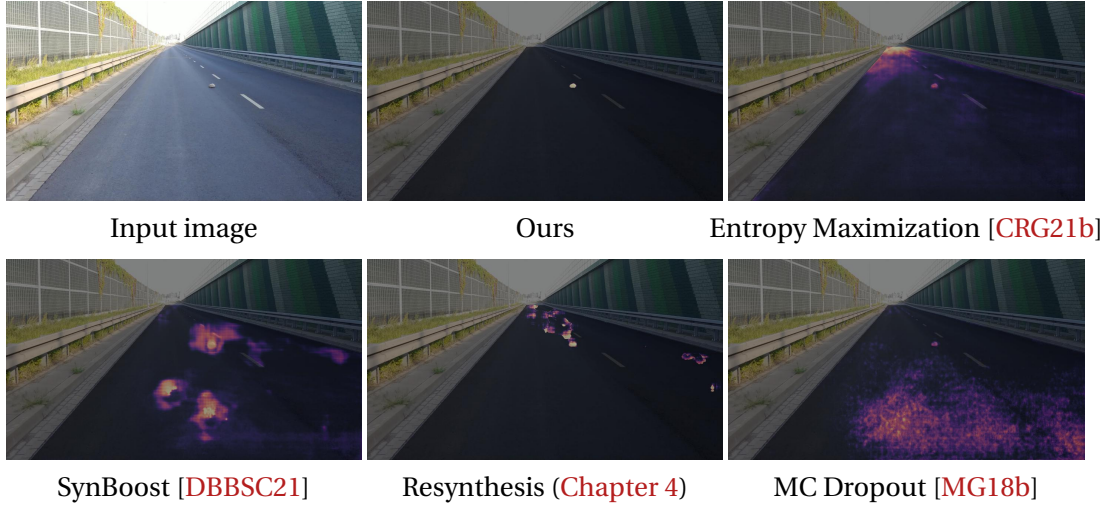


Figure 5.10: **Qualitative comparison** of method outputs. The darkened area corresponds to the ground-truth drivable space.

such as COCO or ImageNet-1k. This strategy works well for some obstacle types, particularly the distinct and colorful ones in the *Road Obstacles - daylight* set, but falls short in the *Fishyscapes* set. In other words, extending the original dataset with others is effective if the test obstacles are similar to the ones in the additional dataset but does not necessarily promote generalization.

5.2.5 Ablation study

In [Table 5.2](#), we report the results of an ablation study during which the discrepancy network was retrained with selected components altered or disabled.

In the *Resynthesis* variant, the inpainting described in [Section 5.1.2](#) is replaced by an image synthesizer [WLZ⁺18] from predicted semantic labels as in [Chapter 4](#). While the inpainter can reconstruct novel road textures based on the visual context, the generator produces a texture similar to the training roads, and this is reflected by degraded performance on *Road Obstacles*.

The *No Inpainting* variant does away with inpainting. We keep the architecture unchanged, but pass two copies of the image into both input streams and retrain the network. This also degrades performance, thus confirming the importance of inpainting. We can also do the reverse, that is, remove the discrepancy network, and compute the L_1 distance between the RGB values of the input image and the inpainted result. The results of the corresponding *No Discrepancy* variant are even worse.

The *Segmentation Alone* entry corresponds to detections made by finding groups of non-road pixels enclosed within the road region detected by the segmentation algorithm [ZSQ⁺17]. This by itself clearly fails, thereby justifying the extra step we propose in this paper.

The *snowfall* subset differs from the Cityscapes training set so significantly that the road detection is imprecise and road edges are inpainted onto its surface, which confuses the discrepancy network. This yields a higher score for the *No Inpainting* variant than for the full approach.

We also show the contribution of the texture augmentation strategy described in Section 5.1.3 and that of using the feature correlation map.

5.2.6 Implementation details

In this section, we present details on the discrepancy network architecture and training.

Discrepancy Network Architecture

The architecture is shown in Figure 5.11. The network has two input streams: the original image and the image where the road area has been inpainted. We use the pretrained ResNeXt101 network of [XGD⁺17] to extract features from both images. We use the PyTorch implementation [Pyt, PGM⁺19] of this backbone and take the outputs of layers labeled `relu`, `layer1`, `layer2`, `layer3`.

At four levels of the feature pyramid, we fuse the two streams of features in these two parallel ways:

- Concatenate stream 1 and 2, followed by a 1×1 convolution,
- Calculate pixel-wise correlation of features, following [LJR18].

The results of the above are concatenated and passed on to an up-convolution pyramid which uses the SeLU [KUMH17] activation function. In the final step, the discrepancy score is multiplied by the binary drivable space mask, since the outputs are only valid within the road area.

Discrepancy Network Training

The discrepancy network was trained for 65 epochs. Each epoch iterates over the 2975 frames of our synthetic training set. The training is done using 768×384 crops of the road area. To improve training reproducibility, we pre-define the crops and their ordering in each epoch, and train all variants of the discrepancy network with the same sequence of samples.

We use binary cross entropy loss and the Adam [KB15] optimizer. We set the initial learning rate to 10^{-4} and then adjust it dynamically, if there is no improvement of validation loss for 5 consecutive epochs, the learning rate is reduced 10 times. We generate the validation set from Cityscapes validation subset in the same way as the training set.

5.3 Conclusion

We have introduced a pipeline capable of detecting road obstacles in driving scenarios, given only single monocular images as input. We perform inpainting of the drivable area, erasing the localized obstacles while preserving the road surface. Our discrepancy network learns to accurately detect the removed obstacles and ignore irrelevant artifacts of reconstruction. This detector, trained only with synthetically altered Cityscapes data, is capable of generalizing to a variety of real-world obstacles and road surfaces. We have demonstrated this on the *Fishyscapes Lost & Found* benchmark, as well as on our newly collected dataset.

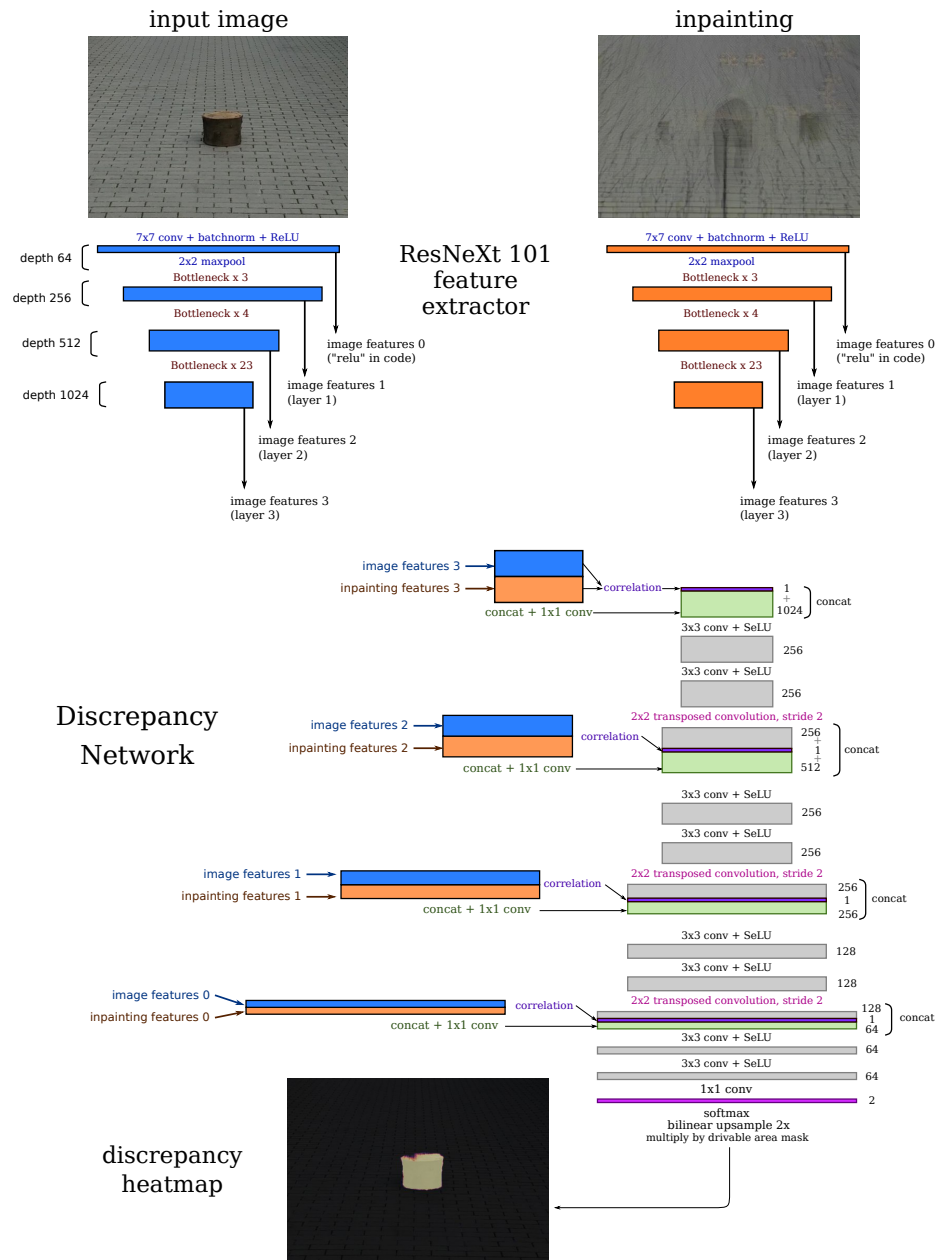


Figure 5.11: Discrepancy network architecture.

6 Perspective Aware Road Obstacle Detection¹

Our previous efforts in [Chapters 4](#) and [5](#) and many state-of-the-art deep learning approaches [[DBBSC21](#), [CRG21b](#)] rely on synthetically-generated training data such as achieved by cutting out objects and inserting them into individual frames of the Cityscapes dataset.

However, these methods fail to leverage, both while generating training data and performing the actual detection, the predictable perspective foreshortening in images captured by vehicles' front-facing cameras. It is a standard practice [[BKOS19](#), [VŠA⁺21](#)] to insert objects of arbitrary sizes at any image location in the training data and to detect objects at multiple-scales irrespective of where they appear in the image. This does not exploit the well-known fact that more distant objects tend to be smaller and that, given a calibrated camera, the relationship between real and projected sizes is known.

In this work, we show that leveraging the perspective information substantially increases performance. To this end, as shown in [Figure 6.1](#), we compute a scale map, whose pixel values denote the apparent size in pixels of a hypothetical meter-wide object placed at that point on the road. We then exploit this information in two complementary ways:

- **Perspective-Aware Synthetic Object Injection.** Instead of uniformly injecting synthetic objects into road scenes to synthesize training data, as in [Chapter 5](#) or [[BKOS19](#), [VŠA⁺21](#)], we use the perspective map to appropriately set the projected size of the objects we insert.
- **Perspective-Aware Architecture.** We feed the perspective map at multiple levels of a feature pyramid network, enabling it to learn the realistic relationship between distance and size embodied in our training set and in real road scenes.

¹© 2023 IEEE. Reprinted, with permission, from
Perspective Aware Road Obstacle Detection
Krzysztof Lis, Sina Honari, Pascal Fua, Mathieu Salzmann
IEEE Robotics and Automation Letters, 2023
<https://doi.org/10.1109/LRA.2023.3245410>

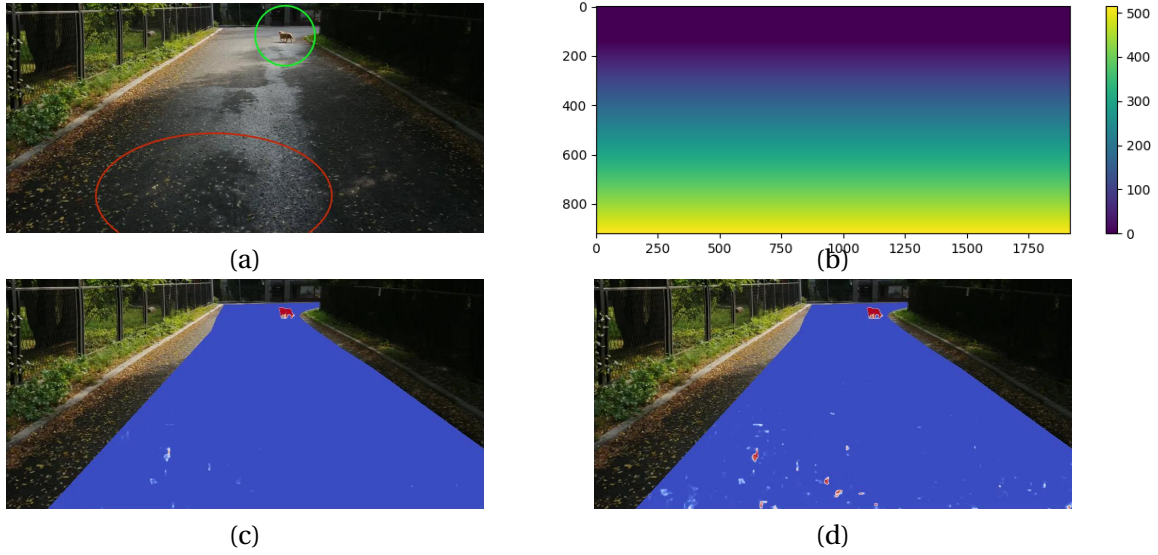


Figure 6.1: **Far and relevant vs close and irrelevant.** (a) Original image. The **green circle** denotes a real obstacle far away, and the **red circle** indicates nearby but harmless leaves. (b) The perspective map indicates, at each pixel, the size in pixels of a hypothetical meter-wide object at that location. (c) Our approach uses the perspective map to distinguish relevant objects from irrelevant ones. It correctly flags in red the pixels of the real obstacle while ignoring the leaves. (d) Without the *perspective aware training set*, a network with a similar architecture flags them all.

The bottom portion of **Figure 6.1** illustrates the benefits of our approach. It not only detects small far-away obstacles but also avoids false alarms arising from small irregularities near the car, such as the leaves here, because their size at this image location does not match that of real threats to the vehicle. Our results show that these strategies together contribute to significantly improving the accuracy of road-obstacle detection, particularly in terms of instance-level detection, which is critical for a self-driving car that need to identify all potential hazards on the road.

We evaluate our approach on the obstacle track introduced in **Chapter 3** and the *Lost&Found* [PRG⁺16] test subset. We demonstrate that it significantly outperforms state-of-the-art techniques that use architectures similar to ours, but without explicit perspective handling.

6.1 Related Work

We have discussed the monocular road anomaly detection methods in **Chapter 2**. Where we discuss other attempts at exploiting perspective information for diverse tasks.

6.1.1 Exploiting Perspective Information

Earlier works [HSB⁺07, HSB⁺09] propose a lightweight sliding-window classifier of drivable space using a pyramid of input patches whose dimension depends on their distance from the horizon. These patches are then rescaled according to their distance to the camera, ensuring

that the similar obstacles have similar pixels sizes when presented to the classifier, regardless of the effects of perspective in the original image. This application of perspective information to overcome scale variance is effective, but it can not be easily combined with standard CNNs which operate on the whole image rather than individually rescaled patches.

For any perspective camera, distortion depends on image position. A popular approach to enabling a deep network to account for this in its predictions is to provide it with pixel coordinates as input. In [LBH18, UVL18, LH18], this is achieved by treating normalized pixel coordinates as two additional channels. In [CKC20] the pixel coordinates are used to compute an attention map, to exploit the fact that the class distribution correlates with the image height, for example the *sky* class is predominantly at the top of the image. Another way to implicitly account for perspective effects is to introduce extra network branches that process the image at different scales and fuse the results [LJW⁺17, HTLH21]. However, this strategy, as those relying on pixel coordinates, does not explicitly leverage the perspective information available when working with a calibrated camera, as is typically the case in self-driving.

None of obstacle-detection algorithms explicitly accounts for the relationship between projected object size and distance. This can be done by creating *scale maps* that encode the expected size in the world of an image pixel depending on its position. Scale maps have been used for obstacle and anomaly detection [BHL⁺19, PAK19]. In [BHL⁺19], the scale information is used to crop and resize image regions before passing them to a vehicle detection network, which then gets to view the cars at an approximately constant scale. This requires running the detector multiple times on the crops. By contrast, our method processes the whole image at once, and the model learns how to leverage the perspective information to adjust the features. In [PAK19], the scale maps are used to rectify the road surfaces, and obstacles are then detected in the rectified views. Unlike these methods, we exploit perspective maps as input to our network, instead of using them for image pre-processing. This prevents the creation of visual artifacts caused by image warping, which yields higher accuracy, as we will show in experiments.

Scale maps have also been extensively investigated for crowd counting purposes [CLV08, SYXC19, ZLWY15, YLW⁺20, LSF19, LLSF19]. In [CLV08, SYXC19], the models predict perspective information based on observed body and head size. In [ZLWY15], an unsupervised meta-learning method is deployed to learn perspective maps, which are then used to warp the input images so that they depict a uniform scale as in [YLW⁺20]. In [LSF19], a scale map serves as an extra channel alongside the RGB image and is passed through the backbone feature extractor, whereas in [LLSF19] an additional branch is added to the backbone to process the single-channel scale map and to concatenate the resulting features afterwards. In short, perspective information is used during feature computation. In this paper, we follow a different track and incorporate the scale map at different levels of a feature pyramid network. Our experiments show this to be more effective. Furthermore, we argue and demonstrate that, for anomaly detection, incorporating perspective information into the network is not enough; one must also exploit it when synthesizing training data.

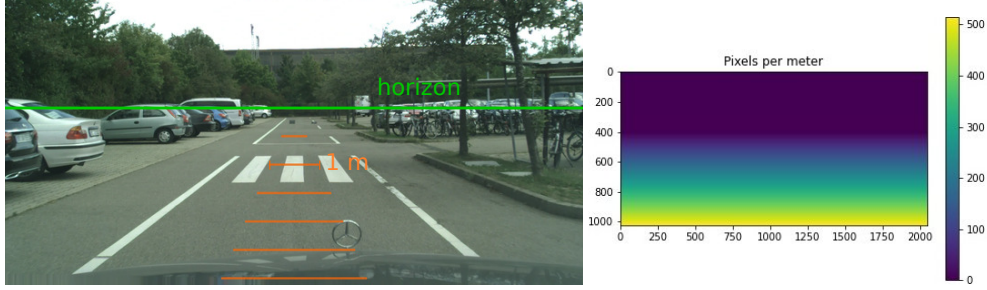


Figure 6.2: **Perspective map.** *Left:* A 1-meter length overlaid at different image heights on an image from the *Lost&Found* dataset. *Right:* The corresponding perspective map.

6.1.2 Fusing RGB and Depth for Road-Obstacle Detection

When depth information is available, from stereo camera disparity or RGB-D sensors, it can be fused with the RGB appearance to improve obstacle detection. For example, [RGP⁺17] combines semantic segmentation with stereo-based detections; MergeNet [GJK18] extracts complementary features from RGB-D; RFNet [SYH⁺20]’s two-stream backbone extracts RGB and depth features and uses them to output joint segmentation of known classes and unusual obstacles. Depth (or disparity) contains important geometric cues about the obstacles, which protrude from the road plane, and the above-mentioned methods exploit these cues to detect the obstacles. By contrast, we do not use stereo images and the associated precise scene geometry; our perspective map is generated using a flat-road assumption and contains no information about the obstacles. Our architecture uses perspective as context for analyzing obstacle appearance, by taking it as an extra feature channel without any processing.

6.2 Approach

Our approach relies on a *perspective map* that captures scale change of objects on the road plane. Therefore, we also refer to it as a *scale map*. In this section, we first describe its construction and then how we use it both to control training data synthesis and as an input to our detection network.

6.2.1 Computing the Perspective Map

A perspective map is a scalar field whose value at a given pixel denotes the width in pixels of a hypothetical meter-wide object placed at that point on the road. Figure 6.2 depicts one. We compute it from the camera calibration parameters, which are known in a self-driving setup because a vehicle’s camera can be calibrated during its production. The camera parameters are f , the camera’s focal length in pixels, H , its elevation above the ground in meters, and θ , its pitch angle.

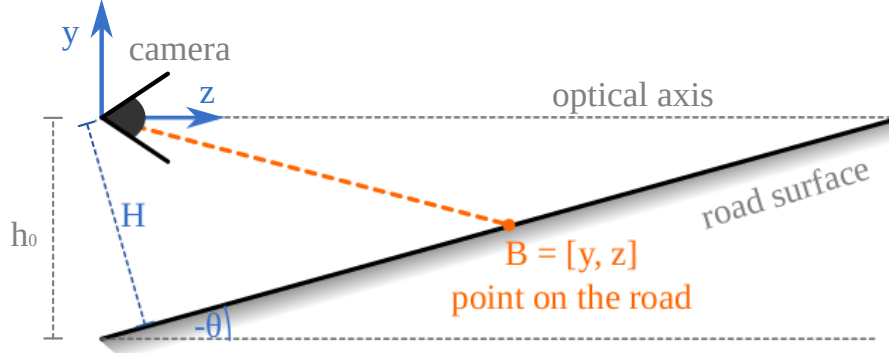


Figure 6.3: **Building the perspective map.** Geometry of a front-facing camera viewing a planar road surface. Here, the y coordinate of the orange point is negative because it is below the optical axis.

We assume the road to be planar, which, locally, is a good approximation in the majority of real driving scenarios. Let us consider a 3D road point $B = [x, y, z]$ in camera coordinates, which projects onto $[u, v]$ in the image space, as shown in Figure 6.3. For simplicity, we denote by $[0, 0]$ the principal point that lies at the center of the image, which is also known in a calibrated camera. Assuming the road to be planar, the pinhole camera model dictates that

$$v = f \frac{y}{z}. \quad (6.1)$$

As B lies on the road plane, which is inclined by an angle θ w.r.t. the camera's optical axis, we can write

$$y = z \tan(\theta) - h_0, \quad (6.2)$$

where $h_0 = \frac{H}{\cos \theta}$, with H being the perpendicular distance from the camera to the road. Solving for z by replacing y in Eq. 6.1 by its definition in Eq. 6.2 yields

$$z([u, v]) = h_0 \frac{f}{f \tan(\theta) - v}, \quad (6.3)$$

where we indicate that z depends on the pixel location $[u, v]$ by $z([u, v])$. The visible scale is inversely proportional to the z coordinate in the camera frame, and so the scale value $P([u, v])$ in the perspective map \mathbf{P} is equal to

$$P([u, v]) = f \frac{1}{z([u, v])} = \frac{\cos(\theta)}{H} (f \tan(\theta) - v). \quad (6.4)$$

Note that this requires the pitch angle θ of the camera optical axis with respect to the road surface to be known. When the car is stable on its four wheels, it only depends on how the camera is mounted, which is known. If the pitch changes while driving, an online camera calibration module, e.g., one that relies on vanishing points [CDI14], could be used to update the value of θ . We also assume that various distortions have been corrected so that a pinhole camera model applies.



Figure 6.4: *Left*: Anchor points distributed along the road surface. We place obstacles at a random subset of anchor points. *Right*: Frame with injected obstacles.

6.2.2 Perspective-Aware Synthetic Object Injection

Collecting a training database of all items that could potentially be left on the road and pose a collision threat is impractical. Effective obstacle detection can thus only be achieved via handling previously unseen objects. To this end, our approach from [Chapter 5](#) or existing methods [[VŠA⁺21](#), [BKOS19](#)] generate synthetic training frames by injecting objects into the road scenes. However, they use random object sizes and locations. Instead, we leverage the perspective map so that the inserted object sizes are consistent with their locations on the road plane.

Placement

We generate a rectangular grid on the road plane, with grid lines being 3.5 meters apart in the direction along the road and 1 meter apart in the width direction. Once this grid is projected onto the image, each grid intersection yields an anchor point offset from the grid point by a random vector whose coordinates are drawn from a zero-centered normal distribution with standard deviation of 0.5 meter. The anchor points are shown in [Figure 6.4](#). We then place obstacles at a random subset of these anchor points.

Size

We extract object instances – vehicles, pedestrians, traffic signs and lights – from the Cityscapes dataset [[COR⁺16](#)]. These yield image cut-outs of diverse shapes, ranging from thin poles to wide vehicles. We take an object’s *overall pixel size* pix_{obj} to be the average of three values: the square root of its pixel area, and its bounding box width and height. We aim to generate synthetic objects within a range of physical sizes $[\text{ph}_{\min}, \text{ph}_{\max}]$ in meters. To simulate the corresponding visible pixel size of an object seen at an image point $[u, v]$, we multiply the physical range by the scale map value at $[u, v]$: $\text{pix}_{(\min, \max)}([u, v]) = \text{ph}_{(\min, \max)} P([u, v])$. Then, we randomly select an object from the training set whose size satisfies $\text{pix}_{\min}([u, v]) \leq \text{pix}_{\text{obj}} \leq \text{pix}_{\max}([u, v])$ and paste it at $[u, v]$. Since we use known classes, such as humans or cars, to later detect unknown classes, such as bottles or tires, we ignore the original size of the known items, and instead choose the object size as a hyper-parameter using the validation-set.

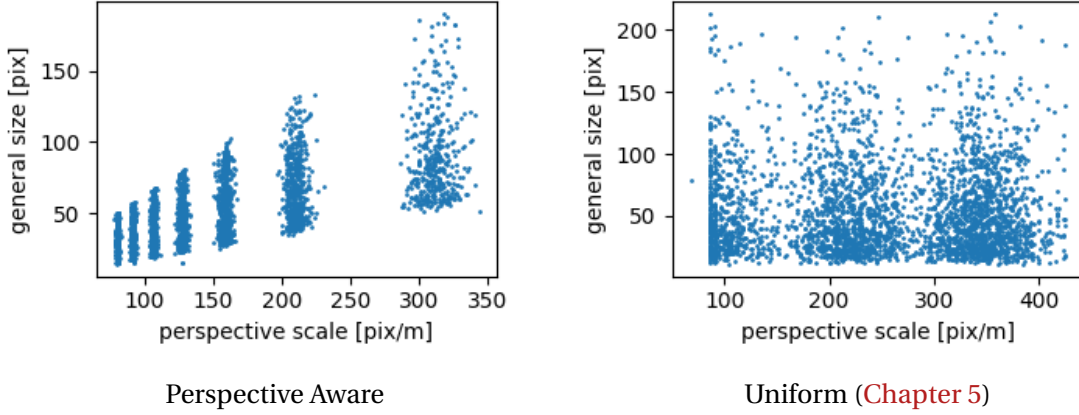


Figure 6.5: Distribution of injected object sizes. *Left*: Our **Perspective Aware Strategy** selects object sizes based on the perspective map to ensure that objects look smaller when they are further away. There are clusters because we inject objects at discrete grid points on the road-plane projected to the image. *Right*: The **Uniform Strategy** chooses random objects from the whole instance database.

Figure 6.4-right depicts the resulting object insertion. Note that this does *not* involve scaling the original cut-out objects. Instead, we simply select objects of the appropriate size, thus avoiding scaling artifacts. In Figure 6.5, we visualize the resulting relationship between object size and perspective map for both our perspective-aware approach and a uniform injection one. We will compare these two approaches quantitatively.

6.2.3 Perspective-Aware Architecture

To distinguish obstacle pixels from the road surface ones, we rely on a U-Net type network architecture, which we train using negative binary cross-entropy loss of pixel classification between the model’s prediction and the ground truth segmentation map. The input image is first processed by a ResNeXt101 [XGD⁺17] feature extractor, pre-trained on ImageNet [DDS⁺09] and frozen at training time. We extract four levels of features with increasing receptive fields.

In each block of our up-convolution pyramid, we concatenate the perspective map to the backbone features, and we use it again before the transposed convolution, as depicted in Figure 6.6. With such an insertion, the scale information presented to each level of the pyramid can then influence the interpretation of the backbone features at different receptive fields and hence locally adjust the effective receptive field of the detector. In practice, this allows the network to distinguish between distant obstacles and small but harmless irregularities such as wet patches, leaves, or tile edges, which ought to be ignored. As evidenced below, our perspective-aware architecture shows its full advantage when used together with the perspective-aware object injection.

In our experiments, the perspective map is scaled by $\frac{1}{400}$ before being passed to the network, following the normalization applied in [LLSF19], to bring it to an approximate value range between 0 and 1, which improves convergence.

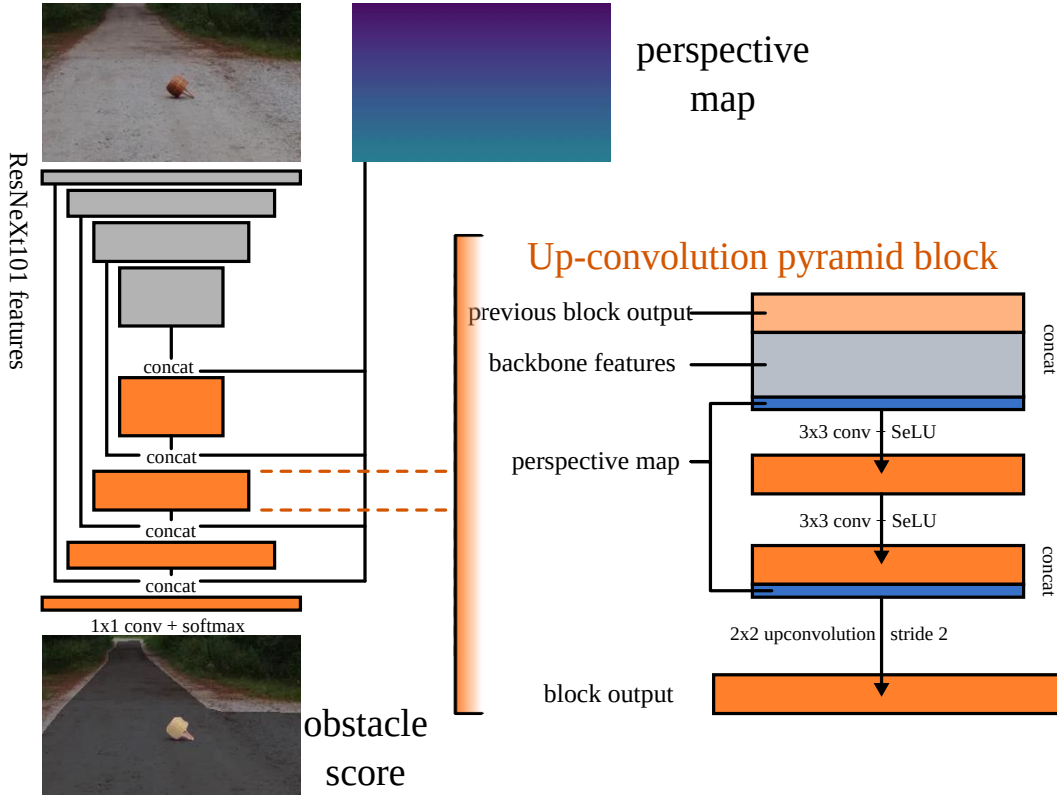


Figure 6.6: Perspective-aware architecture. The perspective map is injected into the decoding blocks at different resolutions. In each block it is appended twice; first to the backbone features, and second to the intermediate activations preceding the transpose convolution for upsampling.

6.3 Experiments

In this section, we present the implementation details, datasets, metrics, and compare our method with the state-of-the-art ones.

6.3.1 Datasets

We train our network on Cityscapes. During training, we sample patches of 768×384 pixels, with random horizontal flipping. We also perform noise augmentation so that the network generalizes to road surfaces rougher than those found in Cityscapes. We follow the evaluation protocol introduced in [Chapter 3](#) and test our network using the following two datasets.

Lost & Found - Test No Known

Lost & Found [\[PRG⁺16\]](#) is an established obstacle dataset captured by placing objects on the road and taking images from an approaching vehicle. The *No Known* variant excludes objects present in the Cityscapes training set, such as pedestrians or bicycles, to focus on the methods' ability to generalize to previously unseen obstacles. It contains 1043 frames

6.3 Experiments

Method	requires OOD	Road Obstacles 21				Lost&Found - Test No Known			
		Component-level			Pixel-level AuPRC \uparrow	Component-level			Pixel-level AuPRC \uparrow
		$\overline{F_1} \uparrow$	sIoU \uparrow	PPV \uparrow		$\overline{F_1} \uparrow$	sIoU \uparrow	PPV \uparrow	
Ours	No	67.1 \pm 1.7	65.2 \pm 0.6	60.2 \pm 2.7	75.2 \pm 0.1	68.6 \pm 0.4	49.8 \pm 0.6	87.6 \pm 1.2	87.4 \pm 0.4
DenseHybrid [GBS22]	Yes	50.7	45.7	50.1	87.1	52.3	46.9	52.1	78.7
Maximized Entropy [CRG21b]	Yes	48.5	47.9	62.6	85.1	49.9	45.9	63.1	77.9
SynBoost [DBBSC21]	Yes	37.6	44.3	41.8	71.3	48.7	36.8	72.3	81.7
Road Inpainting (Chapter 5)	No	36.0	57.6	39.5	54.1	52.3	49.2	60.7	82.9
JSRNet [vSA*21]	No	11.0	18.6	24.5	28.1	36.0	34.3	45.9	74.2
ODIN [LLS18]	No	9.4	21.6	18.5	22.1	34.5	39.8	49.3	52.9
Image Resynthesis (Chapter 4)	No	8.4	16.6	20.5	37.7	19.2	27.2	30.7	57.1
Maximum Softmax [HG17]	No	6.3	19.7	15.9	15.7	10.3	14.2	62.2	30.1
Void Classifier [BSN*19]	Yes	5.4	6.3	20.3	10.4	1.9	1.8	35.1	4.8
Mahalanobis [LLLS18b]	No	4.7	13.5	21.8	20.9	22.1	33.8	31.7	55.0
Embedding Density [BSN*19]	No	2.3	35.6	2.9	0.8	27.5	37.8	35.2	61.7
Ensemble [LPB17]	No	1.3	8.6	4.7	1.1	2.7	6.7	7.6	2.9
MC Dropout [MG18b]	No	1.0	5.5	5.8	4.9	13.0	17.4	34.7	36.8

Table 6.1: Obstacle detection scores on RoadObstacle21 and Lost&Found datasets. Both component-level and pixel-level metrics are reported on each dataset. The primary metric is average component detection $\overline{F_1}$ score. *Requires OOD* column indicates if a model is using out of distribution data by training on additional datasets. Other methods train only on Cityscapes dataset.

Architecture	Object Injection	Road Obstacles 21		Lost&Found - Test No Known	
		$\overline{F_1} \uparrow$	AuPRC \uparrow	$\overline{F_1} \uparrow$	AuPRC \uparrow
1 Ours (perspective-aware)	Ours (perspective-aware)	67.1 \pm 1.7	75.2 \pm 0.1	68.6 \pm 0.4	87.4 \pm 0.4
2 No perspective channel	Ours (perspective-aware)	52.5 \pm 6.7	70.6 \pm 1.0	63.5 \pm 3.9	86.0 \pm 0.8
3 P-map backbone branch [LLSF19]	Ours (perspective-aware)	65.0 \pm 2.2	74.6 \pm 1.2	63.5 \pm 2.4	85.5 \pm 0.7
4 P-map along RGB [LSF19]	Ours (perspective-aware)	58.2 \pm 4.1	64.6 \pm 2.8	66.2 \pm 3.5	73.9 \pm 8.3
5 XY channels	Ours (perspective-aware)	54.8 \pm 4.2	71.1 \pm 2.3	63.8 \pm 0.8	86.1 \pm 0.1
6 Image warping [PAK19, YLW*20]	Ours (perspective-aware)	45.2 \pm 0.5	65.5 \pm 0.7	20.3 \pm 0.6	43.5 \pm 1.3
7 Ours (perspective-aware)	Uniform (Chapter 5)	56.1 \pm 1.7	77.1 \pm 0.9	52.4 \pm 2.6	82.1 \pm 3.3
8 No perspective channel	Uniform (Chapter 5)	43.7 \pm 6.6	73.3 \pm 1.7	48.5 \pm 6.4	74.5 \pm 10.2
9 P-map backbone branch [LLSF19]	Uniform (Chapter 5)	53.3 \pm 4.3	76.8 \pm 0.5	55.2 \pm 1.5	84.3 \pm 0.2
10 P-map along RGB [LSF19]	Uniform (Chapter 5)	50.8 \pm 1.6	69.0 \pm 2.4	50.6 \pm 5.8	79.7 \pm 1.5
11 XY channels	Uniform (Chapter 5)	51.9 \pm 1.4	75.7 \pm 0.5	53.7 \pm 2.1	78.7 \pm 1.7

Table 6.2: Ablation study. We compare different variants of utilizing the perspective map and show their impact while using either uniform or our perspective-aware object injection.

with 1709 occurrences of 7 unique lost cargo items placed in 12 parking lot and street scenes. The camera calibration parameters required to compute the perspective map are part of the dataset.

RoadObstacle21

Like Lost & Found, it contains photos of obstacles placed on roads, but it expands the number of unique objects and the diversity of the scenes to include more road textures and weather conditions. As previously discussed in Chapter 3, it comprises 327 frames containing 388 occurrences of 31 unique objects placed in 8 scenes. There are no camera calibration parameters. We therefore estimated them as follows. We assume $f = 2265\text{pix}$, that is, the same focal length as in the Cityscapes training set, and $H = 1.5\text{m}$ because the dataset was captured using handheld cameras. We estimate the camera’s pitch angle by approximating the horizon level - the image-space position of the road plane’s vanishing line. In the considered datasets, the camera has no side-to-side roll, so we assume the line to be horizontal. The sides of the road and not regular enough to fit lines to them, but with the images depicting forward views along the roads, the horizon is slightly above the end of the visible road. Hence, we

Chapter 6. Perspective Aware Road Obstacle Detection

Object size [m]		Road Obstacles 21 - Validation		Lost&Found - Train	
ph_{\min}	ph_{\max}	$\overline{F}_1 \uparrow$	AuPRC \uparrow	$\overline{F}_1 \uparrow$	AuPRC \uparrow
0.1	- 0.3	59.3 ± 3.4	80.2 ± 1.6	66.1 ± 1.1	77.5 ± 0.5
0.25	- 0.55	65.1 ± 3.0	95.7 ± 0.7	62.5 ± 0.3	89.4 ± 0.8
0.5	- 0.9	65.6 ± 1.4	96.6 ± 0.4	57.5 ± 0.1	87.6 ± 0.4
0.75	- 1.25	49.5 ± 1.7	94.2 ± 0.1	50.3 ± 2.4	86.8 ± 1.4
all	sizes (Chapter 5)	56.1 ± 1.7	77.1 ± 0.9	52.4 ± 2.6	82.1 ± 3.3

Table 6.3: Effect of the size of the injected training objects.

first segmented the road using the semantic segmentation PSP network [ZSQ⁺17], and then took the approximate horizon level to be 16 pixels above its uppermost edge. Given ν_{horiz} , the number of pixels between the image midpoint and the horizon level, the pitch angle is retrieved as

$$\theta = \tan^{-1}\left(\frac{\nu_{\text{horiz}}}{f}\right).$$

Such an estimate is obviously very rough, but it is sufficient to inform our model of the scale changes on the road, as we will empirically show when comparing to other variants of our model that do not leverage perspective information.

6.3.2 Metrics

Our evaluation protocol introduced in Chapter 3 measures the methods' performance at both pixel and component levels. The pixel classification task involves distinguishing pixels belonging to obstacles from those of the road surface. It is primarily evaluated with the area under the precision-recall curve (AuPRC). However, pixel metrics give more importance to nearby and big obstacles than to distant or small ones, because of the image area they occupy. For the purpose of driving safety, it thus is more relevant to reason in terms of obstacle instances and their detection regardless of distance and image size. This is addressed by component-level metrics such as \overline{F}_1 , \overline{sIoU} and \overline{PPV} .

6.3.3 Quantitative Evaluation

In Table 6.1, we compare our approach to the state-of-the-art methods featured in the *Segment Me* benchmark. The *requires OOD (out of distribution)* column indicates whether the method was trained using additional data beyond the commonly-used Cityscapes training set, for example by using objects from COCO [LMB⁺14] as obstacles. Our method outperforms the baselines in terms of instance metrics, and only performs worse than [CRG21b] and [GBŠ22] on two metrics, which leverages OOD, thereby demonstrating the good generalization of our approach without resorting to extra training data.

6.3.4 Ablation study

Impact of Perspective

In Table 6.2, we report the results of an ablation study in which we altered either our architecture to use the perspective map in different ways or the perspective-aware synthetic object insertion strategy.

In particular, we consider the following variants:

- *Ours (perspective-aware)*: Our full architecture using the perspective map as described in Section 6.2.3.
- *No perspective channel*: This variant omits the perspective map from our complete architecture.
- *P-map backbone branch*: We provide the perspective map as input to the network but process it in a feature extractor separately from the RGB feature extractor, as in the architecture of [LLSF19].
- *P-map along RGB*: We provide the perspective map along with RGB as input to the network, following [LSF19]. In this variant, we unfreeze the backbone feature extractor weights during training to let the network train on the perspective inputs.
- *XY channels*: We provide the image coordinates as two additional channels instead of the perspective map, applying the idea from [LBH18, CKC20].
- *Image warping*: We follow the idea of [PAK19, YLW⁺20] and use the perspective map to transform the image into a top-down view of the road.

For synthetic object insertion, we consider two variants; one with perspective-aware object injection as described in Section 6.2.2, and another with uniform object insertion that injects objects uniformly from the full object pool, without restricting it to the objects whose size are inversely proportional to the location where they will be placed. In this variant, the obstacles are placed uniformly in image space rather than on a grid in the road plane. This strategy is identical to the one used in Chapter 5.

In each setup we provide the mean and standard deviations over three training runs. As observed in Table 6.2, using our perspective-aware objection injection together with our perspective-aware network architecture yields the best performance (row 1), and dropping each one of them reduces the accuracy (rows 2, 7, 8). It is worth noting that using the *P-map backbone branch* architecture with our perspective-aware object insertion (row 3) also yields close, yet still inferior results, which indicates that, while there can be other ways of using the perspective map in the architecture, the perspective-aware object insertion plays a key role. The *No perspective channel* variant shows a noticeable drop (row 7), indicating

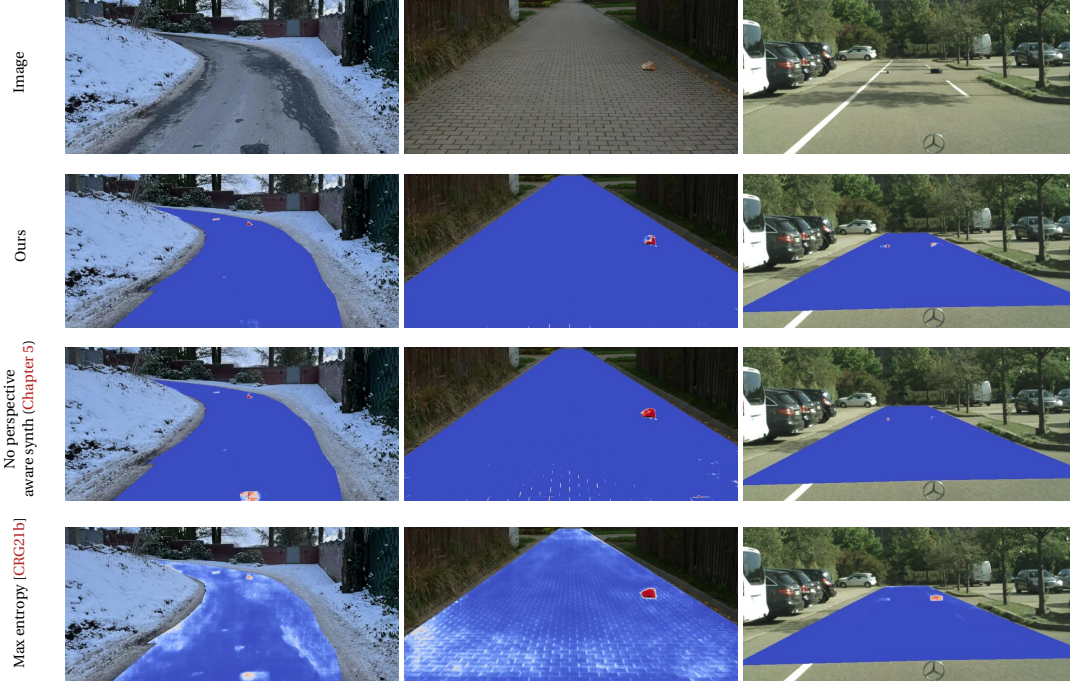


Figure 6.7: *Left, center*: Perspective information guides our detector to ignore nearby small irregularities on the road surface, while the variants without perspective map and perspective-aware object insertion exhibit false positives in that area. The nearby false-positives and distant obstacles are of similar pixel sizes, so the perspective map allows differentiating between them. *Right*: Our method finds both obstacle instances despite imperfect segmentation. While Max entropy [CRG21b] achieves a better pixel-classification score by perfectly segmenting the bigger object, it misses the smaller object.

that the network benefits from exploiting the perspective maps. While using the XY image coordinates (row 5) yields reasonable results in *Lost&Found*, whose camera angle matches that of the training set, its performance drops when faced with the different camera setup of *Road Obstacles*. The *P-map along RGB* and *Image warping* variants also show a significant drop (rows 4, 6), which indicates that the way the perspective map is used plays a role in the network accuracy. In particular, the performance of *Image warping*, where the network operates on the warped image, is much lower.

The results show that training with a uniform injection strategy (rows 7-11) yields a much lower \overline{F}_1 score than with our perspective-aware approach (rows 1-5). However, the pixel classification AuPRC values of the uniform strategy tend to be higher. We observe that the networks trained with the uniform object insertion technique often better segment the large objects (predicting more pixels on the object), but miss small objects and introduce small false positive instances. By contrast, the higher \overline{F}_1 scores obtained with our perspective-aware injection approach evidence the resulting networks' reliability in detecting obstacles more accurately and avoiding false-positives, which is more critical for self-driving cars.

We show qualitative examples in Figure 6.7. The first two columns evidence how the perspective map helps the model to distinguish distant obstacles from nearby harmless details, which can span similar pixel sizes in the image. In the rightmost column, we show a difficult case

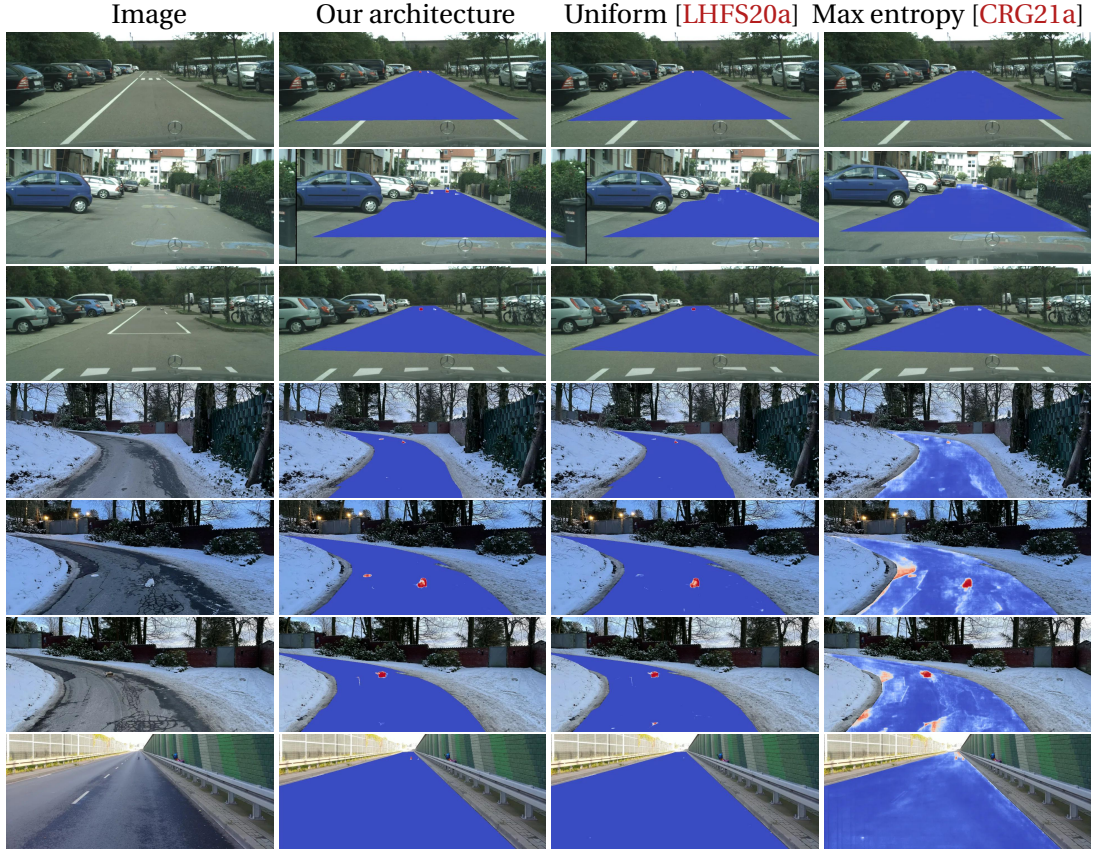


Figure 6.8: Comparison of results: our method is capable of finding obstacles missed by other variants.

where Max-entropy [CRG21b] obtains higher AuPRC but lower $\overline{F_1}$ than us. While such models can segment more pixels of the found objects, they entirely miss some of the objects on the road. In Figure 6.8 we show examples where our network is capable of finding small distant obstacles missed by other variants. In Figure 6.9 we depict our method using perspective information to avoid false-positives near the ego-vehicle. Overall, our network detects the obstacles more reliably, and learns to ignore small irregular regions.

This effect is quantified in Figure 6.10 where we plot the number of false-positives and true-positives as a function of the distance to the camera, estimated as the inverse of the perspective map. Using our training set and architecture strongly decreases the number of nearby false-positives compared to a system without those contributions, and slightly increases the number of correctly detected obstacles.

Object size

In Table 6.3, we show how the chosen range $\text{Obj}_{[\min, \max]}$ for synthetically injected objects affects detection performance. To avoid overfitting to the benchmarks, we performed this study using public validation sets, *Road Obstacles 21 - Validation* and *Lost & Found - Train*, featuring different obstacles and scenes than the test sets.

Chapter 6. Perspective Aware Road Obstacle Detection

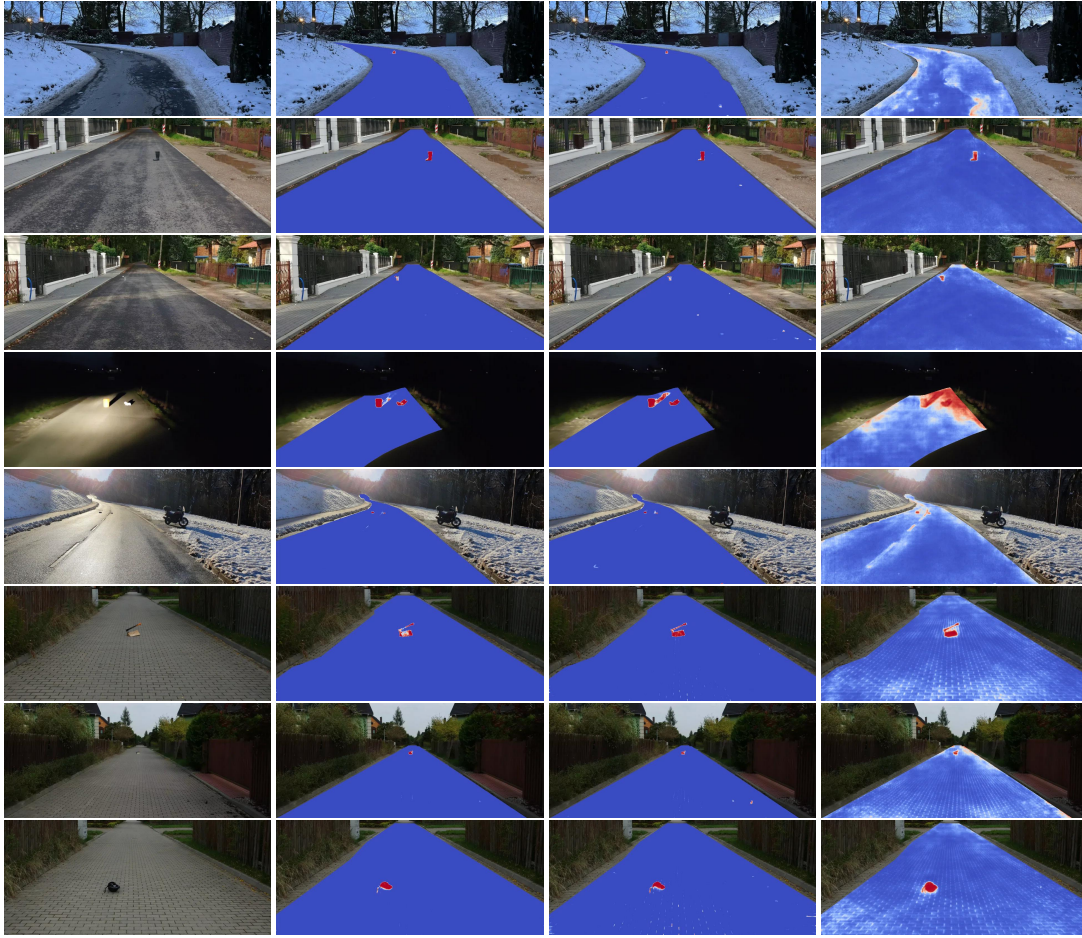


Figure 6.9: Comparison of results: our method avoids false-positives near the ego-vehicle thanks to perspective information.

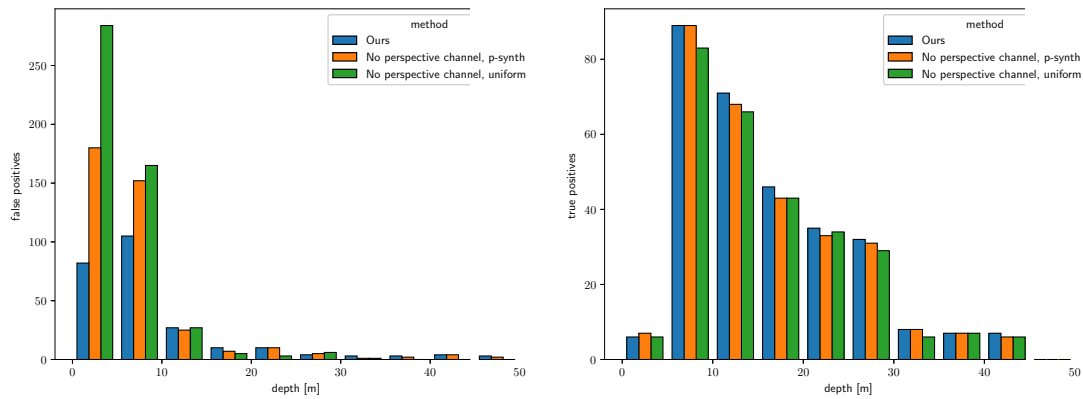
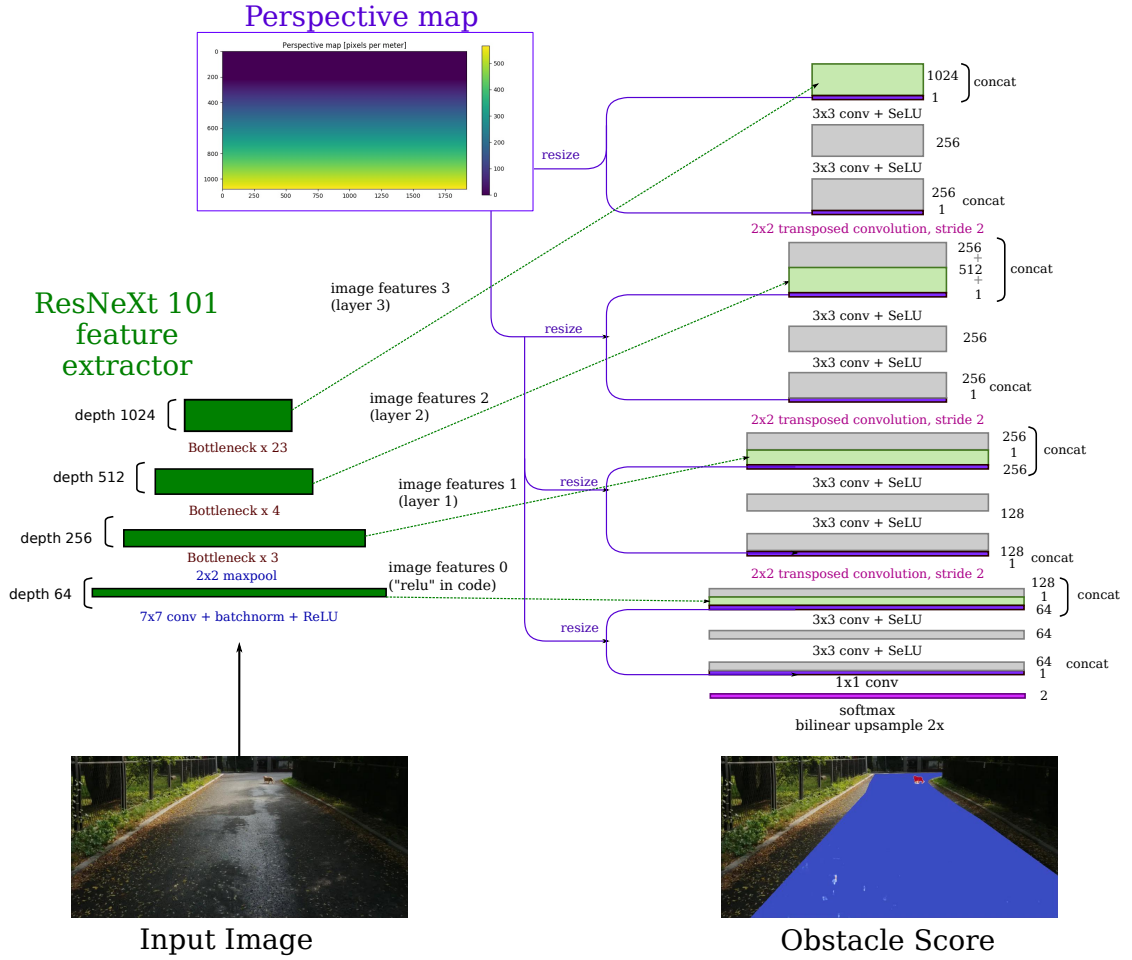


Figure 6.10: Number of false positives (*left*) and true positives (*right*) as a function of the distance from the camera for the *Obstacle Track - test* dataset. Our training set and architecture (Ours) yield much fewer nearby false-positives and slightly more true-positives than a variant without the perspective map (No perspective channel, p-synth) or one trained with the uniformly-injected synthetic obstacles (No perspective channel, uniform). FP and TP are calculated for an IOU threshold of 0.5.

Figure 6.11: **Detector network architecture.**

As described in Section 6.2.2, the minimum and maximum sizes in meters are multiplied by the local perspective-map value at the site of injection, to determine how big the injected object can be in pixels. We then select at random an object fitting within this pixel range. The results in the table indicate that there is no size range that ideally fits all the circumstances; the small 0.1-0.3 meter range is best for the *Lost & Found - Train* set, while the 0.5-0.9 m range prevails in *Road Obstacles 21 - Validation*, presumably due to these ranges matching the typical object sizes in those datasets. We choose the intermediate 0.25-0.55 meter range, behaves well in both datasets. One might conclude that including objects of all sizes would be best for generalization, but that would prevent expressing the perspective-size relationship, as shown in the bottom row. Indeed, such a strategy, used in Chapter 5, yields lower performance than our narrower ranges.

6.3.5 Implementation details

In this section, we present details on the detector network architecture and training.

Chapter 6. Perspective Aware Road Obstacle Detection

Detector Network Architecture

The architecture is shown in [Figure 6.11](#). We use the pretrained ResNeXt101 network of [\[XGD⁺17\]](#) as backbone feature extractor. We take the outputs of `relu`, `layer1`, `layer2`, `layer3` layers of the PyTorch implementation [\[PGM⁺19\]](#) of ResNeXt.

Training process

The training is done using 768×384 crops of the road area. We add noise to the images to adapt to rough road surfaces not present in Cityscapes, the noise augmentation follows [\[LHFS20a\]](#).

We perform 50 epochs of training, each iterating over all the training frames. For the synthetic dataset we generate one frame per source Cityscapes background, resulting in 2975 training and 500 validation frames. We pre-define the crops and epoch ordering, so that all variants are trained with the same sequence of samples.

We use a binary cross entropy loss penalizing the difference in pixel classification between the prediction and ground-truth. We use the Adam [\[KB15\]](#) optimizer with an initial learning rate to 10^{-4} . If the validation loss does not decrease for 5 consecutive epochs, the learning rate is reduced 10 times.

Inference Speed

Our network achieves inference at 12.1 frames of size 1920×1080 per second on an Nvidia V100 GPU; it can be further sped up by network distillation, quantization or TensorRT.

6.4 Conclusion

We have shown that perspective-aware obstacle injection to generate training data, together with incorporating perspective information in the decoding stages of a network outperforms the state-of-the-art road-obstacle detection methods. Our results indicate that the perspective information can guide the model to reduce false positives for small nearby irregularities while still detecting small and far-away objects.

7 Attention Entropy: Generalizing to New Obstacles in New Domains

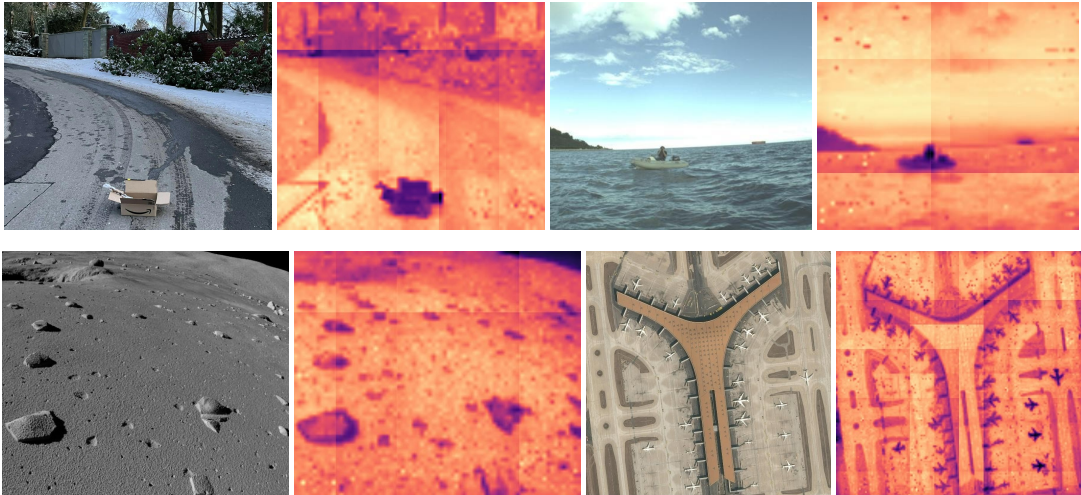


Figure 7.1: The attention-entropy extracted from a SETR [ZLZ⁺21] transformer trained on the urban driving Cityscapes [COR⁺16] dataset lets us segment novel objects in novel scenes: Unusual road obstacles (Chapter 3), maritime hazards [BMPK19], aircraft [FA20], and lunar rocks [PJI19].

In the previous chapters we have utilized specialized synthetic training to allow our detectors to learn to find road anomalies and obstacles without seeing them previously. Here we try to find road obstacles without extra training using only the semantic segmentation network trained on Cityscapes. We find success with visual transformers and discover their impressive capacity to generalize to completely new objects and domains.

In the past few years, vision transformers have become increasingly prominent in the community. In particular, they nowadays tend to outperform their convolutional counterparts. Initially, such vision transformer models were developed for image recognition. However, they now tackle many other tasks such as semantic segmentation [ZLZ⁺21, XWY⁺21, SGLS21, CSK21], monocular depth estimation [CZX21, ZZP⁺22].

Transformers have demonstrated valuable properties when studied more closely. For example, transformers trained for image recognition can be employed to segment the main object

in the input image [SGLS21]. Similarly, several studies have evidenced that self-supervised vision transformers can segment the foreground objects using the attention of the class-token [CTM⁺21] or by partitioning a graph of inter-patch affinity [SPV⁺21, WSY⁺22]. Altogether, these works highlight an interesting phenomenon: Models that were *not* trained for segmentation encode information relevant to this task.

In this chapter, by contrast, we study the behavior of vision transformers that were *explicitly* trained for semantic segmentation in a *supervised* fashion. In this context, one might expect that, because they were trained to segment a specific set of categories, typically quite small for most segmentation datasets, these models have learned to discard information about other categories, essentially encompassing them in the background class. Our analyses show that this is not the case; vision transformers trained to segment a small set of categories contain information that can be leveraged to segment new object classes that were not seen during training. Furthermore, we show that this behavior generalizes not only to new objects but also to new domains; for example, as shown in Figure 7.1, a vision transformer trained to segment traffic scenes contains information that can be exploited to segment boats in maritime scenes, aircraft seen from aerial images, or even moon rocks in lunar landscapes.

To achieve this, we study in detail the intermediate spatial attention maps of segmentation transformers. We then extract the Shannon entropy of spatial attentions and show that it can be used for accurate segmentation of unknown objects in unknown contexts. Specifically, the spatial concentration of attention is well suited to finding multiple small to moderate-size objects, which is valuable in tasks such as obstacle avoidance and not addressed by approaches designed for large foreground items [CTM⁺21, SPV⁺21, WSY⁺22]. We demonstrate the effectiveness of our method quantitatively on our benchmark from Chapter 3, using a model trained on the known categories of the Cityscapes dataset. We further show that the same model, without any further training, can be leveraged to accurately segment unknown objects in maritime scenes [BMPK19], bird’s eye views of airports [FA20], and images of lunar scenes [PJI19].

Our contributions can be summarized as follows:

- We study the behavior of vision transformers trained in a supervised fashion for semantic segmentation when faced with new objects and new domains.
- We extract the entropy of spatial attentions and show that it can be used to segment unknown objects of small to moderate size in unknown context.
- We evidence the robustness of our approach by applying it to datasets with varying degrees of domain shifts.
- We demonstrate the generality of our analysis using two semantic segmentation transformers [ZLZ⁺21, XWY⁺21].

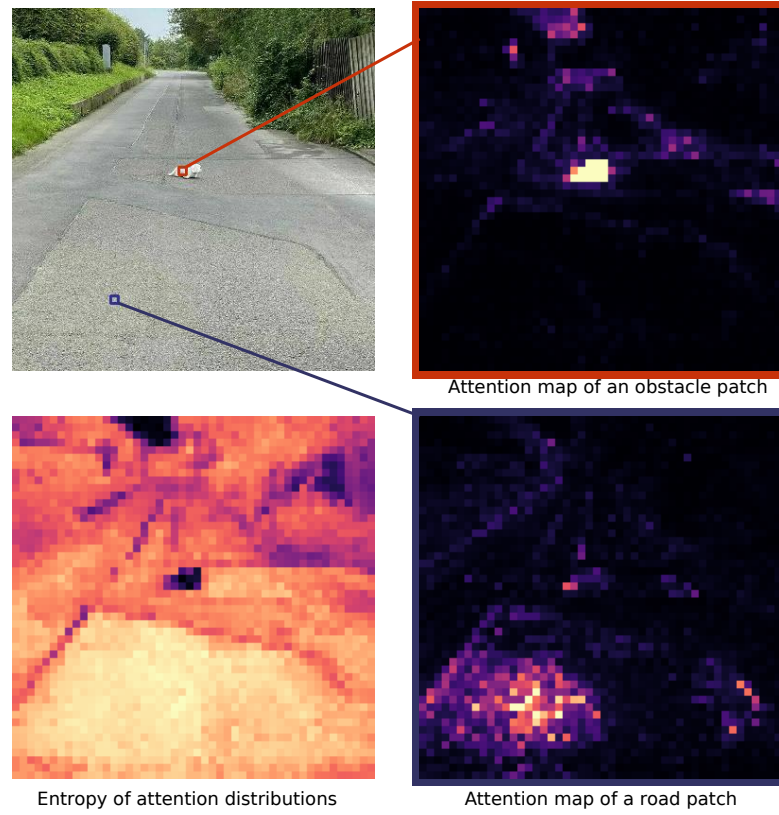


Figure 7.2: **Entropy heatmaps.** For each image patch, we compute the Shannon entropy of outgoing attentions. The two images in the right column depict the spatial attention at two different image locations. The bottom left images shows the Shannon entropy. Small objects receive concentrated attention and thus low corresponding entropy. An interactive tool for attention visualization is included in the supplementary material.

7.1 From Attention to Segmentation

We first remind the reader of the attention mechanism that is at the heart of all transformer architectures. We then show how to use the attention matrices to create entropy heatmaps that highlight small to moderate-size objects. Finally, we discuss our implementation.

7.1.1 Attention Mechanism in Vision Transformers

The Vision Transformer (ViT) [DBK⁺20] is one of the first successful applications of the transformer self-attention mechanism to image inputs. With enough training data, it can be more powerful than a traditional convolutional network for image classification. Furthermore, it serves as a backbone feature extractor for the Segmentation Transformer (SETR) [ZLZ⁺21], providing improvement in semantic segmentation of road scenes over comparable convolutional backbones. We base the following description on the popular ViT architecture, but will show at the end of the section how it can be implemented for the differently structured Segformer [XWY⁺21].

Chapter 7. Attention Entropy: Generalizing to New Obstacles in New Domains

ViT first decomposes the image in a checkerboard fashion into $N \times N$ image patches, each of size 16×16 pixels, calculates their initial encodings and adds a learned spatial embedding, which will allow nearby patches to attend strongly to each other. The patches serve as tokens in the transformer: The attention mechanism connects all the patches to each other.

The backbone consists of L multi-head self-attention (MSA) blocks. Each attention block, or layer, l receives as input a triplet (query Q , key K , value V) computed from an input feature map $Z^{l-1} \in \mathbb{R}^{N^2 \times C}$, where C denotes the number of feature channels. The triplet is computed from Z^{l-1} as

$$Q = Z^{l-1} W_Q, \quad K = Z^{l-1} W_K, \quad V = Z^{l-1} W_V, \quad (7.1)$$

where $W_Q, W_K, W_V \in \mathbb{R}^{C \times d}$ are learnable weight matrices and $d \in \mathbb{N}$ a parameter which we specify below. The attention A^l , which is the matrix that concerns us most in this paper, is taken to be

$$A^l(Z^{l-1}) = \text{softmax}(Q \cdot K^T / \sqrt{d}). \quad (7.2)$$

The softmax acts row-wise, so that the outgoing attention of each token sums up to 1. In the multi-head scenario, the attention is in fact computed multiple times. This yields m independent attention matrices A_i^l , $i = 1, \dots, m$. While we will later focus on the attention matrices, for the sake of completeness, we derive the corresponding self-attention SA. It is expressed as

$$SA^l(Z^{l-1}) = A^l(Z^{l-1}) \cdot V. \quad (7.3)$$

Again, accounting for the multiple heads, this results in m independent self-attention matrices SA_i^l , $i = 1, \dots, m$, that are concatenated

$$SA_{1:m}^l(Z^{l-1}) = [SA_1^l(Z^{l-1}), \dots, SA_m^l(Z^{l-1})] \quad (7.4)$$

and then multiplied with another weight matrix $W_O \in \mathbb{R}^{md \times C}$. Combined with a skip connection, this gives

$$MSA^l(Z^{l-1}) = SA_{1:m}^l(Z^{l-1}) + SA_{1:m}^l(Z^{l-1}) W_O, \quad (7.5)$$

and, as in [ZLZ⁺21], we choose $d = C/m$. Finally, $MSA^l(Z^{l-1})$ is again input to a multilayer perceptron (MLP) block with a skip connection. This yields

$$Z^l = MSA^l(Z^{l-1}) + MLP(MSA^l(Z^{l-1})) \in \mathbb{R}^{N^2 \times C}. \quad (7.6)$$

Properties of the Attention Tensors

As described above, ViT [DBK⁺20] computes attention matrices $A_i^l(Z^{l-1})$ for layers $l = 1, \dots, L$ and for multiple heads $i = 1, \dots, m$. Due to the softmax activation and the decomposition of the image into patches, these attentions can be viewed as probability distributions over the $N \times N$ image patches plus the extra class-token, which we discard due to its lack of a geometric interpretation. For all $j, j' \in \{1, \dots, N^2\}$ in layer l , the element $A_i^l(Z^{l-1})_{j,j'}$ of the attention matrix reflects how much attention patch j pays to patch j' .

These elements can therefore be visualized as heatmaps over the $N \times N$ image patches, as shown in [Figure 7.2](#), where we visualize the attention averaged over the multiple heads

$$\bar{A}^l(Z^{l-1})_{j,j'} = \frac{1}{m} \sum_{i=1}^m A_i^l(Z^{l-1})_{j,j'}. \quad (7.7)$$

Attention is driven by visual similarity and, thanks to the spatial embedding, proximity. We observe that the attention originating from a visually distinct, moderately sized object remains concentrated sharply within the patches j overlapping to a greater extent with that object. By contrast, for larger areas of visual coherence, such as the road in a traffic scene, the attention is more dispersed over the entire region.

7.1.2 From Attention to Entropy Heatmaps

We quantify this behavior by estimating the *Spatial Shannon Entropy*

$$E^l(Z^{l-1})_j = - \sum_{j'=1}^{N^2} \bar{A}^l(Z^{l-1})_{j,j'} \cdot \log(\bar{A}^l(Z^{l-1})_{j,j'}) \quad (7.8)$$

for each image patch $j = 1, \dots, N^2$. The overall spatial entropy $\mathbf{E}^l = [E^l(Z^{l-1})_j]_{j \in \{1, \dots, N^2\}}$ for layer l can be viewed as an aggregated heatmap for layer l that can be used to segment objects of moderate size, as discussed below.

Layer Selection and Averaging

Each self-attention layer yields its own entropy heatmap \mathbf{E}^l . Inspecting the entropy heatmaps reveals that objects, such as the obstacles in [Figure 7.2](#), usually have low entropy, due to their concentrated attention. Some isolated background patches, such as parts of the road in [Figure 7.2](#), nonetheless exhibit lower or higher entropy than their neighbors. Averaging entropy across layers tends to suppress these artifacts.

The entropy segments the objects in roughly the first half of the layers of the ViT backbone. In our experiments, we have observed that the attention maps of those layers are driven primarily by spatial closeness and visual similarity. The second half of these layers, further removed from the initial spatial embedding, appear to lose this property. Hence the object segmentation capability can be further improved by selecting a subset $\mathcal{L} \subseteq \{1, \dots, L\}$ of layer entropy heatmaps and averaging, *i.e.*, computing

$$\bar{\mathbf{E}} = \frac{1}{L} \sum_{l \in \mathcal{L}} \mathbf{E}^l. \quad (7.9)$$

We study a different strategies to select layers \mathcal{L} in [Section 7.2.3](#) and provide a detailed visualization of per-layer attentions and entropies in the supplementary material.

Segmentation of Entropy Heatmaps

The averaged entropy is used as an object detection heatmap with minimal postprocessing. First, we negate the entropy since objects exhibit *lower* entropy than their surroundings. Then we linearly interpolate the entropy signal from the original resolution of $N \times N$ to a per pixel heatmap in the image resolution and apply a thresholding to it. This corresponds to

$$s(u, v) = \delta(\text{Lin}_{k \in \text{Neighbour}(u,v)}(-\tilde{\mathbf{E}}_k)), \quad (7.10)$$

where $s(u, v)$ is the binary segmentation value for pixel location (u, v) , δ is a threshold function for binarization, Lin is the linear interpolation applied to the neighbouring elements k that contribute to pixel (u, v) . The chosen threshold for δ , which is applied to the segmentation heatmap, makes a trade-off between precision and recall. In our quantitative evaluations, we report the AuPRC over all possible thresholds following the benchmark evaluation protocol of [Chapter 3](#).

7.1.3 Interactive attention visualization

Please view the interactive attention visualization at <https://liskr.net/attentropy>. Hover the cursor over the input image to choose the source patch j . The heatmap will display the attention values from the chosen patch j to each other patch j' , that is the value $\bar{A}_{j,j'}^l$. The attention values are extracted from the ViT [DBK⁺20] backbone of the SETR [ZLZ⁺21] network, and have been truncated to the $[0, 0.005]$ range for visual readability.

- By selecting a source patch within an object, we can observe how its attention is concentrated. In contrast, the road and background areas exhibit diffuse attention. This coincides with lower entropy of the attention map.
- Choose the layer shown using left and right arrow keys. We can observe how the earlier layers' attention is spatially localized, and therefore useful for small object segmentation. The later layers show less localized attention and so we exclude them from the entropy average.

7.1.4 Implementation

In our experiments, we evaluate the semantic segmentation transformers SETR [ZLZ⁺21] and Segformer [XWY⁺21] implemented in the MMSegmentation framework [Con20]. Both have been trained for semantic segmentation with the Cityscapes [COR⁺16] dataset. As both architectures are designed to operate on square images, sliding window inference is automatically performed on our rectangular images.

In the case of SETR [ZLZ⁺21], we extract the attention maps from its ViT backbone. We discard the class token leaving the patch-to-patch attention and calculate the Shannon entropy for

each patch. The entropy from each layer has a resolution of 48×48 , and we perform linear interpolation to upscale it to the input image resolution. We use the *PUP* variant of the SETR decoder head as it has the best reported segmentation performance.

The Segformer [XWY⁺21] uses a pyramid of self-attention operations with the base of the pyramid comprising 256^2 patches. We use the *MIT-B3* variant as a compromise between performance and computation cost. It would not be feasible to calculate attention between each pair of patches. Hence it performs a linear operation reducing r -times the number of rows of K and V , where r is a chosen reduction ratio. As a consequence, the attention matrices A_i^l have only $\frac{N^2}{r}$ rows instead of N^2 . This essentially means the attention map’s resolution is reduced. Nevertheless, we can calculate entropy over each row of \bar{A}^l , upscale the heatmaps to a common shape and average them. The number of tokens, and thus the resolution of the entropy maps, varies across layers from 256×256 to 32×32 . We linearly interpolate all of them to 256×256 before averaging, then resize the average heatmap to full image resolution.

7.2 Experiments

In this section we demonstrate that the attention layers trained in a supervised fashion to segment specific categories carry the required information to segment new objects of small to moderate size under varying degrees of domain shift and finally when changing the domain completely. We do so by evaluating our method and several baselines, including the self-supervised DINO, on a variety of datasets. We show that spatial attention information is much more powerful than other widely used training-free strategies, and that DINO’s spatial attention behaves differently from that of segmentation transformers trained in a supervised fashion.

7.2.1 Datasets

In our study, we use segmentation transformers trained on Cityscapes [COR⁺16] due to their availability and because there exist multiple datasets containing obstacles that are semantically different from the objects present in Cityscapes. We then take these pre-trained transformer models and evaluate them on the following datasets to detect new objects.

RoadObstacle21 and Lost and Found - test NoKnown. For road-obstacle detection we follow the evaluation protocol described in Chapter 3.

MaSTr1325 [BMPK19] (Maritime) contains images of maritime scenes viewed from a small unmanned surface vessel with semantic segmentation labels for *water*, *sky* and *obstacles*.

Artificial Lunar Landscape [PJI19] (Lunar) comprises synthetic lunar scenes where rocks act as obstacles.

Airbus Aircraft Segmentation [FA20] (Aircraft) features satellite images of airports.

Our networks are not trained on these datasets. Instead we apply SETR and Segformer with publicly available Cityscapes checkpoints. On the latter 3 datasets, we show the model can detect unseen objects in completely new domains compared to the trained dataset – Cityscapes.

7.2.2 Gradually Changing the Domain

In this section we start by evaluating road-segmentation models on unknown obstacles in similar scenes and then gradually make a domain shift by first evaluating on challenging and un-observed road environments and later on completely different domains such as maritime, lunar, and aerial scenes.

No Domain Shift

We start by examining the obstacle segmentation performance and report results on LostAnd-Found [PRG⁺16]. While the object are unseen by models the domain is similar to Cityscapes, comprising of road scenes with no major environment differences. We report results in the left section of Table 7.1. We follow the evaluation protocol and metrics introduced in Chapter 3.

The upper section of the table contains benchmark results of methods that do not train specifically for the detection of road obstacles. Monte Carlo (MC) dropout and ensembles are approximations to Bayesian inference, relying on multiple inferences. All the other methods in that section, as well as ours, are purely based on post processing of either the network’s output or embedded features. In particular our method applied to the SETR model clearly outperforms all other baselines in the upper section w.r.t. both pixel-level and segment-level metrics.

It can be observed that the Segformer [XWY⁺21] variant performs slightly worse. Possible causes for the reduced performance are the sequence reduction applied to its attention layers, resulting in coarse attention maps, along with the fact that Segformer does not use the standard positional encoding found in most visual transformers. With the positional codes influencing the query and key vectors, the tokens can easily learn to focus their attention on neighbors. In lieu of such encoding, Segformer relies on 3×3 convolutions, interspersed with the attention layers, to leak positional information from zero-padding on the image edges. Therefore, attention concentrations on very small objects may be less likely to emerge. Illustrative examples of the segmentation performance of our method are provided in Figure 7.3-left.

We have also observed that small obstacles close to the horizon can pose problem to our method. Typically, the attention tends to concentrate in that region of the image as the street narrows in. We expect that digging deeper into the attention structure, as well as utilizing an auxiliary model supplied with our attention masks could yield a strongly performing overall system.

7.2 Experiments

		Lost and Found - test no known					Road Obstacles 21 - test				
		Pixel-level		Segment-level			Pixel-level		Segment-level		
		AP \uparrow	FPR ₉₅ \downarrow	sIoU \uparrow	PPV \uparrow	\bar{F}_1 \uparrow	AP \uparrow	FPR ₉₅ \downarrow	sIoU \uparrow	PPV \uparrow	\bar{F}_1 \uparrow
No training	Ensemble [LPB17]	2.9	82.0	6.7	7.6	2.7	1.1	77.2	8.6	4.7	1.3
	Embedding Density [BSN ⁺ 19]	61.7	10.4	37.8	35.2	27.5	0.8	46.4	35.6	2.9	2.3
	LOST [SPV ⁺ 21]	1.1	94.7	8.6	6.0	6.0	4.7	93.8	17.0	8.3	11.0
	MC Dropout [MG18b]	36.8	35.5	17.4	34.7	13.0	4.9	50.3	5.5	5.8	1.0
	Maximum Softmax [HG17]	30.1	33.2	14.2	62.2	10.3	15.7	16.6	19.7	15.9	6.3
	Mahalanobis [LLS18b]	55.0	12.9	33.8	31.7	22.1	20.9	13.1	13.5	21.8	4.7
	ODIN [LLS18]	52.9	30.0	39.8	49.3	34.5	22.1	15.3	21.6	18.5	9.4
	DINO [CTM ⁺ 21]	26.4	38.9	11.7	13.6	5.7	39.9	14.1	26.8	19.1	12.4
	Ours Segformer _{manual}	56.4	6.7	34.7	35.0	28.4	45.5	8.1	25.4	36.3	22.7
		73.0	2.9	37.1	42.8	38.0	72.9	2.5	36.4	47.8	41.6
Obstacle or anomaly training	Void Classifier [BSN ⁺ 19]	4.8	47.0	1.8	35.1	1.9	10.4	41.5	6.3	20.3	5.4
	JSRNet [VSA ⁺ 21]	74.2	6.6	34.3	45.9	36.0	28.1	28.9	18.6	24.5	11.0
	Image Resynthesis (Chapter 4)	57.1	8.8	27.2	30.7	19.2	37.7	4.7	16.6	20.5	8.4
	Road Inpainting (Chapter 5)	82.9	35.7	49.2	60.7	52.3	54.1	47.1	57.6	39.5	36.0
	SynBoost [DBBSC21]	81.7	4.6	36.8	72.3	48.7	71.3	3.2	44.3	41.8	37.6
	Maximized Entropy [CRG21b]	77.9	9.7	45.9	63.1	49.9	85.1	0.8	47.9	62.6	48.5
	DenseHybrid [GBS22]	78.7	2.1	46.9	52.1	52.3	87.1	0.2	45.7	50.1	50.7

Table 7.1: **Obstacle detection scores.** The primary metrics are Average Precision (AP \uparrow) for pixel classification and Average F_1 (\bar{F}_1 \uparrow) for instance level detection. The evaluation protocol and metrics are described in Chapter 3.

Partial Domain Shift

Next we evaluate models on RoadObstacle21 (Chapter 3). While still being a road dataset, it provides unseen objects in new weather, lighting and background environments compared to Cityscapes. The observations from LostAndFound generalize to this dataset, see right-hand section of Table 7.1. In the presence of slight to moderate domain shift, the performance of our method is preserved. See visualization results in Figure 7.3-right, which further stress the capabilities of our method when weather and lighting conditions change drastically. In these visually difficult conditions, most of the obstacles present are still segmented well from the background.

Complete Domain Shift

We evaluate here the segmentation capability of models on Maritime, Lunar, and Aircraft datasets, which contain unseen objects in completely new domains compared to the road scenes. The results obtained from Maritime dataset are presented in Table 7.2 and the ones obtained from Lunar dataset are shown in Table 7.3. In both cases, our method outperforms the training-free baselines in terms of AP. The results show that not only can the attention entropy segment previously unseen road obstacles, but its small object detection property holds in completely different domains.

On Maritime dataset, the objects on the water visually stand out from their surroundings, thus our method achieves remarkable results, outperforming the self-supervised DINO transformer backbone by a large margin. Visual examples are provided in Figure 7.4.

On the more challenging Lunar dataset where gray rocks are located on gray ground, the performance of our method still outperforms the one of DINO, see Table 7.3. We observe that

Chapter 7. Attention Entropy: Generalizing to New Obstacles in New Domains

	AP \uparrow	FPR ₉₅ \downarrow
Ours SETR _{manual}	63.9	35.0
Ours Segformer _{manual}	54.6	36.7
DINO [CTM ⁺ 21]	41.4	39.5
Maximum Softmax [HG17]	18.6	64.9
ODIN [LLS18]	16.6	70.8
SynBoost [DBBSC21]	14.9	80.0
LOST [SPV ⁺ 21]	10.1	94.4
Mahalanobis [LLLS18b]	5.9	97.3

Table 7.2: **Obstacle pixel segmentation performance in the MaStr1325 [BMPK19] dataset.** We do not measure object-level metrics since the maritime scenes contain non-object obstacles such as land.

	AP \uparrow	FPR ₉₅ \downarrow	sIoU \uparrow	PPV \uparrow	$\overline{F_1}$ \uparrow
Ours Segformer _{manual}	37.8	76.1	5.6	38.2	5.1
Ours SETR _{manual}	35.9	85.0	7.9	29.8	8.3
SynBoost [DBBSC21]	35.6	81.9	4.0	28.8	3.6
DINO [CTM ⁺ 21]	32.5	93.0	6.1	22.4	4.3
ODIN [LLS18]	29.7	75.1	2.6	33.6	2.3
Maximum Softmax [HG17]	28.5	75.1	2.2	33.2	1.9
LOST [SPV ⁺ 21]	18.4	94.9	0.6	39.4	0.5

Table 7.3: **Obstacle pixel segmentation performance in the Artificial Lunar Landscape [PJH19] dataset.** The evaluation was performed on every 10-th image due to big dataset size.

our method tends to also segment shadows caused by uneven ground. Visual examples are provided in Figure 7.5.

Furthermore, we also provide visual examples for the Aircraft dataset [FA20], where no segmentation ground truth is available, see Figure 7.6. In this setting with a completely different viewing angle, our method still segments the objects, as the planes clearly stand out in the entropy maps. We expect this behavior to extend to most small salient objects that are visually distinct from the surrounding environment.

7.2.3 Spatial Attention Study

Layer-Wise Study

In Figure 7.8 we show the attention-entropy heatmaps of each layer of the backbones of SETR (ViT backbone) and Segformer. As discussed in Section 7.1.2 attention is stronger between similar patches. As a consequence, patches containing a distinct object attend strongly to each other and weakly to the background. Thus the entropy of their narrow attention distributions is low, shown as dark in the heatmaps. Meanwhile the attention of the uniform backgrounds is distributed more evenly across the whole area of the image and the resulting entropy is high.

Visual inspection indicates that this concentration of attention within distinct objects is



Figure 7.3: Qualitative results on obstacle detection in traffic scenes. The left two images are from LostAnd-Found [PRG⁺16] and the right three images are from RoadObstacle21 (Chapter 3), including ones with difficult weather and limited light. The middle and bottom rows show the averaged entropy of SETR and Segformer respectively, both using manual layer averaging. The heatmap is overlaid in the evaluation ROI. Slight rectangular artifacts arise from MMSegmentation’s sliding window inference.

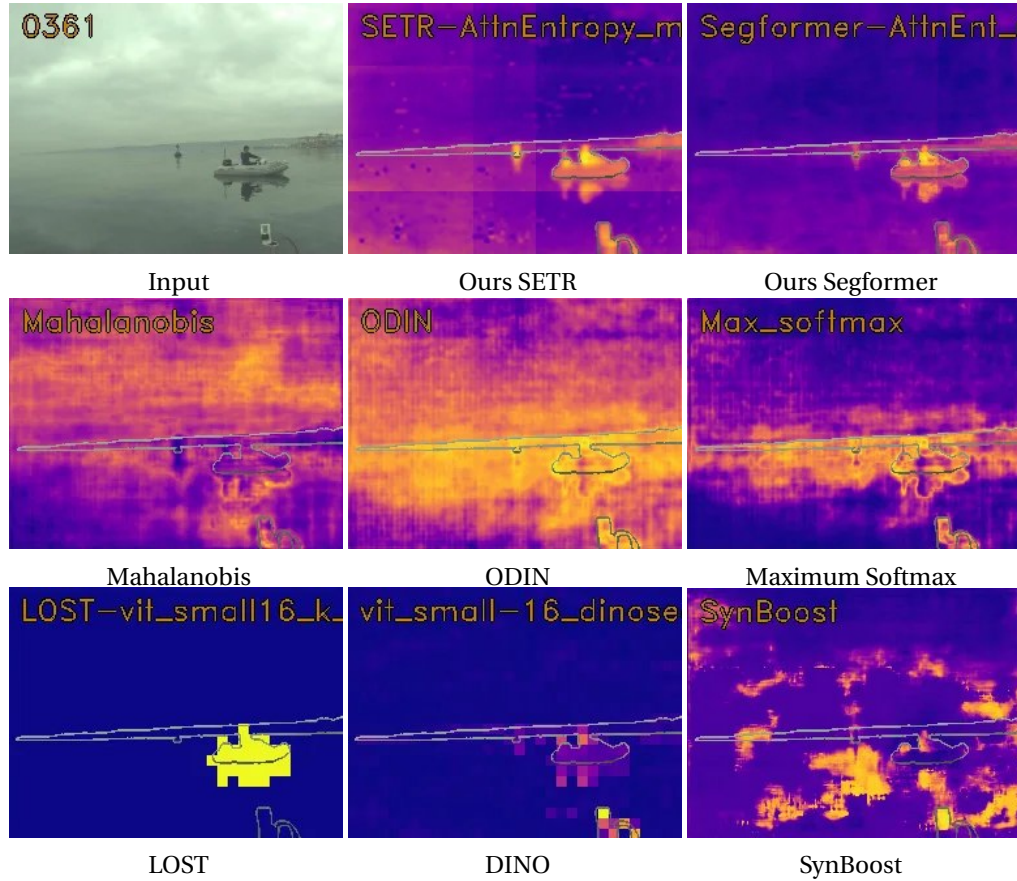


Figure 7.4: Qualitative results on obstacle detection on Maritime Dataset [BMPK19]. The contour of the ground truth obstacle areas is highlighted. Compared to the other training-free obstacle detection methods, our attention entropy generalizes better to the distant domain of maritime scenes.

Chapter 7. Attention Entropy: Generalizing to New Obstacles in New Domains

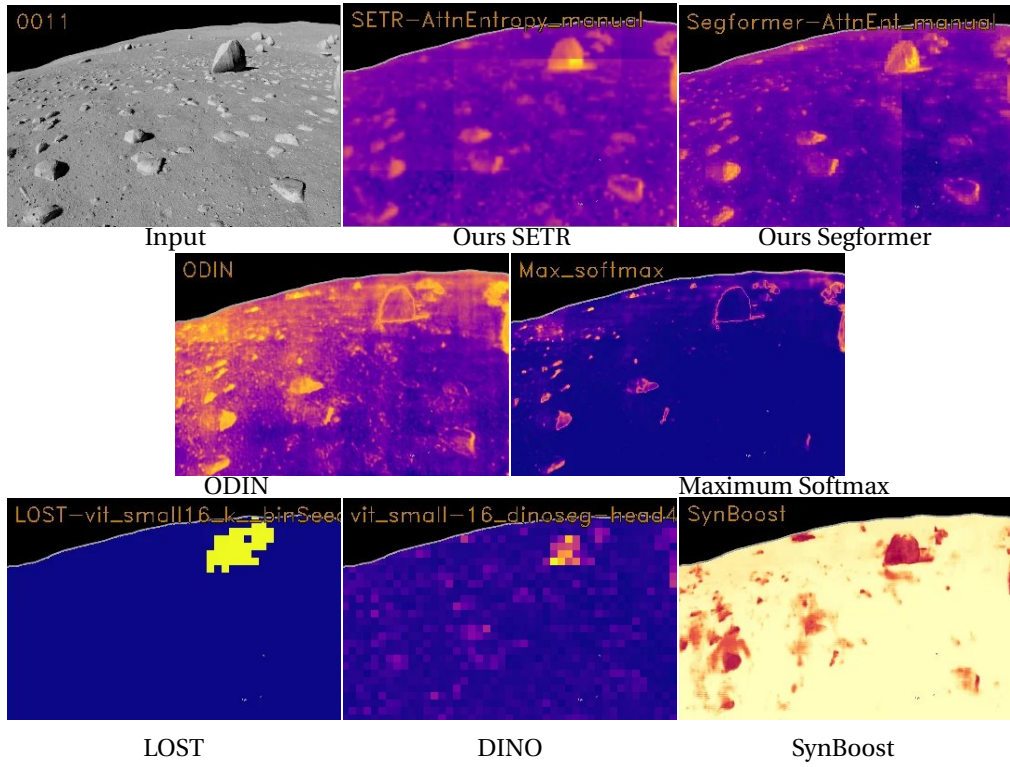


Figure 7.5: Qualitative results on obstacle detection on Lunar dataset [PJI19]. The contour of the ground truth obstacle areas is highlighted.



Figure 7.6: Qualitative results on Aircraft [FA20] dataset. The middle and right columns show the averaged entropy of SETR and Segformer respectively.

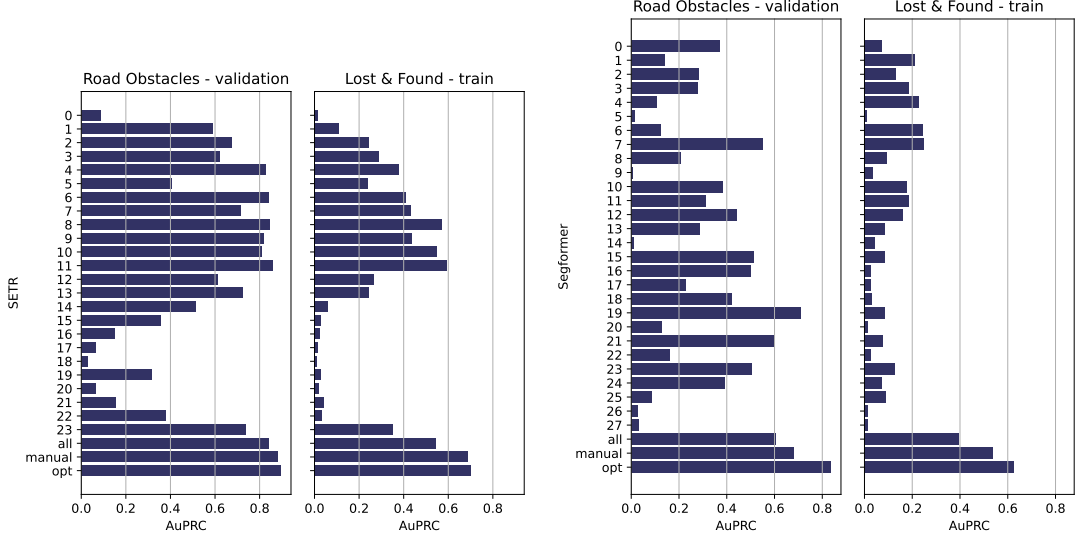


Figure 7.7: Usefulness of individual layers’ entropy for obstacle detection, measured by the area under the precision-recall curve of obstacle segmentation on *Road Obstacles 21 - validation* and *Lost and Found - train* datasets. In addition to layers denoted by their 0-based index, we list the scores for layer averaging strategies discussed in Section 7.2.3.

		Road Obstacles 21 - test					Lost and Found - test no known				
		Pixel-level		Segment-level			Pixel-level		Segment-level		
		AP ↑	FPR ₉₅ ↓	sIoU ↑	PPV ↑	$\overline{F_1}$ ↑	AP ↑	FPR ₉₅ ↓	sIoU ↑	PPV ↑	$\overline{F_1}$ ↑
SETR	all layers	67.0	5.8	30.6	22.1	22.2	62.2	11.2	27.4	28.2	24.9
	manual layers	72.9	2.5	36.4	47.8	41.6	73.0	2.9	37.1	42.8	38.0
	regression weights	71.2	2.3	36.6	40.9	36.9	73.7	4.6	35.4	46.6	38.4
Segformer	all layers	34.7	12.0	22.2	35.9	19.9	47.4	15.6	30.2	35.6	25.1
	manual layers	45.5	8.1	25.4	36.3	22.7	56.4	6.7	34.7	35.0	28.4
	regression weights	57.4	6.3	34.2	33.8	29.1	62.8	5.3	35.4	34.5	29.5

Table 7.4: **Obstacle detection scores for layer averaging strategies.** The primary metrics are Average Precision (AP↑) for pixel classification and Average F_1 ($\overline{F_1}$ ↑) for instance level detection. The evaluation protocol and metrics are described in Chapter 3.

stronger in some layers than others. For best object detection, we choose a subset of layers, indicated by a green border in Figure 7.8.

We study the usefulness of individual layers’ entropy heatmaps for obstacle detection. We measure the area under the precision-recall curve of obstacle segmentation on the *Road Obstacles validation* set and the *Lost and Found* training set – those datasets have publicly available obstacle labels and do not intersect with the benchmark test sets used for comparisons. We plot the layers’ scores in Figure 7.7, together with scores calculated for the following averaging strategies:

- *all layers*: the output heatmap is the average of all attention-entropy heatmaps \mathbf{E}^l , *i.e.*, $\mathcal{L} = \{1, \dots, l\}$.
- *manual layers*: we average a subset of layers whose entropy corresponds well to obsta-

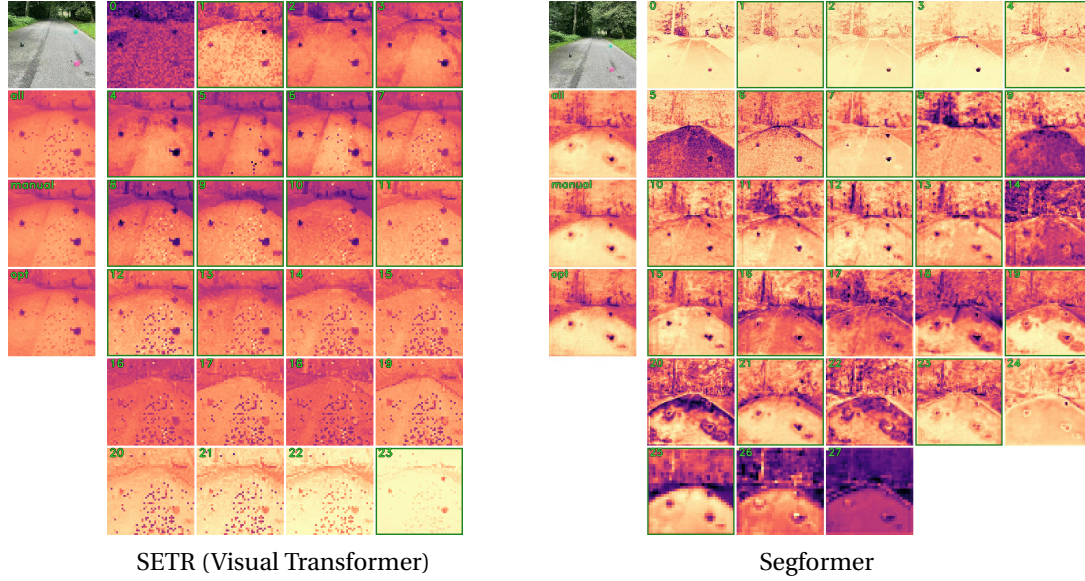


Figure 7.8: Attention-entropy heatmaps for layers of SETR’s Visual Transformer and Segformer backbone. The individual layer entropies are denoted with a 0-based layer index. We observe that not all layers are equally useful for segmenting obstacles from the road. Based on visual inspection of examples, we select a subset of useful layers, marked here by a green border. In the case of SETR, roughly the earlier half of them (except the 0-th layer) exhibit distinctly low entropy in the regions of obstacles. The attention of the later layers becomes less localized and so the entropy differences across the image decrease. Segformer’s layers do not form such a clear pattern, but still its layers vary in the extent to which their entropies segment the obstacles. Many layers have artifact tokens with concentrated attention (thus low entropy) which do not correspond to any visible object. Averaging the layers tends to suppress these noisy artifacts, as visible on the averaged heatmaps; The *all*, *manual* and *optimized* averaging strategies are described in Sec. 4.3.

cles, as confirmed by visual inspection of example outputs. The chosen layers are shown in the supplementary material.

- *optimized weights*: we use logistic regression to determine the optimal weights to mix the layer entropy heatmaps, *i.e.*, $\sigma(\sum_{l \in \mathcal{L}} a_l \mathbf{E}^l + b\mathbf{1})$, where σ is the sigmoid activation, $a_l, b \in \mathbb{R}$ and $\mathbf{1} \in \mathbb{R}^{N \times N}$ is a matrix with all elements equal to one. The weights are obtained using the frames from *Road Obstacles - Validation* and *Lost and Found - train*.

We compare the layer averaging strategies in Table 7.4. The results show that while averaging entropy across all layers obtains decent results, a simple visual choice of layers yields further improvement in manual layers. The regression weighting obtains either similar results as manual (for SETR) or further improves compared to it (for Segformer). We used the manual selection in our comparisons as it does not apply any fine-tuning or learning on top of pre-trained network.

Concentration of Attention in Self-Supervision

In Figure 7.9 we visualize the attention-entropy sum of the self-supervised DINO [CTM⁺21] transformer and the transformers trained for Cityscapes segmentation. Both DINO and SETR

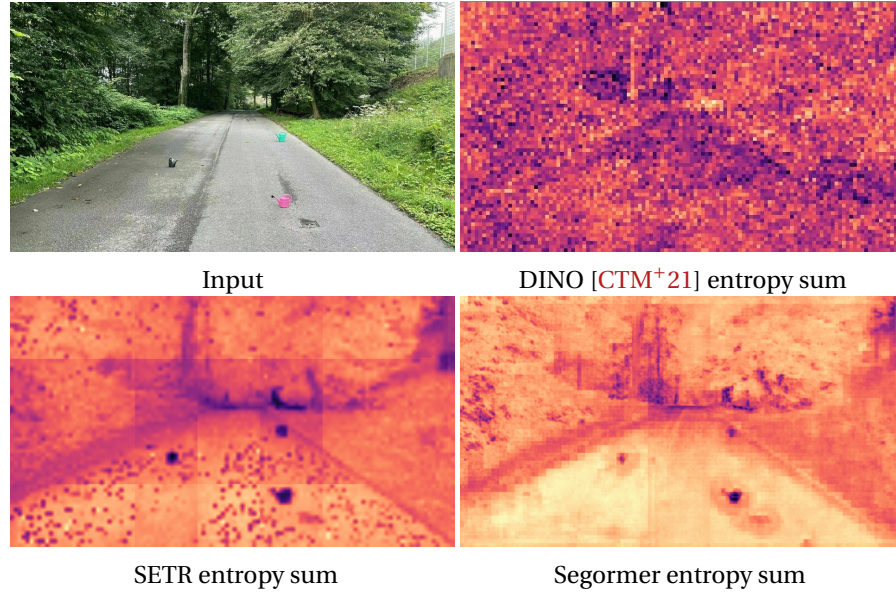


Figure 7.9: Sums of attention-entropy over all layers - comparison between DINO [CTM⁺21] trained in a self-supervised manner on ImageNet versus SETR [ZLZ⁺21] and Segformer [XWY⁺21] trained for Cityscapes semantic segmentation.

have a ViT [DBK⁺20] network as their backbones. Even though the self-supervised DINO’s class-token attention of the last layer can segment foreground objects, its intermediate layer attention does not exhibit the clear object segmenting property we observe in semantic segmentation networks.

7.3 Conclusion

We performed an in-depth study of the spatial attentions of vision transformers trained in a supervised manner for semantic segmentation [ZLZ⁺21, XWY⁺21, DBK⁺20]. For image patches associated to salient objects, the attention of intermediate layers tends to concentrate on that object, which we quantified via the Shannon entropy. On the other hand, for larger areas of coherent appearance, the entropy diffuses around the given image patch under consideration. This observation holds for many different layers of the transformer backbone and averaging their attentions improves the segmentation performance. We demonstrated that the attention in self-supervised trained transformers such as DINO exhibits a different behavior. Our observations hold for different degrees of domain shift, leading to a method that can segment unknown objects of small to moderate size in unknown context. We demonstrated this on several datasets with varying degrees of domain shift, up to a complete change of domain. Due to the negligible computational overhead w.r.t. the supervised model, our method might be suitable for automated driving applications, robotics applications and also for pre-segmenting objects for the sake of data annotation.

8 Benchmark Results

Having presented our anomaly and obstacle detection methods, in this chapter we return to the benchmark introduced in [Chapter 3](#) to present the comparative results of our approaches as well as those of others. Since the beginning of our research project we have seen fast growing interest in the field of road anomaly detection, including a significant number of new methods submitted to the *Segment Me If You Can* benchmark after its release. The results shown in this chapter have been updated to include the latest submissions and overall they reflect tremendous progress in solutions to the problem.

8.1 Evaluated Methods

Several anomaly segmentation methods have already been evaluated on our benchmark and constitute our initial leader board. In the original benchmark we evaluate at least one method per type discussed in [Chapter 2](#), namely

- *Methods originating from image classification*: **maximum softmax probability** [[HG17](#)], **ODIN** [[LLS18](#)], **Mahalanobis distance** [[LLLS18b](#)];
- *Bayesian model uncertainty*: **Monte Carlo (MC) dropout** [[MG18b](#)], **ensemble** [[LPB17](#)];
- *Learning to identify anomalies*: **learned embedding density** [[BSN⁺19](#)], **void classifier** [[BSN⁺19](#)], **maximized softmax entropy** [[CRG21b](#)];
- *Reconstruction via generative models*: **image resynthesis** ([Chapter 4](#)), **SynBoost** [[DBBSC21](#)] and **road inpainting** (obstacle track only) ([Chapter 5](#)).

All methods have an underlying semantic segmentation DNN trained on Cityscapes and provide pixel-wise anomaly scores. A semantic segmentation DNN trained on Cityscapes is also our recommendation as underlying model, however, we leave it up to the participants which network and training data they use. Furthermore, some evaluated methods additionally employ out-of-distribution (OoD) data to tune the anomaly detector. For our set of methods,

Chapter 8. Benchmark Results

this would be any data with labels semantically different from the Cityscapes train classes. OoD data is also allowed to be used to alleviate the effects of a potential domain shift.

All methods provide pixel-wise anomaly scores $s(x) \in \mathbb{R}^{|\mathcal{Z}|}$, $x \in \mathcal{X}$ where anomalies correspond to higher values. As a reminder, \mathcal{Z} denotes the set of image coordinates and $\mathcal{X} \subseteq [0, 1]^{N \times |\mathcal{Z}| \times 3}$ a dataset with N images. Below, we describe how s is obtained for the baselines in the original benchmark.

Maximum softmax probability

Let $f: \mathcal{X} \rightarrow \mathbb{R}^{|\mathcal{Z}| \times |\mathcal{C}|}$ denote the output of a semantic segmentation DNN. The maximum softmax probability (MSP) is a commonly-used baseline for OoD detection at image level [HG17]. It computes an anomaly score for each pixel $z \in \mathcal{Z}$ as

$$s_z(x) = 1 - \max_{c \in \mathcal{C}} \sigma(f_z^c(x)), \quad x \in \mathcal{X}, \quad (8.1)$$

where $\sigma(\cdot): \mathbb{R}^{|\mathcal{C}|} \rightarrow (0, 1)^{|\mathcal{C}|}$ denotes the softmax function over the non-anomalous class set \mathcal{C} .

ODIN

Let $t \in \mathbb{R} \setminus \{0\}$ be a temperature scaling parameter and $\varepsilon \in \mathbb{R}$ a perturbation magnitude. Following [LLS18] small perturbations are added to every pixel $z \in \mathcal{Z}$ of image x by

$$\tilde{x}_z = x_z - \varepsilon \text{sign} \left(-\frac{\partial}{\partial x_z} \log \max_{c \in \mathcal{C}} \sigma(f_z^c(x)/t) \right). \quad (8.2)$$

Then, an anomaly score is obtained analogously to Equation (8.1) via the MSP as

$$s_z(x) = 1 - \max_{c \in \mathcal{C}} \sigma(f_z^c(\tilde{x})/t). \quad (8.3)$$

Mahalanobis distance

Let $h^{L-1}(\cdot)$ denote the output of the penultimate layer of a DNN with $L \in \mathbb{N}$ layers, *i.e.* $f(x) = h^L(x)$, $x \in \mathcal{X}$. Under the assumption that

$$P(h_z^{L-1}(x) \mid y_z(x) = c) = \mathcal{N}(h_z^{L-1}(x) \mid \mu^c, \Sigma^c), \quad (8.4)$$

an anomaly score for each pixel z can be computed as the Mahalanobis distance [LLS18b]

$$s_z(x) = \min_{c \in \mathcal{C}} (h_z^{L-1}(x) - \hat{\mu}^c)^T \hat{\Sigma}^{c^{-1}} (h_z^{L-1}(x) - \hat{\mu}^c), \quad (8.5)$$

where $\hat{\mu}^c$ and $\hat{\Sigma}^c$ are estimates of the class mean μ^c and class covariance Σ^c , respectively, of the latent features in the penultimate layer. This Mahalanobis distance yields an estimate of

the likelihood of a test sample with respect to the closest class distribution in the training data, which are assumed to be class-conditional Gaussians.

Monte Carlo dropout

Let $M \in \mathbb{N}$ denote the number of Monte Carlo sampling rounds and $\hat{q}_m^c := \sigma(f_z^c(x))$ the softmax probability of class $c \in \mathcal{C}$ for sample $m \in \{1, \dots, M\}$. The predictive entropy is computed as

$$\hat{E}(f(x)) = - \sum_{c \in \mathcal{C}} \left(\frac{1}{M} \sum_{m=1}^M \hat{q}_m^c \right) \log \left(\frac{1}{M} \sum_{m=1}^M \hat{q}_m^c \right). \quad (8.6)$$

As suggested in [MG18b], the mutual information can then be used to define an anomaly score

$$s_z(x) = \hat{E}(f(x)) - \frac{1}{M} \sum_{c \in \mathcal{C}} \sum_{m=1}^M \hat{q}_m^c \log(\hat{q}_m^c). \quad (8.7)$$

Ensemble

Similar to Monte Carlo dropout, multiple samples of softmax probabilities $\hat{q}_m^c := \sigma(f_z^c(x))$, $c \in \mathcal{C}$, $m \in \{1, \dots, M\}$ are drawn from multiple semantic segmentation models. Those models have the same network architecture but are trained with different weights initialization [LPB17]. Again, the mutual information is used as anomaly score

$$s_z(x) = \hat{E}(f(x)) - \frac{1}{M} \sum_{c \in \mathcal{C}} \sum_{m=1}^M \hat{q}_m^c \log(\hat{q}_m^c). \quad (8.8)$$

Void classifier

In [DT18], an approach to learning the confidence with respect to the presence of anomalies was proposed. Here, we adapt this by using the Cityscapes void class to approximate the anomaly distribution. We then trained a Cityscapes DNN $f: \mathcal{X} \mapsto \mathbb{R}^{|Z| \times (|\mathcal{C}|+1)}$ with an additional class such as a trash can [ZL17], and compute the anomaly score for each pixel $z \in \mathcal{Z}$ as the softmax score for the void class, which yields

$$s_z(x) = \sigma(f_z^{\text{void}}(x)), \quad x \in \mathcal{X}. \quad (8.9)$$

Learned embedding density

Let $h^l(x) \in \mathbb{R}^{|\mathcal{Z}'| \times n_l}$, $n_l \in \mathbb{N}$, $\mathcal{Z}' \subset \mathcal{Z}$, be the embedding vector of a segmentation DNN at layer $l \in \{1, \dots, L\}$ for image $x \in \mathcal{X}$. The true distribution $p^*(h^l(x))$, $x \in \mathcal{X}_{\text{train}} \subset \mathcal{X}$ can be approximated with a normalizing flow $\hat{p}(h^l(x)) \approx p^*(h^l(x))$. At test time, the negative log-likelihood $-\log \hat{p}_{z'}(h^l(x)) \in (0, \infty)$ for each embedding location $z' \in \mathcal{Z}'$ then measures the discrepancy of a test embedding with respect to training embeddings, where higher discrepancies indicate

Chapter 8. Benchmark Results

anomalies [BSN⁺19]. The resulting anomaly score map are of size $|\mathcal{Z}'| = \frac{1}{n}|\mathcal{Z}|$, with $n \in \mathbb{N}$ the rescaling factor for \mathcal{Z}' to match the size of \mathcal{Z} , and hence bring back latent features to the full image resolution $|\mathcal{Z}|$ via bilinear interpolation $u: \mathbb{R}^{|\mathcal{Z}'|} \rightarrow \mathbb{R}^{|\mathcal{Z}|}$. This yields an anomaly score for each $z \in \mathcal{Z}$ as

$$s_z(x) = u_z \left((-\log \hat{p}(h_{z'}^l(x)))_{z' \in \mathcal{Z}'} \right), x \in \mathcal{X}. \quad (8.10)$$

SynBoost

This approach follows a similar idea as image resynthesis but includes further inputs in the discrepancy module. In particular, for all $z \in \mathcal{Z}$ the pixel-wise softmax entropy

$$H_z(x) = - \sum_{c \in \mathcal{C}} \sigma(f_z^c(x)) \log(\sigma(f_z^c(x))) \quad (8.11)$$

and the pixel-wise softmax distance

$$D_z(x) = 1 - \max_{c \in \mathcal{C}} \sigma(f_z^c(x)) + \max_{c' \in \mathcal{C} \setminus \{\arg \max_{c \in \mathcal{C}} \sigma(f_z^c(x))\}} \sigma(f_z^{c'}(x)) \quad (8.12)$$

are included. The anomaly score for $x \in \mathcal{X}$ is then obtained via

$$s_z(x) = d_z \left(\hat{y}(x), g(\hat{y}(x)), x, H(x), D(x) \right). \quad (8.13)$$

Maximized entropy

Starting from a pretrained DNN, a second training objective is introduced to maximize the softmax entropy on OoD pixels [CRG21b, HMD19, JRF20]. This yields the multi-criteria loss function

$$(1 - \lambda) \mathbb{E}_{(x,y) \sim \mathcal{D}_{in}} [\ell_{in}(\sigma(f_z(x)), y_z(x))] + \lambda \mathbb{E}_{x' \sim \mathcal{D}_{out}} [\ell_{out}(\sigma(f_z(x')))], \lambda \in [0, 1], \quad (8.14)$$

where ℓ_{in} is the empirical cross entropy and ℓ_{out} the averaged negative log-likelihood over all classes for the in-distribution data \mathcal{D}_{in} and the out-distribution data \mathcal{D}_{out} , respectively. To approximate \mathcal{D}_{out} , a subset of the COCO dataset [LMB⁺14] is used whose images do not depict any object classes also available in \mathcal{D}_{in} , which is the Cityscapes dataset [COR⁺16]. The COCO subset together with the Cityscapes training data are then included into a tender retraining of the pretrained Cityscapes model. The anomaly score is then computed via the softmax entropy as

$$s_z(x) = - \sum_{c \in \mathcal{C}} \sigma(f_z^c(x)) \log(\sigma(f_z^c(x))). \quad (8.15)$$

8.2 Numerical Experiments

Method	requires OoD data	Pixel-level			Component-level											
		Anomaly scores			$k \in \mathcal{K}$ sIoU \uparrow	$k \in \mathcal{K}$ PPV \uparrow	$\tau = 0.25$			$\tau = 0.50$			$\tau = 0.75$			$\overline{F_1} \uparrow$
Ensemble [LPB17]	No	17.7	91.1	27.8	16.4	20.8	197	1454	7.3	233	1511	3.2	254	1553	0.9	3.4
Mahalanobis [LLLS18b]	No	20.0	87.0	31.9	14.8	10.2	206	1564	6.0	241	1602	2.2	257	1619	0.5	2.7
Maximum Softmax [HG17]	No	28.0	72.0	34.2	15.5	15.3	204	778	10.6	233	810	5.3	256	833	1.1	5.4
MC Dropout [MG18b]	No	28.9	69.5	39.1	20.5	17.3	175	1592	9.0	225	1640	3.8	252	1667	1.0	4.3
ODIN [LLS18]	No	33.1	71.7	39.1	19.5	17.9	182	1123	10.9	226	1163	4.9	254	1191	1.1	5.2
JSRNet [vSA ⁺ 21]	No	33.6	43.9	42.6	20.2	29.3	180	331	24.3	218	362	13.2	250	383	3.7	13.7
Void Classifier [BSN ⁺ 19]	Yes	36.6	63.5	44.3	21.1	22.1	181	987	12.2	219	1034	6.4	253	1066	1.3	6.5
Embedding Density [BSN ⁺ 19]	No	37.5	70.8	48.7	33.9	20.5	107	1750	14.3	176	1811	8.0	250	1884	1.1	7.9
PEBAL [TLP ⁺ 22]	Yes	49.1	40.8	54.5	38.9	27.2	98	981	23.3	161	1024	14.6	229	1089	4.8	14.5
Image Resynthesis (Chapter 4)	No	52.3	25.9	60.5	39.7	11.0	94	1242	20.1	153	1287	13.1	231	1364	3.7	12.5
SynBoost [DBBSC21]	Yes	56.4	61.9	58.0	34.7	17.8	111	1255	18.1	179	1317	10.0	248	1387	1.7	10.0
NFlowJS [GBS21a]	No	56.9	34.7	58.4	36.9	18.0	103	792	26.2	173	840	14.9	241	936	3.4	14.9
ObsNet [BBPA21]	No	75.4	26.7	79.7	44.2	52.6	104	175	53.1	126	182	46.9	181	209	29.3	45.1
DenseHybrid [GBS22]	Yes	78.0	9.8	78.6	54.2	24.1	59	557	39.7	99	577	32.5	180	642	16.6	31.1
Maximized Entropy [CRG21b]	Yes	85.5	15.0	77.4	49.2	39.5	85	563	35.3	115	586	29.5	163	626	20.1	28.7
RbA [NYHG23]	No	86.1	15.9	79.1	56.3	41.4	67	339	49.0	93	352	43.2	147	380	30.4	42.0
EAM [Mat23]	Yes	93.7	4.1	91.0	67.1	53.8	54	167	65.3	68	175	61.5	94	180	55.1	60.9

Table 8.1: Benchmark results for our RoadAnomaly21 dataset. This dataset contains 262 ground-truth components in total. The main performance metrics are highlighted with gray columns. The rows corresponding to our methods are highlighted. The results have been updated since the original publication of the papers to reflect new submissions and slight changes in the evaluation protocol.

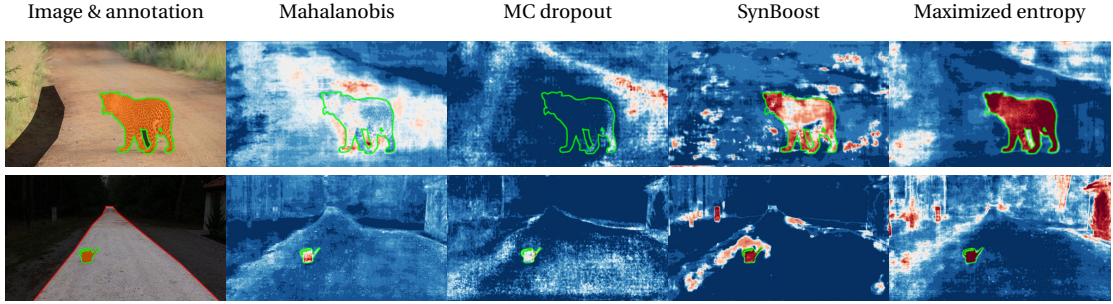


Figure 8.1: Qualitative comparison of the anomaly scores produced by the methods introduced in Section 8.1 for one example image of RoadAnomaly21 (top row) and one example image of RoadObstacle21 (bottom row). Here, red indicates higher anomaly / obstacle scores and blue lower ones. The ground-truth anomaly / obstacle component is indicated by green contours.

8.2 Numerical Experiments

In our benchmark suite we integrate a default method to generate the anomaly segmentation from pixel-wise anomaly scores. We choose the threshold δ^* , at which one pixel is classified as anomaly, by means of the optimal pixel-wise F_1 -score, that we denote with F_1^* . Then, δ^* is computed as

$$\delta^* = \arg \max_{\delta \in \mathbb{R}} 2 \cdot \text{precision}(\delta) \cdot \text{recall}(\delta) / (\text{precision}(\delta) + \text{recall}(\delta)), \quad (8.16)$$

subject to $\text{precision}(\delta) + \text{recall}(\delta) > 0 \forall \delta$.

Moreover, for the anomaly track, components smaller than 500 pixels are discarded from the predicted segmentation, and for the obstacle track, components smaller than 50 pixels are discarded. These sizes are chosen based on the smallest ground-truth components. All methods

Chapter 8. Benchmark Results

Method	requires OoD data	Pixel-level			Component-level										
		Anomaly (obstacle) scores			$k \in \mathcal{K}$	$\hat{k} \in \hat{\mathcal{K}}$	$\tau = 0.25$			$\tau = 0.50$			$\tau = 0.75$		
		AuPRC \uparrow	FPR ₉₅ \downarrow	F_1^* \uparrow	sIoU \uparrow	PPV \uparrow	FN \downarrow	FP \downarrow	F_1 \uparrow	FN \downarrow	FP \downarrow	F_1 \uparrow	FN \downarrow	FP \downarrow	F_1 \uparrow
Embedding Density [BSN ⁺ 19]	No	0.8	46.4	2.0	35.6	2.9	145	11166	4.1	244	11271	2.4	370	11393	0.3
Ensemble [LPB17]	No	1.1	77.2	3.1	8.6	4.7	335	3758	2.5	365	3768	1.1	382	3782	0.3
MC Dropout [MG18b]	No	4.9	50.3	9.0	5.5	5.8	356	2322	2.3	375	2339	0.9	387	2351	0.1
PEBAL [TLP ⁺ 22]	Yes	5.0	12.7	9.1	29.9	7.6	199	3508	9.3	284	3529	5.2	342	3567	2.3
Void Classifier [BSN ⁺ 19]	Yes	10.4	41.5	23.3	6.3	20.3	350	403	9.2	365	421	5.5	381	435	1.7
Maximum Softmax [HG17]	No	15.7	16.6	22.5	19.7	15.9	255	1673	12.1	326	1738	5.7	372	1783	1.5
Mahalanobis [LLLS18b]	No	20.9	13.1	25.8	13.5	21.8	295	1265	10.7	353	1321	4.0	385	1352	0.3
ODIN [LLS18]	No	22.1	15.3	30.1	21.6	18.5	260	1226	14.7	307	1271	9.3	360	1324	3.2
JSRNet [vSA ⁺ 21]	No	28.1	28.9	30.3	18.6	24.5	267	768	19.0	315	772	11.8	375	788	2.2
Image Resynthesis (Chapter 4)	No	37.7	4.7	38.8	16.6	20.5	286	895	14.7	334	940	7.8	374	978	2.0
AttEntropy Segformer (Chapter 7)	No	45.5	8.1	47.3	25.8	35.1	226	404	34.0	282	414	23.3	347	437	9.5
Road Inpainting (Chapter 5)	No	54.1	47.1	67.4	57.6	39.5	79	674	45.1	131	731	37.4	240	847	21.4
SynBoost [DBBSC21]	Yes	71.3	3.2	70.8	44.3	41.8	136	388	49.0	185	440	39.4	283	538	20.4
AttEntropy SETR (Chapter 7)	No	72.9	2.5	70.9	36.4	47.8	156	120	62.7	217	170	46.9	343	276	12.7
Perspective Aware (Chapter 6)	No	75.2	53.7	73.4	65.2	60.2	68	223	68.7	95	228	64.5	174	269	49.1
DaCUP [VM23]	No	81.5	1.1	75.9	37.7	60.1	156	124	62.4	216	134	49.6	335	160	17.6
Maximized Entropy [CRG21b]	Yes	85.1	0.8	79.6	47.9	62.6	136	202	59.9	177	250	49.7	247	321	33.2
NFlowJS [GBS21a]	No	85.5	0.4	79.1	45.5	49.5	143	197	59.0	171	201	53.8	271	276	30.0
DenseHybrid [GBS22]	Yes	87.1	0.2	81.9	45.7	50.1	113	151	67.6	164	187	56.1	309	289	20.9
RbA [NYHG23]	No	87.8	3.3	83.1	47.4	56.2	142	211	58.2	170	212	53.3	253	220	36.3
EAM [Mat23]	Yes	92.9	0.5	90.4	65.9	76.5	71	83	80.5	86	83	78.1	167	92	63.1

Table 8.2: Benchmark results for our RoadObstacle21 dataset. This dataset contains 388 ground-truth components in total. The main performance metrics are highlighted with gray columns. The rows corresponding to our methods are highlighted. The results have been updated since the original publication of the papers to reflect new submissions and slight changes in the evaluation protocol.

Method	requires OoD data	Pixel-level			Component-level										
		Anomaly scores			$k \in \mathcal{K}$	$\hat{k} \in \hat{\mathcal{K}}$	$\tau = 0.25$			$\tau = 0.50$			$\tau = 0.75$		
		AuPRC \uparrow	FPR ₉₅ \downarrow	F_1^* \uparrow	sIoU \uparrow	PPV \uparrow	FN \downarrow	FP \downarrow	F_1 \uparrow	FN \downarrow	FP \downarrow	F_1 \uparrow	FN \downarrow	FP \downarrow	F_1 \uparrow
Ensemble [LPB17]	No	2.9	82.0	8.2	6.7	7.6	1523	5633	4.9	1604	5649	2.8	1695	5705	0.4
Void Classifier [BSN ⁺ 19]	Yes	4.8	47.0	13.7	1.8	35.1	1661	864	3.7	1686	887	1.8	1704	905	0.4
Maximum Softmax [HG17]	No	30.1	33.2	32.5	14.2	62.2	1256	1222	26.8	1575	1522	8.0	1701	1647	0.5
MC Dropout [MG18b]	No	36.8	35.5	42.0	17.4	34.7	1204	2072	23.6	1428	2279	13.2	1635	2484	3.5
ODIN [LLS18]	No	52.9	30.0	55.7	39.8	49.3	701	1552	47.2	954	1803	35.4	1319	2163	18.3
Mahalanobis [LLLS18b]	No	55.0	12.9	54.8	33.8	31.7	777	2559	35.8	1126	2911	22.4	1527	3309	7.0
AttEntropy Segformer (Chapter 7)	No	56.4	6.7	56.1	34.6	36.1	736	1675	44.7	1073	1821	30.5	1543	2024	8.5
Image Resynthesis (Chapter 4)	No	57.1	8.8	55.1	27.2	30.7	947	2379	31.4	1232	2667	19.7	1560	2989	6.1
Embedding Density [BSN ⁺ 19]	No	61.7	10.4	61.7	37.8	35.2	646	2205	42.7	963	2562	29.7	1526	3103	7.3
AttEntropy SETR (Chapter 7)	No	73.0	2.9	67.9	37.1	42.8	745	932	53.5	1008	1091	40.0	1424	1332	17.1
JSRNet [vSA ⁺ 21]	No	74.2	6.6	68.7	34.3	45.9	864	1168	45.4	1032	1187	37.9	1363	1246	21.0
Maximized Entropy [CRG21b]	Yes	77.9	9.7	76.8	45.9	63.1	639	777	60.2	781	911	52.3	1113	1244	33.6
DenseHybrid [GBS22]	Yes	78.7	2.1	78.0	46.9	52.1	579	761	62.8	720	808	56.4	1202	1089	30.7
DaCUP [VM23]	No	81.4	7.4	75.6	38.3	67.3	765	395	61.9	930	409	53.8	1293	476	32.0
SynBoost [DBBSC21]	Yes	81.7	4.6	75.2	36.8	72.3	775	292	63.6	942	459	52.3	1381	898	22.4
Road Inpainting (Chapter 5)	No	82.9	35.7	79.1	49.2	60.7	631	816	59.8	749	944	53.1	958	1163	41.5
Perspective Aware (Chapter 6)	No	85.8	2.5	79.3	47.2	86.1	648	155	72.5	777	158	66.6	1015	167	54.0
NFlowJS [GBS21a]	No	89.3	0.7	82.8	54.6	59.7	478	622	69.1	612	658	63.3	939	795	47.0

Table 8.3: Benchmark results for the Lost and Found test-NoKnown dataset. This dataset contains 1709 ground-truth components in total. The main performance metrics are highlighted with gray columns. The rows corresponding to our methods are highlighted. The results have been updated since the original publication of the papers to reflect new submissions and slight changes in the evaluation protocol.

presented in Section 8.1 produce anomaly scores for which we apply the default segmentation method. We emphasize that using our proposed default method for anomaly segmentation masks is completely optional. We allow and encourage competitors in the benchmark to submit their own anomaly segmentation masks generated via more sophisticated image operations.

In our evaluation, we additionally include the average sIoU per component $\overline{\text{sIoU}}$, which can be computed by averaging sIoU over all ground-truth components $k \in \mathcal{K}$. Analogously, we also include the average PPV per component $\overline{\text{PPV}}$ for all predicted components $\hat{k} \in \hat{\mathcal{K}}$. As the number of component-wise TP, FN and FP depends on some threshold τ for sIoU and PPV, respectively (see Section 3.5), we average these quantities over different thresholds $\tau \in \mathcal{T} = \{0.25, 0.30, \dots, 0.75\}$, similarly to [LMB⁺14], yielding the averaged component-wise F_1 score $\overline{F_1} = \frac{1}{|\mathcal{T}|} \sum_{\tau \in \mathcal{T}} F_1(\tau)$.

Discussion of the Results

Our benchmark results for RoadAnomaly21 and RoadObstacle21 are summarized in Table 8.1 and Table 8.2, respectively. The tables have been updated since the original publication of the papers to reflect new submissions and slight changes in the evaluation protocol. The benchmark receives continuous new submissions presented on the website¹. In general, we observe that methods originally designed for image classification, including maximum softmax, ODIN, and Mahalanobis, do not generalize well to anomaly and obstacle segmentation. For methods based on statistics of the Cityscapes dataset, such as Mahalanobis as well as learned embedding density, anomaly detection is typically degraded by the presence of a domain shift. This results in a poor performance, particularly in RoadObstacle21, where various road surfaces can be observed. Interestingly, learned embedding density, MC dropout and the void classifier yield worse performance than maximum softmax on RoadObstacle21, whereas we observe the opposite on RoadAnomaly21.

The detection methods based on generative models, namely image resynthesis and SynBoost, appear to be better suited to both anomaly and obstacle segmentation at pixel as well as component level, clearly being superior to all the approaches discussed previously. This observation also holds for road inpainting in the obstacle track. These autoencoder-based methods are nonetheless limited by their discrepancy module, and they are outperformed in our experiments by maximized softmax entropy, which peaks at an AuPRC of 86% and a component-wise $\overline{F_1}$ of 49%. This highlights the importance of anomaly and obstacle proxy data. Illustrative example score maps produced by the discussed methods are shown in Figure 8.1.

The recent methods based on mask segmentation [NYHG23, Mat23] have shown a great improvement in both anomaly and obstacle detection. These approaches are also powered by

¹Segment Me If You Can leaderboard: <https://segmentmeifyoucan.com/leaderboard/>

Chapter 8. Benchmark Results

new transformer-based semantic segmenters. Overall this direction is very promising for the future of anomaly and obstacle detection.

A very promising approach to anomaly detection has been enabled by mask-based semantic segmentation [CSK21, CMS⁺22, HOLH21] around the time of the conclusion of this thesis. These methods forgo outputting a simple one-hot logit distribution from the network’s final layer. Instead, one branch produces per-pixel embeddings while a transformer branch creates query embeddings. The dot product of pixel and query embeddings yields a collection of masks. These masks are trained to respond to different semantic classes, or object instances in the case of panoptic segmentation.

Since the masks respond only to their chosen class, the anomaly areas are not covered by any mask. Anomalies are found simply as areas where the sum of all masks is very low [NYHG23]. The performance is even better when false-positives at semantic borders are reduced [Mat23]. The rejection approach is particularly impressive since it achieves great anomaly detection performance without need for retraining or modification of the mask-based segmentation system.

In summary, the component-level evaluation highlights the methods’ weaknesses even more clearly than the pixel-wise evaluation, the latter giving a stronger weight to larger anomalies and obstacles. All methods indeed tend to face difficulties in the presence of smaller anomalies and obstacles. In addition, we observe a much lower component-wise $\overline{F_1}$ score than a pixel-wise F_1^* , demonstrating the importance of evaluating at component level. The results w.r.t. the different categories of methods are challenging for models, hence leaving room for improvement.

Our benchmark suite enables a unified evaluation across different datasets whenever ground truth is available.

In Table 8.3 we summarize our results for the LostAndFound test split, with original labels fitting the obstacle track. We filtered out all images that contain humans and bicycles labeled as obstacles (therefore called LostAndFound test-NoKnown) because we applied anomaly segmentation methods out of the box to the task of obstacle segmentation, and these methods focus on previously-unseen objects.

In comparison to our datasets, for the LostAndFound dataset we observe a less pronounced gap, in terms of both main performance metrics, the pixel-level AuPRC and component-level $\overline{F_1}$ scores, between the methods originally designed for image classification, especially ODIN and Mahalanobis, and those specifically designed for anomaly segmentation, especially road inpainting and maximized entropy. This signals that both of our datasets contribute new challenges for anomaly and obstacle segmentation.

Finally, we also applied our benchmark suite to the LiDAR guided Small obstacle Segmentation dataset [SKGMK20]. Our main findings are that our whole set of methods yields weak

8.3 Extra evaluations

		Pixel-level				Component-level											
								$\tau = 0.25$			$\tau = 0.50$			$\tau = 0.75$			
Method	requires OoD data	Anomaly scores				$k \in \mathcal{K}$	$\hat{k} \in \hat{\mathcal{K}}$										
		AuPRC \uparrow	FPR ₉₅ \downarrow	F_1^* \uparrow		sIoU \uparrow	PPV \uparrow	FN \downarrow	FP \downarrow	F_1 \uparrow	FN \downarrow	FP \downarrow	F_1 \uparrow	FN \downarrow	FP \downarrow	F_1 \uparrow	$\overline{F_1}$ \uparrow
Maximum softmax [HG17]	\times	0.7	57.1	2.2		0.5	1.5	1196	1652	0.5	1202	1653	0.1	1203	1653	0.0	0.2
ODIN [LLS18]	\times	1.7	51.7	5.7		2.7	3.9	1151	1829	3.4	1176	1834	1.8	1197	1841	0.4	1.9
Mahalanobis [LLLS18b]	\times	1.4	45.5	2.4		7.1	4.0	1039	4863	5.3	1137	4882	2.1	1198	4907	0.2	2.4
MC dropout [MG18b]	\times	0.5	82.2	2.1		0.5	2.8	1191	1406	0.9	1200	1407	0.2	1203	1408	0.0	0.3
Void classifier [BSN ⁺ 19]	\checkmark	0.8	59.6	2.1		1.5	4.9	1169	813	3.3	1193	816	1.0	1200	819	0.3	1.5
Embedding density [BSN ⁺ 19]	\times	0.5	66.0	1.1		9.8	1.8	1010	12421	2.8	1122	12502	1.2	1200	12587	0.0	1.3
SynBoost [DBBSC21]	\checkmark	12.5	62.8	22.8		11.5	14.4	1009	1204	14.9	1040	1217	12.6	1116	1234	6.9	12.0
Maximized entropy [CRG21b]	\checkmark	4.9	63.1	11.6		2.0	9.7	1159	586	4.8	1184	586	2.1	1202	586	0.1	2.4

Table 8.4: Benchmark results for the LiDAR guided Small obstacle Segmentation dataset. This dataset contains 1203 ground truth components in total.

performance on that dataset. The main purpose of this dataset is the detection of small obstacles from multiple sensors including LiDAR. Hence, the conditions for the other sensor modalities are purposely challenging (*e.g.*, low illumination), making this dataset less suitable to camera-only methods. We present the corresponding results in [Section 8.3.1](#).

Through the course of the project our methods have pushed the state of the art in obstacle and anomaly segmentation and we are glad that further work has continued the impressive progress in the field.

8.3 Extra evaluations

Here we present several additional evaluations performed for the original benchmark. We don't perform them for all the new submitted methods but they still offer interesting insights.

8.3.1 LiDAR Guided Small Obstacle Dataset

The results corresponding to the LiDAR guided Small obstacle Segmentation dataset [SKGMK20] are given in [Table 8.4](#). In general, the given set of methods exhibits poor performance on this dataset. As discussed in [Section 3.4.3](#), the dataset is designed for multi-sensor detection and very challenging in a pure-vision setting. More precisely, obstacles are mostly overlooked, even SynBoost as best-performing method still misses 1100 of 1203 components in total at the lowest sIoU threshold $\tau = 0.25$. This dataset can easily be included into our benchmark and it also fits the obstacle track, however, from our experiments we conclude that this dataset is less suitable to camera-only obstacle segmentation as obstacles are not well captured via cameras.

8.3.2 Evaluation per Environment Category

We already emphasized that in our RoadObstacle21 dataset a wide variety of road surfaces are available, representing different scenes which might pose unique challenges. In this section, we provide more insights by evaluating our set of methods on each of these surfaces. In total,

Chapter 8. Benchmark Results



Figure 8.2: The scenes of our RoadObstacle21 dataset feature a variety of road surfaces.

we split our datasets into 9 different scenes, shown in [Figure 8.2](#):

1. cracked road, surrounded by snow (road cracked)
2. dark asphalt after rain, with leaves (asphalt dark)
3. gravel road, no snow (road gravel)
4. gray asphalt in village and forest (asphalt gray)
5. motorway with side railing (motorway)
6. sun reflection off wet road (sun reflection)
7. road made of bricks (road bricks)
8. night images (asphalt night)
9. and snowstorm images .

We evaluate each subset using our benchmark suite and report the results in [Table 8.5](#). This more detailed evaluation shows that the reported set of methods perform differently across the data splits, with no method having consistent performance on each of these subsets. Our

8.3 Extra evaluations

		road cracked		asphalt dark		road gravel		asphalt gray		motorway		sun reflection		road bricks	
	OoD	$N = 40$		$N = 47$		$N = 33$		$N = 66$		$N = 30$		$N = 72$		$N = 39$	
Method	data	AuPRC	$\overline{F_1}$	AuPRC	$\overline{F_1}$	AuPRC	$\overline{F_1}$	AuPRC	$\overline{F_1}$	AuPRC	$\overline{F_1}$	AuPRC	$\overline{F_1}$	AuPRC	$\overline{F_1}$
Maximum softmax [HG17]	✗	11.7	3.2	69.3	25.7	39.5	21.0	43.4	14.9	4.8	0.8	2.1	4.4	32.7	26.8
ODIN [LLS18]	✗	14.9	4.8	74.8	30.8	65.3	37.0	73.8	22.5	9.9	7.2	2.8	5.4	48.8	22.0
Mahalanobis [LLS18b]	✗	25.9	1.6	46.7	18.3	65.8	21.9	84.7	53.8	61.2	35.6	13.9	0.5	83.6	41.0
MC dropout [MG18b]	✗	6.5	1.0	21.7	4.9	27.9	5.5	11.4	1.7	0.6	0.0	0.2	0.2	18.5	3.6
Ensemble [LPB17]	✗	34.3	0.0	5.6	0.8	33.4	0.0	17.3	12.3	1.2	4.3	0.2	0.0	17.6	0.6
Void classifier [BSN ⁺ 19]	✓	15.9	6.4	35.0	15.4	6.3	3.1	38.2	11.0	18.7	8.4	10.7	0.6	13.4	10.1
Embedding density [BSN ⁺ 19]	✗	2.5	0.8	3.3	0.8	2.7	2.2	1.8	2.4	1.1	3.0	0.1	1.1	18.3	2.7
Image resynthesis [LNSF19]	✗	48.2	9.6	42.0	12.3	77.0	42.4	66.6	22.2	23.7	12.9	34.4	9.0	12.1	2.4
Road inpainting [LHFS20b]	✗	21.0	18.4	77.0	47.2	88.4	74.7	93.5	79.8	83.1	78.1	29.4	22.0	93.5	73.7
SynBoost [DBBSC21]	✓	46.1	14.7	89.3	66.5	84.7	54.5	81.2	54.0	53.8	48.8	43.1	25.4	89.8	70.3
Maximized entropy [CRG21b]	✓	77.1	42.5	96.9	71.9	98.6	88.7	94.8	70.2	64.3	35.1	43.2	30.6	93.9	61.0

		asphalt night		snowstorm	
	OoD	$N = 30$		$N = 55$	
Method	data	AuPRC	$\overline{F_1}$	AuPRC	$\overline{F_1}$
Maximum softmax [HG17]	✗	6.0	2.5	1.6	0.8
ODIN [LLS18]	✗	8.0	1.8	6.7	4.6
Mahalanobis [LLS18b]	✗	14.2	5.5	21.2	13.2
MC dropout [MG18b]	✗	4.2	1.1	0.5	0.6
Ensemble [LPB17]	✗	11.5	16.9	0.6	0.0
Void classifier [BSN ⁺ 19]	✓	5.9	5.5	3.0	5.1
Embedding density [BSN ⁺ 19]	✗	16.7	3.6	0.9	2.6
Image resynthesis [LNSF19]	✗	16.5	6.3	19.2	4.0
Road inpainting [LHFS20b]	✗	51.2	28.0	55.3	35.0
SynBoost [DBBSC21]	✓	14.5	10.2	46.4	20.7
Maximized entropy [CRG21b]	✓	41.0	12.1	30.5	17.5

Table 8.5: Effect of different of scenes in the RoadObstacle21 dataset. Here, N denotes the number of images in a subset. As main evaluation metrics we consider the pixel-wise AuPRC and the component-wise $\overline{F_1}$.

dataset offers extra difficulty caused by the diversity of road texture, surrounding environments, weather and lighting variations. Cracks and leaves may trigger false positives, and a gravel or wet road surface may itself be sufficiently different from training images to be mistaken for an anomaly.

9 Conclusion

Our research offers an extensive study of the topic of anomaly and obstacle detection in road scenes for the purpose of improving self-driving safety. We were one of the pioneers in the field as before the start of this project anomaly detection was primarily focused on classifying whole images rather than segmenting anomalies within the scene composed of known objects.

We define anomalies in semantic segmentation as objects outside of the predefined semantic classes used in training. This is a uniquely challenging topic since most the deep learning methods most successful in computer vision are powered by labeled training examples. Motivated by the practical needs of the industry we then explored unusual obstacles - any objects on the road which might cause a collision, with the added challenge of lack of training examples of these particular items.

In this document we have presented several solutions to these problems. One line of thought is to reconstruct the input image in a way which removes the anomalies and obstacles. To this end we applied an information bottleneck of a semantic map (Section 4.1), or inpainted the road surface (Section 5.1.2). The anomalies can be then found by comparing the input and reconstructed image. The comparison can be performed by a discrepancy network.

Another idea pertains to training anomaly and obstacle detectors in the absence of training examples. To this end, we have devised synthetic training schemes where anomalies are introduced into the usual urban scenes by altering object class labels (Section 4.1.2) or by injecting object cut-outs onto the road (Section 5.1.3). Taking perspective warping into account can further improve the synthetic generation process (Section 6.2.2).

We have also explored the valuable information that happens to be encoded by transformer self-attention maps which can be used to segment small objects in a variety of domains even though the networks were never trained to perform that task (Chapter 7).

But perhaps the most lasting impact is the introduction of the anomaly and obstacle benchmark (Chapter 3) which has inspired a wealth of quality research and has measured the impressive progress made in the span of just a few years (Section 8.2).

9.1 Future Directions

We have explored several interesting ideas but there are many more to try. Likewise the recent works published in the field inspire further investigation.

Automated anomaly and obstacle example gathering

To collect the anomaly and obstacle examples for the benchmark described in [Chapter 3](#), we sought out published photos of unusual objects on the road, or placed obstacles and photographed them ourselves. Examples of real obstacles found while driving are still rare, even though collisions with animals or debris are a cause of traffic accidents. The next step to finding impactful data in that domain could be to detect obstacles automatically within a vast amount of video captured by vehicles. A system capable of segmenting new objects in new domains, such as the one we proposed in [Chapter 7](#), could reveal obstacles which would later be integrated into testing sets for self-driving perception. The distribution of objects captured this way would have the benefit of being aligned with what dangers are most common in real situations.

However indiscriminate recording and storage of video would pose a threat to privacy and risk aiding surveillance. To avoid that, a detection system could run onboard the vehicle and only capture images where anomalies and obstacles are present. Using a detector operating only on the road surface ([Chapters 5 and 6](#)) could also help by ensuring the capture only happens when the objects are relevant the driving area.

Perspective-aware visual transformer

A visual transformer backbone such as ViT [[DBK⁺20](#)] differs significantly from a traditional convolutional neural network in that the first step involves extracting patches from the image without overlap. These patches become tokens for the transformer architecture and an additional spatial embedding is added to retain the information on which part of the image they originate from.

In [Chapter 6](#) we have discussed the significant impact of perspective foreshortening on obstacle detection from vehicle forward cameras - a detector needs to be capable of processing obstacles at wildly different pixel sizes. The works of [[HSB⁺07](#), [HSB⁺09](#)] address this by extracting patches of different sizes (big patches nearby, small patches far away) and resizing them to a common dimension. Therefore the detector operates constantly at the same scale. This approach is difficult to apply to a CNN with its overlapping convolution kernels. However it would be a great fit for the ViT which does not need a grid of patches but can work with any sequence of them. Patches would be resized before passing them on into the transformer. The spatial embedding could contain further valuable perspective scale information.

Performance

The end goal of anomaly and obstacle detection in road scenes is deployment in vehicles for the purpose of enhancing safety, whether in the form of reliable self-driving or driver assistance. In a driving scenario decisions need to be made quickly, therefore computational cost is a key concern. Many of the current solutions rely on expensive operations: full-resolution semantic segmentation, resynthesis, inpainting, transformer self-attention, or even combine multiple thereof. Further work is needed to refine these methods to run in real time on onboard computational devices. It is worth exploring architectural changes and network distillation and quantization.

Mask-based segmentation

The results shown in [Section 8.2](#) indicate the extraordinary ability of the recent mask-based segmenters [[CSK21](#), [CMS⁺22](#), [HOLH21](#)] to distinguish anomalies from known categories [[NYHG23](#), [Mat23](#)]. And this is usually achieved by the semantic segmentation network itself without retraining or extra anomaly branches. Therefore we consider those to be the most promising direction in anomaly detection. In the future it would be worth investigating the reliability of these approaches, in particular in the presence of stronger domain shift such as difficult weather and lighting conditions, or a variety of different locations and landscapes. Finding where their limitations lie will inform the future direction of research in the field.

The nature of the query mechanism is promising for domain adaptation. Queries produce embeddings and taking the dot product of a query embedding with the pixel embedding yields the mask. It may be possible to adjust to domain shift by merely altering the query vectors and keeping most of the network unchanged.

It would also be interesting to see if the masks can reveal insights for which they were not specifically trained - analogously to how in [Chapter 7](#) transformer attention was capable of segmenting previously unseen objects in new domains. For example, Mask2Former [[CMS⁺22](#)] predicts 64 masks but only 19 become assigned to semantic classes in the case of Cityscapes training according to the analysis performed in [[NYHG23](#)]. The other masks could reveal valuable information as well.

Medical applications

While driving safety is important, health concerns affect everyone and so medical applications have unique value. The methods discussed here could be adapted to segmentation of anomalous areas in medical imaging where the training data is also often limited.

Chapter 9. Conclusion

Exploring failure cases

In this project I have learned the value of taking an active effort to seek out the failure cases of the system one is developing. This way we can know the capabilities and limitations of the system during deployment as well as focus the development efforts in the area where they matter most.

9.2 Summary

In this project we have explored the task of locating semantic anomalies and unusual obstacles within diverse road scenes using cameras and computer-vision techniques. While this topic is only recently gaining attention, it is important to the safety of self-driving and driving-assistance systems. We have proposed several novel detection methods involving information bottlenecks, synthetic training, and transformer self-attention. To address the scarcity of data inherent to the problem, we introduce a new dataset and protocol for benchmarking anomaly and obstacle detection methods.

Our findings bear significance in a broader context; when machine-learning methods are used in a safety-critical setting, they require testing against out-of-distribution samples and domain shift.

Bibliography

- [AAAB18] S. Akcay, A. Atapour-Abarghouei, and T. P. Breckon. Ganomaly: Semi-Supervised Anomaly Detection via Adversarial Training. In Asian Conference on Computer Vision, 2018. 10, 31
- [ACS19] Matt Angus, Krzysztof Czarnecki, and Rick Salay. Efficacy of pixel-level OOD detection for semantic segmentation. CoRR, abs/1911.02897, 2019. 8, 15
- [AMT22] Ioannis Athanasiadis, Georgios Moschovis, and Alexander Tuoma. Weakly-supervised semantic segmentation via transformer explainability. In ML Reproducibility Challenge 2021 (Fall Edition), 2022. 13
- [BBG⁺19] L. Berlincioni, F. Becattini, L. Galteri, L. Seidenari, and A. Del Bimbo. Road Layout Understanding by Generative Adversarial Inpainting. In Inpainting and Denoising Challenges, pages 111–128. 2019. 11
- [BBPA21] Victor Besnier, Andrei Bursuc, David Picard, and Briot Alexandre. Triggering Failures: Out-Of-Distribution detection by learning from local adversarial attacks in Semantic Segmentation. In International Conference on Computer Vision. 2021. 12, 99
- [BCR⁺20] Dominik Brüggemann, Robin Chan, Matthias Rottmann, Hanno Gottschalk, and Stefan Bracke. Detecting Out of Distribution Objects in Semantic Segmentation of Street Scenes. In The 30th European Safety and Reliability Conference (ESREL), 2020. 9, 15
- [BDW⁺21] Christoph Baur, Stefan Denner, Benedikt Wiestler, Nassir Navab, and Shadi Albarqouni. Autoencoders for unsupervised anomaly segmentation in brain mr images: A comparative study. Medical Image Analysis, 69:101952, 2021. 8
- [BEP13] Kendrick Boyd, Kevin H. Eng, and C. David Page. Area under the precision-recall curve: Point estimates and confidence intervals. In Hendrik Blockeel, Kristian Kersting, Siegfried Nijssen, and Filip Železný, editors, Machine Learning and Knowledge Discovery in Databases, pages 451–466, Berlin, Heidelberg, 2013. Springer Berlin Heidelberg. 26

Bibliography

- [BFSS19] Paul Bergmann, Michael Fauser, David Sattlegger, and Carsten Steger. MVTec AD—A comprehensive Real-World dataset for unsupervised anomaly detection. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pages 9592–9600, 2019. 8
- [BHL⁺19] S. Bai, Z. He, Y. Lei, W. Wu, C. Zhu, M. Sun, and J. Yan. Traffic Anomaly Detection via Perspective Map Based on Spatial-Temporal Information Matrix. In Conference on Computer Vision and Pattern Recognition, 2019. 65
- [BKC17a] Vijay Badrinarayanan, Alex Kendall, and Roberto Cipolla. Bayesian SegNet: Model Uncertainty in Deep Convolutional Encoder-Decoder Architectures for Scene Understanding. In Proceedings of the British Machine Vision Conference (BMVC), pages 57.1–57.12. BMVA Press, 9 2017. 8, 31, 36
- [BKC17b] Vijay Badrinarayanan, Alex Kendall, and Roberto Cipolla. Segnet: A Deep Convolutional Encoder-Decoder Architecture for Image Segmentation. IEEE Transactions on Pattern Analysis and Machine Intelligence, 2017. 33, 42
- [BKOS19] P. Bevandić, I. Kreso, M. Orsic, and S. Segvić. Simultaneous Semantic Segmentation and Outlier Detection in Presence of Domain Shift. In German Conference on Pattern Recognition, 2019. 11, 46, 54, 55, 56, 63, 68
- [BMPK19] Borja Bovcon, Jon Muhovič, Janez Perš, and Matej Kristan. The MaSTr1325 dataset for training deep USV obstacle detection models. In International Conference on Intelligent Robots and Systems, 2019. 79, 80, 85, 88, 89
- [BNSC19] B. Bescos, J. Neira, R. Siegwart, and C. Cadena. Empty Cities: Image Inpainting for a Dynamic-Object-Invariant Space. In International Conference on Robotics and Automation, 2019. 11
- [BRF18] A. Bhattad, J. Rock, and D. Forsyth. Detecting Anomalous Faces with ‘No Peeking’ Autoencoders. In Conference on Computer Vision and Pattern Recognition, 2018. 45
- [BSN⁺19] Hermann Blum, Paul-Edouard Sarlin, Juan Nieto, Roland Siegwart, and Cesar Cadena. Fishyscapes: A benchmark for safe semantic segmentation in autonomous driving. In International Conference on Computer Vision, 10 2019. 2, 8, 15, 17, 18, 21, 24, 46, 53, 54, 55, 56, 71, 87, 95, 98, 99, 100, 103, 105
- [CANK17] M. M. Cisse, Y. Adi, N. Neverova, and J. Keshet. Houdini: Fooling Deep Structured Visual and Speech Recognition Models with Adversarial Examples. In Advances in Neural Information Processing Systems, pages 6977–6987, 2017. 33
- [CC17] Zhe Chen and Zijing Chen. RBNet: A Deep Neural Network for Unified Road and Road Boundary Detection. In International Conference on Neural Information Processing, 2017. 1

- [CCR⁺17] Florian Chabot, Mohamed Chaouch, Jaonary Rabarisoa, Céline Teuliere, and Thierry Chateau. Deep manta: A coarse-to-fine many-task network for joint 2d and 3d vehicle analysis from monocular image. In Conference on Computer Vision and Pattern Recognition, 2017. 1
- [CDI14] K. Chaudhury, S. Diverdi, and S. Ioffe. Auto-Rectification of User Photos. In International Conference on Image Processing, 2014. 67
- [CJA18] Hyunsun Choi, Eric Jang, and Alexander A Alemi. WAIC, but why? generative ensembles for robust anomaly detection, October 2018. 16
- [CKC20] S. Choi, J. T. Kim, and J. Choo. Cars Can’t Fly Up in the Sky: Improving Urban-Scene Segmentation via Height-Driven Attention Networks. In Conference on Computer Vision and Pattern Recognition, 2020. 65, 73
- [CLU⁺21] R. Chan, K. Lis, S. Uhlemeyer, H. Blum, S. Honari, R. Siegwart, P. Fua, M. Salzmann, and M. Rottmann. Segmentmeifyoucan: A Benchmark for Anomaly Segmentation. In Advances in Neural Information Processing Systems, 2021. 3
- [CLV08] A.B. Chan, Z.S.J. Liang, and N. Vasconcelos. Privacy Preserving Crowd Monitoring: Counting People Without People Models or Tracking. In Conference on Computer Vision and Pattern Recognition, 2008. 65
- [CM15] C. Creusot and A. Munawar. Real-Time Small Obstacle Detection on Highways Using Compressive RBM Road Reconstruction. In Intelligent Vehicles Symposium, 2015. 10, 12, 31, 33, 37, 38, 42
- [CMS⁺22] Bowen Cheng, Ishan Misra, Alexander G. Schwing, Alexander Kirillov, and Rohit Girdhar. Masked-attention Mask Transformer for Universal Image Segmentation. In Conference on Computer Vision and Pattern Recognition, 2022. 9, 102, 109
- [Con20] MMSegmentation Contributors. MMSegmentation: Openmmlab semantic segmentation toolbox and benchmark. <https://github.com/open-mmlab/mms Segmentation>, 2020. 84
- [COR⁺16] Marius Cordts, Mohamed Omran, Sebastian Ramos, Timo Rehfeld, Markus Enzweiler, Rodrigo Benenson, Uwe Franke, Stefan Roth, and Bernt Schiele. The Cityscapes Dataset for Semantic Urban Scene Understanding. In Conference on Computer Vision and Pattern Recognition, 2016. 1, 2, 7, 11, 15, 17, 21, 22, 26, 36, 42, 46, 47, 51, 68, 79, 84, 85, 98
- [CPI⁺18] L. Chen, G. Papandreou, I.Kokkinos, K. Murphy, and A. L. Yuille. Deeplab: Semantic Image Segmentation with Deep Convolutional Nets, Atrous Convolution, and Fully Connected CRFs. IEEE Transactions on Pattern Analysis and Machine Intelligence, 2018. 53

Bibliography

- [CPSA17] L.-C. Chen, G. Papandreou, F. Schroff, and H. Adam. Rethinking Atrous Convolution for Semantic Image Segmentation. In arXiv Preprint, 2017. 1, 7
- [CRG21a] R. Chan, M. Rottmann, and H. Gottschalk. Entropy Maximization and Meta Classification for Out-Of-Distribution Detection in Semantic Segmentation. In International Conference on Computer Vision, 2021. 75
- [CRG21b] Robin Chan, Matthias Rottmann, and Hanno Gottschalk. Entropy Maximization and Meta Classification for Out-Of-Distribution Detection in Semantic Segmentation. In International Conference on Computer Vision, pages 5128–5137, 10 2021. 9, 12, 15, 54, 55, 56, 58, 63, 71, 72, 74, 75, 87, 95, 98, 99, 100, 103, 105
- [CSK21] Bowen Cheng, Alex Schwing, and Alexander Kirillov. Per-pixel classification is not all you need for semantic segmentation. In Advances in Neural Information Processing Systems, volume 34, pages 17864–17875, 2021. 9, 79, 102, 109
- [CTM⁺21] Mathilde Caron, Hugo Touvron, Ishan Misra, Hervé Jégou, Julien Mairal, Piotr Bojanowski, and Armand Joulin. Emerging properties in self-supervised vision transformers. In International Conference on Computer Vision, pages 9650–9660, 2021. 13, 80, 87, 88, 92, 93
- [CZP⁺18] L.-C. Chen, Y. Zhu, G. Papandreou, F. Schroff, and H. Adam. Encoder-Decoder with Atrous Separable Convolution for Semantic Image Segmentation. In European Conference on Computer Vision, 2018. 7
- [CZX21] Wenjie Chang, Yueyi Zhang, and Zhiwei Xiong. Transformer-based monocular depth estimation with attention supervision. In British Machine Vision Conference, 2021. 79
- [DBBSC21] Giancarlo Di Biase, Hermann Blum, Roland Siegwart, and Cesar Cadena. Pixel-Wise Anomaly Detection in Complex Driving Scenes. In Conference on Computer Vision and Pattern Recognition, pages 16918–16927, June 2021. 11, 50, 54, 55, 58, 63, 71, 87, 88, 95, 99, 100, 103, 105
- [DBK⁺20] 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. In arXiv Preprint, 2020. 5, 81, 82, 84, 93, 108
- [DDS⁺09] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei. Imagenet: A Large-Scale Hierarchical Image Database. In Conference on Computer Vision and Pattern Recognition, 2009. 11, 13, 51, 54, 69
- [DL91] J. S. Denker and Y. LeCun. Transforming Neural-Net Output Levels to Probability Distributions. In Advances in Neural Information Processing Systems, 1991. 8

-
- [DSDB17] L. Dinh, J. Sohl-Dickstein, and S. Bengio. Density Estimation Using Real NVP. In Advances in Neural Information Processing Systems, 2017. 12
 - [DT05] N. Dalal and B. Triggs. Histograms of Oriented Gradients for Human Detection. In Conference on Computer Vision and Pattern Recognition, pages 886–893, 2005. 35
 - [DT18] Terrance DeVries and Graham W. Taylor. Learning Confidence for Out-of-Distribution Detection in Neural Networks, 2 2018. 97
 - [DWSP12] P. Dollár, C. Wojek, B. Schiele, and P. Perona. Pedestrian Detection: An Evaluation of the State of the Art. IEEE Transactions on Pattern Analysis and Machine Intelligence, 34(4):743–761, 2012. 1
 - [FA20] Jeff Faudi and Airbus Defense and Space Intelligence. Airbus Aircrafts Detection Sample Dataset. <https://www.kaggle.com/datasets/airbusgeo/airbus-aircrafts-sample-dataset>, 2020. 79, 80, 85, 88, 90
 - [GBŠ21a] Matej Grcić, Petra Bevandić, and Siniša Šegvić. Dense anomaly detection by robust learning on synthetic negative data. In arXiv Preprint, 2021. 12, 99, 100
 - [GBS21b] Matej Grcic, Petra Bevandic, and Sinisa Segvic. Dense Open-set Recognition with Synthetic Outliers Generated by Real NVP. In VISIGRAPP (4: VISAPP), 2021. 12
 - [GBŠ22] Matej Grcić, Petra Bevandić, and Siniša Šegvić. DenseHybrid: Hybrid Anomaly Detection for Dense Open-Set Recognition. In European Conference on Computer Vision, 2022. 12, 71, 72, 87, 99, 100
 - [GG16] Yarín Gal and Zoubin Ghahramani. Dropout as a Bayesian Approximation: Representing Model Uncertainty in Deep Learning. In Proceedings of The 33rd International Conference on Machine Learning, volume 48 of Proceedings of Machine Learning Research, pages 1050–1059, New York, New York, USA, 6 2016. PMLR. 8
 - [GHH⁺20] Jian Guo, He He, Tong He, Leonard Lausen, Mu Li, Haibin Lin, Xingjian Shi, Chenguang Wang, Junyuan Xie, Sheng Zha, Aston Zhang, Hang Zhang, Zhi Zhang, Zhongyue Zhang, Shuai Zheng, and Yi Zhu. Gluoncv and gluonnlp: Deep learning in computer vision and natural language processing. Journal of Machine Learning Research, 21(23):1–7, 2020. 47
 - [GJFF⁺18] A. Gupta, J. Johnson, L. Fei-Fei, S. Savarese, and A. Alahi. Social GAN: Socially Acceptable Trajectories with Generative Adversarial Networks. In Conference on Computer Vision and Pattern Recognition, 2018. 12
 - [GJGK18] Krishnam Gupta, Syed Ashar Javed, Vineet Gandhi, and K Madhava Krishna. MergeNet: A Deep Net Architecture for Small Obstacle Discovery. In International Conference on Robotics and Automation, 2018. 66

Bibliography

- [GKM⁺19] Jakob Geyer, Yohannes Kassahun, Mentar Mahmudi, Xavier Ricou, Rupesh Durgesh, Andrew S. Chung, Lorenz Hauswald, Viet Hoang Pham, Maximilian Mühlegg, Sebastian Dorn, Tiffany Fernandez, Martin Jänicke, Sudesh Mirashi, Chiragkumar Savani, Martin Sturm, Oleksandr Vorobiov, Martin Oelker, Sebastian Garreis, and Peter Schuberth. A2D2: AEV Autonomous Driving Dataset. <http://www.a2d2.audi>, 2019. 15
- [GNJ⁺22] Akshita Gupta, Sanath Narayan, KJ Joseph, Salman Khan, Fahad Shahbaz Khan, and Mubarak Shah. Ow-detr: Open-world detection transformer. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pages 9235–9244, 2022. 13
- [GR18] J. Gast and S. Roth. Lightweight Probabilistic Deep Networks. In Conference on Computer Vision and Pattern Recognition, 2018. 8, 31
- [HAB19] Matthias Hein, Maksym Andriushchenko, and Julian Bitterwolf. Why relu networks yield high-confidence predictions far away from the training data and how to mitigate the problem. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 6 2019. 8, 15
- [HBM⁺22] Dan Hendrycks, Steven Basart, Mantas Mazeika, Andy Zou, Joe Kwon, Mohammadreza Mostajabi, Jacob Steinhardt, and Dawn Song. Scaling Out-of-Distribution Detection for Real-World Settings. In International Conference on Machine Learning, 2022. 15, 17, 18, 21, 26
- [HCG⁺18] X. Huang, X. Cheng, Q. Geng, B. Cao, D. Zhou, P. Wang, Y. Lin, and R. Yang. The Apolloscape Dataset for Autonomous Driving. In Conference on Computer Vision and Pattern Recognition, 2018. 7
- [HG17] Dan Hendrycks and Kevin Gimpel. A Baseline for Detecting Misclassified and Out-of-Distribution Examples in Neural Networks. In 5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings, 2017. 8, 9, 16, 71, 87, 88, 95, 96, 99, 100, 103, 105
- [HGDG17] K. He, G. Gkioxari, P. Dollár, and R. B. Girshick. Mask R-CNN. In International Conference on Computer Vision, 2017. 1
- [HGT18] M. Haselmann, D. P. Gruber, and P. Tabatabai. Anomaly Detection Using Deep Learning Based Image Completion. In International Conference on Machine Learning, 2018. 11, 45
- [HMD19] Dan Hendrycks, Mantas Mazeika, and Thomas Dietterich. Deep anomaly detection with outlier exposure. Proceedings of the International Conference on Learning Representations, 2019. 98

- [HOLH21] Jaedong Hwang, Seoung Wug Oh, Joon-Young Lee, and Bohyung Han. Exemplar-based open-set panoptic segmentation network. In Conference on Computer Vision and Pattern Recognition, pages 1175–1184, 2021. 9, 102, 109
- [HSB⁺07] Raia Hadsell, Pierre Sermanet, Jan Ben, A Erkan, Jeff Han, Beat Flepp, Urs Muller, and Yann LeCun. Online Learning for Offroad Robots: Using Spatial Label Propagation to Learn Long-Range Traversability. In Robotics: Science and Systems Conference, volume 11, page 32, 2007. 64, 108
- [HSB⁺09] Raia Hadsell, Pierre Sermanet, Jan Ben, Ayse Erkan, Marco Scoffier, Koray Kavukcuoglu, Urs Muller, and Yann LeCun. Learning Long-Range Vision for Autonomous Off-Road Driving. Journal of Field Robotics, 26(2):120–144, 2009. 64, 108
- [HTLH21] C. Huynh, A. Tran, K. Luu, and M. Hoai. Progressive Semantic Segmentation. In Conference on Computer Vision and Pattern Recognition, 2021. 65
- [HZH⁺22] Mark Hamilton, Zhoutong Zhang, Bharath Hariharan, Noah Snavely, and William T. Freeman. Unsupervised semantic segmentation by distilling feature correspondences. In International Conference on Learning Representations, 2022. 13
- [IA17a] Shuya Isobe and Shuichi Arai. A Semantic Segmentation Method Using Model Uncertainty. In IJAE International Conference on Intelligent Systems and Image Processing, 2017. 8
- [IA17b] Shuya Isobe and Shuichi Arai. Deep Convolutional Encoder-Decoder Network with Model Uncertainty for Semantic Segmentation. In IEEE International Conference on INnovations in Intelligent SysTems and Applications, 2017. 8, 15
- [IA17c] Shuya Isobe and Shuichi Arai. Inference with Model Uncertainty on Indoor Scene for Semantic Segmentation. In IEEE Global Conference on Signal and Information Processing, 2017. 8
- [IZZE16] P. Isola, J. Zhu, T. Zhou, and A. Efros. Image-To-Image Translation with Conditional Adversarial Networks. In arXiv Preprint, 2016. 10
- [JLG⁺21] Sanghun Jung, Jungsoo Lee, Daehoon Gwak, Sungha Choi, and Jaegul Choo. Standardized max logits: A simple yet effective approach for identifying unexpected road obstacles in urban-scene segmentation. In International Conference on Computer Vision, pages 15425–15434, 2021. 9
- [JRF20] Nicolas Jourdan, Eike Rehder, and Uwe Franke. Identification of uncertainty in artificial neural networks. In Proceedings of the 13th Uni-DAS e.V. Workshop Fahrerassistenz und automatisiertes Fahren, 7 2020. 15, 98

Bibliography

- [KB15] D. P. Kingma and J. Ba. Adam: A Method for Stochastic Optimisation. In International Conference on Learning Representations, 2015. 42, 59, 78
- [KG17] A. Kendall and Y. Gal. What Uncertainties Do We Need in Bayesian Deep Learning for Computer Vision? In Advances in Neural Information Processing Systems, 2017. 8, 31
- [KHG⁺19] A. Kirillov, K. He, R. Girshick, C. Rother, and P. Dollár. Panoptic Segmentation. In Conference on Computer Vision and Pattern Recognition, 2019. 1
- [KTP18] B. R. Kiran, D. M. Thomas, and R. Parakkal. An Overview of Deep Learning Based Methods for Unsupervised and Semi-Supervised Anomaly Detection in Videos. Journal of Imaging, 2018. 10
- [KUMH17] G. Klambauer, T. Unterthiner, A. Mayr, and S. Hochreiter. Self-Normalizing Neural Networks. In Advances in Neural Information Processing Systems, 2017. 42, 59
- [LBH18] M. Liu, W. Buntine, and G. Haffari. Learning How to Actively Learn: A Deep Imitation Learning Approach. In Annual Meeting of the Association for Computational Linguistics, 2018. 65, 73
- [LH18] Y. Lyu and X. Huang. Road Segmentation Using CNN with GRU. In arXiv Preprint, 2018. 65
- [LHFS20a] Krzysztof Lis, Sina Honari, Pascal Fua, and Mathieu Salzmann. Detecting Road Obstacles by Erasing Them. In arXiv Preprint, 2020. 75, 78
- [LHFS20b] Krzysztof Lis, Sina Honari, Pascal Fua, and Mathieu Salzmann. Detecting Road Obstacles by Erasing Them, 2020. 105
- [LHFS23] Krzysztof Lis, Sina Honari, Pascal Fua, and Mathieu Salzmann. Perspective Aware Road Obstacle Detection. In IEEE Robotics and Automation Letters, 2023. In Press. 4
- [LJR18] W. Li, O. H. Jafari, and C. Rother. Deep Object Co-Segmentation. In Asian Conference on Computer Vision, 2018. 34, 42, 59
- [LJW⁺17] X. Li, Z. Jie, W. Wang, C. Liu, J. Yang, X. Shen, Z. Lin, Q. Chen, S. Yan, and J. Feng. Foveanet: Perspective-Aware Urban Scene Parsing. In Conference on Computer Vision and Pattern Recognition, 2017. 65
- [LLLS18a] K. Lee, H. Lee, K. Lee, and J. Shin. Training Confidence-Calibrated Classifiers for Detecting Out-Of-Distribution Samples. In International Conference on Learning Representations, 2018. 53, 54, 55
- [LLLS18b] Kimin Lee, Kibok Lee, Honglak Lee, and Jinwoo Shin. A Simple Unified Framework for Detecting Out-of-Distribution Samples and Adversarial Attacks. In S Bengio, H Wallach, H Larochelle, K Grauman, N Cesa-Bianchi, and R Garnett, editors,

- Advances in Neural Information Processing Systems, volume 31, pages 7167–7177. Curran Associates, Inc., 2018. 8, 16, 71, 87, 88, 95, 96, 99, 100, 103, 105
- [LLS18] Shiyu Liang, Yixuan Li, and R. Srikant. Enhancing The Reliability of Out-of-distribution Image Detection in Neural Networks. In International Conference on Learning Representations, 2018. 8, 9, 16, 71, 87, 88, 95, 96, 99, 100, 103, 105
- [LLSF19] W. Liu, K. Lis, M. Salzmann, and P. Fua. Geometric and Physical Constraints for Drone-Based Head Plane Crowd Density Estimation. International Conference on Intelligent Robots and Systems, 2019. 65, 69, 71, 73
- [LMB⁺14] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. Microsoft COCO: Common Objects in Context. In David Fleet, Tomas Pajdla, Bernt Schiele, and Tinne Tuytelaars, editors, Computer Vision – ECCV 2014, pages 740–755. Springer International Publishing, 2014. 12, 15, 54, 72, 98, 101
- [LNSF19] K. Lis, K. Nakka, M. Salzmann, and P. Fua. Detecting the Unexpected via Image Resynthesis. In International Conference on Computer Vision, 2019. 2, 4, 33, 35, 105
- [LPB17] B. Lakshminarayanan, A. Pritzel, and C. Blundell. Simple and Scalable Predictive Uncertainty Estimation Using Deep Ensembles. In Advances in Neural Information Processing Systems, 2017. 8, 31, 37, 71, 87, 95, 97, 99, 100, 105
- [LSF19] W. Liu, M. Salzmann, and P. Fua. Context-Aware Crowd Counting. In Conference on Computer Vision and Pattern Recognition, 2019. 65, 71, 73
- [LST15] Brenden M Lake, Ruslan Salakhutdinov, and Joshua B Tenenbaum. Human-level concept learning through probabilistic program induction. Science, pages 1332–1338, 2015. 16
- [LYLX21] Ruijin Liu, Zejian Yuan, Tie Liu, and Zhiliang Xiong. End-to-end lane shape prediction with transformers. In IEEE Winter Conference on Applications of Computer Vision, pages 3694–3702, 2021. 1
- [Mac92] D. J. MacKay. A Practical Bayesian Framework for Backpropagation Networks. Neural Computation, 4(3):448–472, 1992. 8
- [Mac95] D. J. Mackay. Bayesian Neural Networks and Density Networks. Nuclear Instruments and Methods in Physics Research Section A: Accelerators, Spectrometers, D 354(1):73–80, 1995. 8
- [Mat23] Matej Grcić and Josip Šarić and Siniša Šegvić. On advantages of mask-level recognition for outlier-aware segmentation. In CVPR 2023 workshop on Visual Anomaly and Novelty Detection (VAND), 2023. 9, 99, 100, 101, 102, 109

Bibliography

- [MBFP⁺17] Jesús Munoz-Bulnes, Carlos Fernandez, Ignacio Parra, David Fernández-Llorca, and Miguel A Sotelo. Deep fully convolutional networks with random data augmentation for enhanced generalization in road detection. In IEEE International Conference on Intelligent Transportation Systems (ITSC), 2017. 1
- [MC15] A. Munawar and C. Creusot. Structural Inpainting of Road Patches for Anomaly Detection. In IAPR International Conference on Machine Vision Applications, 2015. 10, 11, 12
- [MG18a] A. Malinin and M. Gales. Predictive Uncertainty Estimation via Prior Networks. In Advances in Neural Information Processing Systems, 2018. 8, 46, 54, 55
- [MG18b] Jishnu Mukhoti and Yarin Gal. Evaluating Bayesian Deep Learning Methods for Semantic Segmentation. In arXiv Preprint, 2018. 54, 55, 58, 71, 87, 95, 97, 99, 100, 103, 105
- [MH20] Alexander Meinke and Matthias Hein. Towards neural networks that provably know when they don't know. In International Conference on Learning Representations, 2020. 8, 16
- [MVD17] A. Munawar, P. Vinayavekhin, and G. De Magistris. Limiting the Reconstruction Capability of Generative Neural Network Using Negative Learning. In IEEE International Workshop on Machine Learning for Signal Processing, 2017. 10, 12, 31
- [MWT⁺20] A. Mehrtash, W. M. Wells, C. M. Tempany, P. Abolmaesumi, and T. Kapur. Confidence calibration and predictive uncertainty estimation for deep medical image segmentation. IEEE Transactions on Medical Imaging, pages 1–1, 2020. 15
- [NORK17] G. Neuhold, T. Ollmann, S. Rota Bulò, and P. Kotschieder. The Mapillary Vistas Dataset for Semantic Understanding of Street Scenes. In Conference on Computer Vision and Pattern Recognition, 2017. 11, 54
- [NYHG23] Nazir Nayal, Mısrâ Yavuz, João F Henriques, and Fatma Güney. RbA: Segmenting Unknown Regions Rejected by All. In arXiv Preprint, 2023. 9, 99, 100, 101, 102, 109
- [ORF20] Philipp Oberdiek, Matthias Rottmann, and Gernot A. Fink. Detection and retrieval of out-of-distribution objects in semantic segmentation. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) Workshops, June 2020. 9, 15
- [PAK19] C. D. Prakash, F. Akhbari, and L. J. Karam. Robust Obstacle Detection for Advanced Driver Assistance Systems Using Distortions of Inverse Perspective Mapping of a Monocular Camera. Robotics and Autonomous Systems, 114:172–186, 2019. 65, 71, 73

- [PCKC16] A. Paszke, A. Chaurasia, S. Kim, and E. Culurciello. Enet: A Deep Neural Network Architecture for Real-Time Semantic Segmentation. In arXiv Preprint, 2016. 42
- [PGM⁺19] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. PyTorch: An Imperative Style, High-Performance Deep Learning Library. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, editors, Advances in Neural Information Processing Systems, pages 8024–8035. 2019. 59, 78
- [PJI19] Romain Pessia, Quentin Jodelet, and Genya Ishigami. Artificial lunar landscape dataset. <https://www.kaggle.com/datasets/romainpessia/artificial-lunar-rocky-landscape-dataset>, 2019. 79, 80, 85, 88, 90
- [PRG⁺16] P. Pinggera, S. Ramos, S. Gehrig, U. Franke, C. Rother, and R. Mester. Lost and Found: Detecting Small Road Hazards for Self-Driving Vehicles. In International Conference on Intelligent Robots and Systems, 2016. 7, 12, 17, 18, 24, 32, 33, 37, 38, 45, 46, 53, 54, 64, 70, 86, 89
- [Pyt] ResNeXt implementation in PyTorch Vision. https://pytorch.org/vision/stable/_modules/torchvision/models/resnet.html#resnext101_32x8d. version 0.10.0. 59
- [QWL20] Z. Qin, H. Wang, and X. Li. Ultra Fast Structure-Aware Deep Lane Detection. In European Conference on Computer Vision, 2020. 1
- [RABA17] E. Romera, J. M. Alvarez, L. M. Bergasa, and R. Arroyo. ERFNet: Efficient Residual Factorized Convnet for Real-Time Semantic Segmentation. IEEE Transactions on Intelligent Transportation Systems, 2017. 1
- [RCH⁺20] Matthias Rottmann, Pascal Colling, Thomas Paul Hack, Robin Chan, Fabian Hüger, Peter Schlicht, and Hanno Gottschalk. Prediction error meta classification in semantic segmentation: Detection via aggregated dispersion measures of softmax probabilities. In 2020 IEEE International Joint Conference on Neural Networks (IJCNN), 2020. 9, 27
- [RGP⁺17] S. Ramos, S. Gehrig, P. Pinggera, U. Franke, and C. Rother. Detecting Unexpected Obstacles for Self-Driving Cars: Fusing Deep Learning and Geometric Modeling. In IEEE Intelligent Vehicles Symposium, 2017. 12, 66
- [RHGS15] S. Ren, K. He, R. Girshick, and J. Sun. Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks. In Advances in Neural Information Processing Systems, 2015. 1

Bibliography

- [RKB04] C. Rother, V. Kolmogorov, and A. Blake. "GrabCut" - Interactive Foreground Extraction Using Iterated Graph Cuts. In ACM SIGGRAPH, pages 309–314, 2004. 19
- [RNS⁺17] M. Ravanbakhsh, M. Nabi, E. Sangineto, L. Marcenaro, C. Regazzoni, and N. Sebe. Abnormal Event Detection in Videos Using Generative Adversarial Nets. In International Conference on Image Processing, 2017. 10
- [SGLS21] Robin Strudel, Ricardo Garcia, Ivan Laptev, and Cordelia Schmid. Segmenter: Transformer for semantic segmentation. In International Conference on Computer Vision, pages 7262–7272, 2021. 79, 80
- [SHK⁺14] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov. Dropout: A Simple Way to Prevent Neural Networks from Overfitting. Journal of Machine Learning Research, 15:1929–1958, 2014. 8
- [SKGK20] Aasheesh Singh, Aditya Kamireddypalli, Vineet Gandhi, and K Madhava Krishna. LiDAR guided Small Obstacle Segmentation. arXiv Preprint, 2020. 45
- [SKGMK20] Aasheesh Singh, Aditya Kamireddypalli, Vineet Gandhi, and K Madhava Krishna. LiDAR guided small obstacle segmentation, March 2020. 17, 18, 25, 102, 103
- [SOS⁺20] Philipp Seeböck, José Ignacio Orlando, Thomas Schlegl, Sebastian M. Waldstein, Hrvoje Bogunovic, Sophie Klimscha, Georg Langs, and Ursula Schmidt-Erfurth. Exploiting epistemic uncertainty of anatomy segmentation for anomaly detection in retinal OCT. IEEE Trans. Medical Imaging, 39(1):87–98, 2020. 8
- [SPV⁺21] Oriane Siméoni, Gilles Puy, Huy V. Vo, Simon Roburin, Spyros Gidaris, Andrei Bursuc, Patrick Pérez, Renaud Marlet, and Jean Ponce. Localizing objects with self-supervised transformers and no labels. November 2021. 13, 80, 87, 88
- [SSW⁺17] T. Schlegl, P. Seeböck, S. M. Waldstein, U. Schmidt-Erfurth, and G. Langs. Un-supervised Anomaly Detection with Generative Adversarial Networks to Guide Marker Discovery. In International Conference on Information Processing in Medical Imaging, 2017. 10
- [SYH⁺20] Lei Sun, Kailun Yang, Xinxin Hu, Weijian Hu, and Kaiwei Wang. Real-time fusion network for RGB-D semantic segmentation incorporating unexpected obstacle detection for road-driving images. IEEE Robotics and Automation Letters, 5(4):5558–5565, 2020. 66
- [SYXC19] M. Shi, Z. Yang, C. Xu, and Q. Chen. Revisiting Perspective Information for Efficient Crowd Counting. In Conference on Computer Vision and Pattern Recognition, 2019. 65
- [SZ15] K. Simonyan and A. Zisserman. Very Deep Convolutional Networks for Large-Scale Image Recognition. In International Conference on Learning Representations, 2015. 34, 42

-
- [TLP⁺22] Yu Tian, Yuyuan Liu, Guansong Pang, Fengbei Liu, Yuanhong Chen, and Gustavo Carneiro. Pixel-wise Energy-biased Abstention Learning for Anomaly Segmentation on Complex Urban Driving Scenes. In European Conference on Computer Vision, 2022. 99, 100
 - [UVL18] D. Ulyanov, A. Vedaldi, and V. Lempitsky. Deep Image Prior. In Conference on Computer Vision and Pattern Recognition, pages 9446–9454, 2018. 65
 - [vASTG20] Joost van Amersfoort, Lewis Smith, Yee Whye Teh, and Yarin Gal. Simple and scalable epistemic uncertainty estimation using a single deep deterministic neural network, March 2020. 16
 - [VM23] Tomáš Vojtř and Jiří Matas. Image-Consistent Detection of Road Anomalies As Unpredictable Patches. In IEEE Winter Conference on Applications of Computer Vision, pages 5491–5500, January 2023. 100
 - [VŠA⁺21] T. Vojtř, T. Šipka, R. Aljundi, N. Chumerin, D. O. Reino, and J. Matas. Road Anomaly Detection by Partial Image Reconstruction with Segmentation Coupling. In International Conference on Computer Vision, October 2021. 11, 63, 68, 71, 87, 99, 100
 - [WLZ⁺18] T.-C. Wang, M.-Y. Liu, J.-Y. Zhu, A. Tao, J. Kautz, and B. Catanzaro. High-Resolution Image Synthesis and Semantic Manipulation with Conditional GANs. In Conference on Computer Vision and Pattern Recognition, 2018. 10, 32, 33, 35, 58
 - [WSY⁺22] Yangtao Wang, Xi Shen, Yuan Yuan, Yuming Du, Maomao Li, Shell Xu Hu, James L. Crowley, and Dominique Vaufreydaz. Tokencut: Segmenting objects in images and videos with self-supervised transformer and normalized cut. arXiv preprint arXiv:2209.00383, 2022. 13, 80
 - [XGD⁺17] S. Xie, R. Girshick, P. Dollár, Z. Tu, and K. He. Aggregated Residual Transformations for Deep Neural Networks. In Conference on Computer Vision and Pattern Recognition, 2017. 51, 59, 69, 78
 - [XLZ⁺19] Y. Xiong, R. Liao, H. Zhao, R. Hu, M. Bai, E. Yumer, and R. Urtasun. UPSNet: A Unified Panoptic Segmentation Network. In Conference on Computer Vision and Pattern Recognition, 2019. 1
 - [XMZZ19] F. Xue, A. Ming, M. Zhou, and Y. Zhou. A Novel Multi-Layer Framework for Tiny Obstacle Discovery. In International Conference on Robotics and Automation, 2019. 12
 - [XWY⁺21] Enze Xie, Wenhai Wang, Zhiding Yu, Anima Anandkumar, Jose M Alvarez, and Ping Luo. SegFormer: Simple and efficient design for semantic segmentation with transformers. In Advances in Neural Information Processing Systems, 2021. 5, 79, 80, 81, 84, 85, 86, 93

Bibliography

- [XWZ⁺17] C. Xie, J. Wang, Z. Zhang, Y. Zhou, L. Xie, and A. Yuille. Adversarial Examples for Semantic Segmentation and Object Detection. In International Conference on Computer Vision, 2017. 33
- [XZL⁺20] Yingda Xia, Yi Zhang, Fengze Liu, Wei Shen, and Alan Yuille. Synthesize then compare: Detecting failures and anomalies for semantic segmentation. In Proceedings of the European Conference on Computer Vision (ECCV), 2020. 11, 50
- [YCW⁺20] Fisher Yu, Haofeng Chen, Xin Wang, Wenqi Xian, Yingying Chen, Fangchen Liu, Vashisht Madhavan, and Trevor Darrell. Bdd100k: A diverse driving dataset for heterogeneous multitask learning. In Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, pages 2636–2645, 2020. 15, 17, 26
- [YLW⁺20] Y. Yang, G. Li, Z. Wu, L. Su, Q. Huang, and N. Sebe. Reverse Perspective Network for Perspective-Aware Object Counting. In Conference on Computer Vision and Pattern Recognition, 2020. 65, 71, 73
- [YLY⁺19] J. Yu, Z. Lin, J. Yang, X. Shen, X. Lu, and T. S. Huang. Free-Form Image Inpainting with Gated Convolution. In Conference on Computer Vision and Pattern Recognition, 2019. 49
- [YWP⁺18] C. Yu, J. Wang, C. Peng, C. Gao, G. Yu, and N. Sang. Learning a Discriminative Feature Network for Semantic Segmentation. In Conference on Computer Vision and Pattern Recognition, 2018. 7
- [YXC⁺18] F. Yu, W. Xian, Y. Chen, F. Liu, M. Liao, V. Madhavan, and T. Darrell. BDD100K: A Diverse Driving Video Database with Scalable Annotation Tooling. In arXiv Preprint, 2018. 7, 36
- [ZHM⁺18] Oliver Zendel, Katrin Honauer, Markus Murschitz, Daniel Steininger, and Gustavo Fernandez Dominguez. Wilddash-creating hazard-aware benchmarks. In Proceedings of the European Conference on Computer Vision (ECCV), pages 402–416. openaccess.thecvf.com, 2018. 16
- [ZKS20] V. Zavrtanik, M. Kristan, and D. Skoaj. Reconstruction by Inpainting for Visual Anomaly Detection. Pattern Recognition, 2020. 11, 45
- [ZL17] Xiang Zhang and Yann LeCun. Universum prescription: Regularization using unlabeled data. In Thirty-First AAAI Conference on Artificial Intelligence, 2017. 15, 97
- [ZLK⁺17] B. Zhou, A. Lapedriza, A. Khosla, A. Oliva, and A. Torralba. Places: A 10 Million Image Database for Scene Recognition. IEEE Transactions on Pattern Analysis and Machine Intelligence, 2017. 49

- [ZLWY15] C. Zhang, H. Li, X. Wang, and X. Yang. Cross-Scene Crowd Counting via Deep Convolutional Neural Networks. In Conference on Computer Vision and Pattern Recognition, pages 833–841, 2015. 65
- [ZLZ⁺21] Sixiao Zheng, Jiachen Lu, Hengshuang Zhao, Xiatian Zhu, Zekun Luo, Yabiao Wang, Yanwei Fu, Jianfeng Feng, Tao Xiang, Philip H.S. Torr, and Li Zhang. Rethinking Semantic Segmentation from a Sequence-to-Sequence Perspective with Transformers. In Conference on Computer Vision and Pattern Recognition, 2021. 5, 79, 80, 81, 82, 84, 93
- [ZSQ⁺17] H. Zhao, J. Shi, X. Qi, X. Wang, and J. Jia. Pyramid Scene Parsing Network. In Conference on Computer Vision and Pattern Recognition, 2017. 1, 7, 33, 36, 37, 42, 47, 48, 56, 58, 72
- [ZZP⁺17] Bolei Zhou, Hang Zhao, Xavier Puig, Sanja Fidler, Adela Barriuso, and Antonio Torralba. Scene Parsing through ADE20K Dataset. In Conference on Computer Vision and Pattern Recognition, 2017. 12
- [ZZP⁺22] Chaoqiang Zhao, Youmin Zhang, Matteo Poggi, Fabio Tosi, Xianda Guo, Zheng Zhu, Guan Huang, Yang Tang, and Stefano Mattoccia. Monovit: Self-supervised monocular depth estimation with a vision transformer. In International Conference on 3D Vision, 2022. 79

KRZYSZTOF LIS

lis.krzysztof@protonmail.com



Education

- **PhD in Computer Science** 2017 - 2023
Computer Vision Laboratory with Pascal Fua, **EPFL** (École polytechnique fédérale de Lausanne)
My research area is semantic scene understanding for autonomous vehicles, with focus on detecting unusual obstacles on the road.
- **MSc degree in Electrical Engineering and Information Technology** 2015 - 2017
ETH Zürich (Swiss Federal Institute of Technology)
Thesis: "Design and evaluation of viewpoint invariant feature descriptors using convolutional neural networks" [\[PDF\]](#) [\[code\]](#)
GPA: 5.67/6
- **BSc degree in Computer Science** 2011 - 2014
University of Warsaw, Poland: Inter-Faculty Individual Studies in Mathematics and Natural Sciences
Thesis: "Framework for creating database applications using Rich Internet Application" [\[PDF\]](#)
I was the leader of a team researching frameworks used in rich internet applications, we also made an example application using the ZK framework (Java).
Grade: 4.5/5
- **BSc degree in Physics,** 2011 - 2015
University of Warsaw, Poland: Inter-Faculty Individual Studies in Mathematics and Natural Sciences
Thesis: "Compressive sensing methods in nuclear magnetic resonance" [\[PDF\]](#) Grade: 5/5

Experience

- **Research Intern, DeepMind, New York** 2022
Developed deep learning experiments for video analysis using Jax and DeepMind's compute cluster.
- **Doctoral Assistant, Computer Vision Lab, EPFL Lausanne** 2017 - 2023
 - Research in computer vision and deep learning for autonomous driving, with focus on safety by detecting rare obstacles.
 - I help with the lab's computational infrastructure based on Docker and Kubernetes.
 - As a teaching assistant for *Introduction to Computer Graphics*, I have created a set of WebGL exercises for students.
 - I advise student projects: practical deployments of computer vision (object detection for autonomous vehicle

Curriculum Vitae

racing and XR game), finite element simulation for game physics

■ **Student Help, Scientific Center for Optical and Electron Microscopy, ETH Zürich** 2016 - 2017

Analysis and processing of microscopy images in cooperation with researchers in the field of life sciences.

■ **Software Engineer, Google, Warsaw** 2015

Cloud Functions development at Google Cloud.

■ **Research student, University of Oxford, Department of Physics** 2014

Optimized performance of MATLAB numeric physics simulations using GPU computation.

■ **Technical Student, CERN, Geneva** 2013

(European Organization for Nuclear Research)

- Developer for the open source digital library framework **Invenio** for Digital Library Services department.
- Worked in an international team, participated in team meetings and discussions, took part in the process of designing new features and making decisions.
- Personally created a new module with Python business logic, database communication using object-relational-mapping and a web interface in HTML and JavaScript. The module was finished and accepted by supervisors.
- Participated in the life of the whole institution, cooperated with other departments - worked as a volunteer during CERN OpenDays (event where large number of people were visiting CERN), attended lectures and other events.

■ **Volunteer Tutor, Stanisław Staszic High School 14 / University of Warsaw** 2011 - 2014

- Organization of 4 summer camps / 2 courses and training students for Polish Olympiad in Informatics and **Scientific Summer School**.
- Gave lectures about algorithms and data structures: graph theory, dynamic programming, number theory, text algorithms, analytic geometry, physics.
- Created algorithmic tasks for students, solutions and answer checking programs.

Publications

■ *Perspective Aware Road Obstacle Detection* **IEEE Robotics and Automation Letters**, 2023
Krzysztof Lis, Sina Honari, Pascal Fua, Mathieu Salzmann [\[PDF\]](#) [\[IEEE\]](#)

■ *AttEntropy: Segmenting Unknown Objects in Complex Scenes using the Spatial Attention Entropy of Semantic Segmentation Transformers* 2022
Krzysztof Lis, Matthias Rottmann, Sina Honari, Pascal Fua, Mathieu Salzmann [\[arXiv\]](#)

■ *Detecting Road Obstacles by Erasing Them* 2022
Krzysztof Lis, Pascal Fua, Sina Honari, Mathieu Salzmann [\[arXiv\]](#)

■ *Segment Me If You Can: A Benchmark for Anomaly Segmentation* **NeurIPS 2021**
Robin Chan[†], Krzysztof Lis[†], Svenja Uhlemeyer[†], Hermann Blum[†], Sina Honari, Roland Siegwart, Mathieu Salzmann, Pascal Fua, Matthias Rottmann ([†]equal contribution) [\[PDF\]](#) [\[project page\]](#)

■ *Detecting the Unexpected via Image Resynthesis* **ICCV 2019**
Krzysztof Lis, Krishna Nakka, Pascal Fua, Mathieu Salzmann [\[PDF\]](#) [\[2 min talk\]](#)

- *Geometric and Physical Constraints for Drone-Based Head Plane Crowd Density Estimation*
Weizhe Liu, [Krzysztof Lis](#), Mathieu Salzmann, Pascal Fua [\[PDF\]](#) IROS 2019

Public presentations:

- *How Synthetic Training Powers Anomaly and Obstacle Detection in Traffic Scenes* Sept 2022
Invited talk at **Robust Understanding of Street Scenes Using Computer Vision** workshop,
University of Zagreb

Skills

- Computer vision, machine learning, deep learning, image processing
 - Deep learning frameworks: PyTorch, TensorFlow
 - Scientific applications: NumPy, ~~MT~~TeX, Jupyter
 - Deep learning for computer vision: semantic segmentation (PSPNet, DeepLab), object detectors (detectron2, FasterRCNN, RetinaNet, Yolo), conditional GANs for image synthesis (pix2pixHD), inpainting, style transfer, monocular depth estimation (MiDaS), patch descriptors (TFeat)
 - 3D computer vision: OpenCV, keypoint detection and matching, 3D reconstruction and SLAM
- Software engineering
 - Programming: Python, C++, Javascript, Java
 - Deployment: Docker, Kubernetes, docker-compose
 - Web development: JavaScript, HTML, CSS/Sass, React, Django/Flask/Jinja/aiohttp, Jekyll
 - Graphics and games: **Unreal Engine** (game / visualization engine), Blender, WebGL
 - Databases: SQL, SQLAlchemy
 - git version control
 - QT/PyQT graphical user interface framework
 - Build systems: CMake, Gradle
 - Linux - everyday usage, API, Docker
 - Also familiar with: ROS, Scala, Haskell, Lua, VHDL, MATLAB, OpenGL, OpenACC
- Algorithms (competitive programming), performance, computer networks, computer security
- Ability to quickly learn new technologies
- Communication, teamwork and teaching skills
- Languages:
 - **English** - fluent - CAE (Certificate in Advanced English) and TOEFL (Test Of English as a Foreign Language) certifications
 - **Polish** - native

Curriculum Vitae

- **German** - intermediate level (Zertifikat Deutsch certification)
- **Russian** - basic, reading

Projects

- **Kubernetes Job Dashboard** [\[github\]](#) 2019-2020
A web dashboard for jobs running on EPFL's Kubernetes cluster. Lists pods per user, the number of GPUs allocated, GPU memory and compute utilization.
The backend is using an async Python HTTP server `aiohttp` and interfacing with the Kubernetes API. The frontend web-application uses `Preact`.
- **Comparing Python, Go, and C++ on the N-Queens Problem** [\[web\]](#) [\[arxiv\]](#) 2020
Comparing the performance of programming languages and the Numba accelerator in Python.
- **Conquering a 5 GiB XML file with XML-native databases** [\[web\]](#) 2020
Using the XML-native database *BaseX* to process big XML data.
- **LabelGrab labeling tool** [\[github\]](#) 2019
Annotation tool for semantic and instance segmentation, with automated help from the GrabCut algorithm, using OpenCV and Qt.
- **Quadrotor pilot training using augmented reality** at ETH Zürich Mar 2016 - Dec 2016
[\[video presentation\]](#) [\[report\]](#) [\[code\]](#)
 - Training application: pilot controls a real drone, maneuvers between augmented reality obstacles.
 - Intuitive drone piloting interface using virtual reality and motion controllers.
 - Created an open source augmented reality plugin for Unreal Engine 4 [\[github\]](#)
- **Road Extraction from Aerial Images** [\[report\]](#) at ETH Zürich May - Jun 2016
 - Application detecting roads in aerial images of cities using Convolutional Neural Networks.
 - Evaluation of additional techniques for improving accuracy.
- **Kinect 3D Editor** [\[report\]](#) at ETH Zürich Oct 2015 - Feb 2016
 - Research into new applications of gesture controls.
 - Game where the user moves virtual objects with hand movements and competes to place them in a goal location.
- **Warlock** game [\[community site\]](#) 2008 - now
Leading an international team which created a custom game with a physics engine as a mod for Warcraft 3 and Dota 2, and is now working on a standalone version using Unreal Engine 4. The game achieved worldwide popularity among Warcraft 3 players.
- **LEGO plotter** at University of Warsaw Sep 2012 - Jan 2013
 - LEGO device capable of drawing graphics on paper, controlled by ATMEGA32 microcontroller.

- C program for the microcontroller and a desktop application which allows editing of the printed image and controls the device through a TCP/IP connection (the plotter behaves like a network printer)

Honors and Awards

- **EESTech Challenge 2017** 2017
1st place in Local Round at ETH Zürich, 6th place in Final Round
Developed an object detection and recognition system.
- 1st place at **ETH Hackathon** competition in California San Francisco, Apr 2016
 - Member of team representing ETH.
 - Developed an application for monitoring sensors in a smart home.
- 3rd place and bronze medal in XVIII Polish Olympiad in Informatics Poland 2011
- Bronze medal at **42nd International Physics Olympiad** Thailand 2011
- Bronze medal at **41st International Physics Olympiad** Croatia 2010
- 1st place at 60th Polish Physics Olympiad 2011
- 4th place at 59th Polish Physics Olympiad 2010
- Jan Zydler Award for best graduate of Stanisław Staszic High School no. 14 in Warsaw
Minister of National Education Scholarships "For outstanding educational achievements"
in 2011 and 2010
- President of Warsaw Scholarships "For outstanding educational achievements" in 2010 and 2008

Interests

- Science-fiction and fantasy
- Cats
- Gardening
- Cycling
- Game development and art

