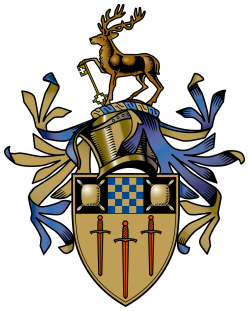
Learning Generic Deep Feature Representations

Jaime Spencer Martin

Submitted for the Degree of
Doctor of Philosophy
from the
University of Surrey



Centre for Vision, Speech and Signal Processing
Faculty of Engineering and Physical Sciences
University of Surrey
Guildford, Surrey GU2 7XH, U.K.

October 2021

Abstract

Feature representations are the backbone of computer vision. They allow us to summarize the overwhelming amount of visual information and distil it into its core components. Traditionally, features were hand-crafted to encode patterns that the researcher believed would be useful when solving a given task. Nowadays, with the advent of deep learning, the features have become part of the learning process. Unfortunately, most deep learning frameworks assume these features will be implicitly learnt as a by-product of the training process, resulting in features that only solve one task and cannot be easily reused. This slows down the development of solutions to new problems, as features must be re-learnt from scratch.

This thesis revisits the idea of dedicated feature learning as its own independent task in the age of deep learning. The aim is to learn representations containing useful properties that are effective across a range of different tasks. We achieve this by bridging the gap between explicit features used for geometric correspondence estimation and the implicit features used in deep learning frameworks for depth estimation, semantic segmentation, visual localization and more.

In the field of correspondence estimation, learned feature descriptors have recently started outperforming their hand-crafted counterparts. We make the observation that these approaches are simply learning a generic embedding that captures the (dis)similarity between different points. We argue that this goal is also shared by the implicit feature learning step of nearly all computer vision algorithms. The first contribution of this thesis proposes a generic dense feature learning approach based on this observation. We further show how the selection of negatives used to train the network affects the properties of the learnt features and downstream tasks. To this end, we introduce the concept of spatial negative mining, where negative samples are drawn based on their geometric relationship to the original positive correspondence.

As with other feature learning approaches, our first contribution requires pixel-wise ground truth correspondences—obtained via LiDAR or SfM data—during training. Obtaining this data can be challenging, especially if the images come from complex cross-seasonal environments with wide temporal or spatial baselines. The second and third contributions aim to learn features robust to the drastic changes in appearance caused by challenging weather and seasonal conditions. To achieve this we developed two self-supervised techniques that do not require ground truth annotations, but still show the network real-world data. The first achieves this by replacing strong spatial constraints with a global statistical criterion. This assumes each feature descriptor should only match well with a single feature in the other image. The third contribution re-incorporates within-season spatial constraints by simultaneously learning dense monocular depth, visual odometry and dense feature descriptors in a self-supervised manner.

Whilst the features presented above are generic, they typically need to be finetuned on a task to maximize their performance. Unfortunately, this process removes their generality. To solve this we turn to multi-task learning, where the objective is to learn features that perform well on the given set of training tasks. The final contribution of this thesis shows how spatial attention allows us to learn a generic feature space from which multiple downstream tasks can select pertinent features. This allows us to effectively exploit the relationships between multiple tasks and easily adapt to new ones, resulting in better performance and a significantly reduced resource usage.

Key words: Dense Feature Descriptor Learning, Correspondence Estimation, Depth Estimation, Multi-task Learning, Seasonal Robustness, Deep Learning

Email: jaime.spencer@surrey.ac.uk

WWW: https://www.surrey.ac.uk/people/jaime-spencer-martin

Acknowledgements

To begin, I would like to thank my supervisors, Dr. Simon Hadfield and Prof. Richard Bowden. You introduced me to the world of computer vision and ignited my passion for this field. Thank you for giving me this opportunity and for helping me reach my full potential. Most of all, thanks for the continued advice, feedback and support, especially during the tight deadlines and challenging times. You have helped make me a better researcher.

I would also like to thank everyone at the University of Surrey and CVSSP. Lunch & coffee breaks, weekly catch-ups, camping, Friday drinks, Christmas parties... My whole experience was enriched by these social gatherings—in-person or virtual—and the kindness of everyone surrounding me. This has been my home for the past 8 years, starting out as an Undergraduate, then a PhD Student and now a Research Fellow. I can't wait to see what the future brings.

Finally, I would like to thank my family and loved ones. Your support throughout this whole process has been invaluable and means the world to me. Thank you for keeping me sane and helping me believe in myself. I hope to be able to support you in any challenges you may face. I could not have done this without you. You have helped make me a better person.

Contents

Nomenclature

AP	Average Precision
AUC	Area Under the Curve
CNN	Convolutional Neural Network
DeFeat-Net	Depth & Feature Network
DoF	Degrees of Freedom
DoG	Difference of Gaussians
FCN	Fully Convolutional Network
FPS	Frames Per Second
GPS	Global Positioning System
IMU	Inertial Measurement Unit
LiDAR	Light Detection And Ranging
m-IoU	mean-Intersection over Union
MMA	Mean Matching Accuracy
MNN	Mutual Nearest Neighbour
MSA	Multi-Scale Attention
MSE	Mean Squared Error
MTL	Multi-Task Learning
NMS	Non-Maxima Suppression
NN	Nearest Neighbour
NT-Xent	Normalized Temperature-scaled Cross-entropy
PCA	Principal Components Analysis
PixCon	Pixel-wise Contrastive
RGB	Red, Green and Blue
RMSE	Root Mean Squared Error
ROC	Receiver Operating Characteristic
SAND	Scale-Adaptive Neural Dense

SGD	Stochastic Gradient Descent
SFA	Shared Feature Attention
SfM	Structure from Motion
SLAM	Simultaneous Localization and Mapping
SNN	Second Nearest Neighbour
SOTA	State-of-the-Art
SPP	Spatial Pooling Pyramid
SSD	Sum of Squared Differences
UFL	Universal Feature Learning
VO	Visual Odometry

Symbols

Introduced in Chapter 3

d	A depth value, typically bilinearly sampled from **D**.
D	A dense depth map.
$\hat{d}$	A disparity value, typically bilinearly sampled from $\hat{\mathbf{D}}$.
$\hat{\mathbf{D}}$	A dense disparity map.
E	The set of encoder features at multiple scales.
f	Dense feature map **F** downsampling factor.
f	A feature descriptor vector, typically bilinearly sampled from **F**.
F	A dense feature descriptor map.
h	The image height.
I	An RBG input image.
K	A 3×3 matrix defining the intrinsic parameters of a camera.
ℓ_{xent}	The Normalized-Temperature Cross-entropy loss between a point and a dense feature map.
$\mathcal{L}_{xent}$	The dense version of ℓ_{xent}.
ℓ_{con}	The pixel-wise contrastive loss between two points.
$\mathcal{L}_{con}$	The dense version of ℓ_{con}.
m	Contrastive loss negative margin.
n_{dim}	Dense feature map **F** descriptor dimensionality.
$N_{\hat{d}}$	Number of disparity levels when building the cost volume **V**.
N_{hca}	Number of Hierarchical Context Aggregation negative mining scales.
$\mathbb{N}$	The set of natural numbers.
p	A 2-D point in an image **I**.
$\dot{\mathbf{p}}$	The homogeneous version of **p**.
$\dot{\mathbf{p}}'$	The reprojection of $\dot{\mathbf{p}}$ onto a new image.
P	A 4×4 transform matrix defining the pose of a camera.

$\mathbf{q}$	A 3-D point in the world.
$\dot{\mathbf{q}}$	The homogeneous version of $\mathbf{q}$.
$\mathbb{R}$	The set of real numbers.
$\mathbf{V}$	A feature matching cost volume formed by two dense feature maps $\mathbf{F}$ shifted by $N_{\hat{d}}$ disparities.
w	The image width.
y	A label value extracted from $\mathbf{Y}$.
$\mathbf{Y}$	A dense label map indicating (non-)matching relationships between pairs of pixels in different images.
κ_{min}	Spatial negative mining minimum distance threshold.
κ_{max}	Spatial negative mining maximum distance threshold.
π	The function to project a world point $\mathbf{q}$ to an image point $\mathbf{p}$.
π^{-1}	The function to backproject an image point $\mathbf{p}$ to world point $\mathbf{q}$.
Π	The function to reproject an image point $\mathbf{p}$ onto a different image point $\dot{\mathbf{p}}'$.
τ	Cross-entropy softmax temperature.
$\Phi_{\mathbf{F}}$	A CNN producing a dense feature map $\mathbf{F}$ from an input image $\mathbf{I}$.
$\Phi_{\mathbf{F}}^{enc}$	The encoder portion of $\Phi_{\mathbf{F}}$.
$\Phi_{\mathbf{F}}^{dec}$	The decoder portion of $\Phi_{\mathbf{F}}$.

Introduced in Chapter 4

$\mathbf{C}$	Distance matrix between each combination of descriptors in $\mathbf{F}_1$ & $\mathbf{F}_2$.
$\hat{\mathbf{C}}$	Normalized distances from $\mathbf{C}$.
$\widetilde{\mathbf{C}}$	Similarities from applying softmax to $\hat{\mathbf{C}}$.
CX	The contextual similarity between two dense feature maps.
$\mathbf{F}_A$	Dense feature map corresponding to the Anchor.
$\mathbf{F}_N$	Negative feature map w.r.t. the Anchor.
$\mathbf{F}_P$	Positive feature map w.r.t. the Anchor.
$\mathbf{I}_A$	Triplet image corresponding to the Anchor.
$\mathbf{I}_N$	The negative pair w.r.t. the Anchor.
$\mathbf{I}_P$	The positive pair w.r.t. the Anchor.
ℓ_{a-con}	The aligned pixel-wise contrastive loss between two dense feature maps.
$\mathcal{L}_{a-con}$	The triplet version of ℓ_{a-con} using $\mathbb{T}_X$.
ℓ_{cx}	Triplet loss based on the contextual similarity CX.
$\mathcal{L}_{cx}$	Triplet loss based on ℓ_{cx} using both $\mathbb{T}_X$ and $\mathbb{T}_S$.
$\mathbb{T}_S$	Triplet where the Positive is the following frame in the sequence.

$\mathbb{T}_X$	Triplet where the Positive is drawn from the same location, different season.
λ_{cx}	Weight controlling the contribution of $\mathbb{T}_S$ in $\mathcal{L}_{cx}$.

Introduced in Chapter 5

a	A scaling factor for mapping $\hat{\mathbf{D}}_t$ to $\mathbf{D}_t$.
b	A bias factor for mapping $\hat{\mathbf{D}}_t$ to $\mathbf{D}_t$.
C_{t+k}	The set of pseudo-ground truth correspondences between images $\mathbf{I}_t$ & $\mathbf{I}_{t+k}$.
$\mathbf{D}_t$	The scaled depth map for an image $\mathbf{I}_t$.
$\hat{\mathbf{D}}_t$	The predicted disparity map for an image $\mathbf{I}_t$.
$\mathbf{F}_t$	The dense feature map for an image $\mathbf{I}_t$.
$\mathbf{F}_{t+k}$	The dense feature map for an image $\mathbf{I}_{t+k}$.
$\hat{\mathbf{F}}_{t+k}$	The reconstructed dense features $\mathbf{F}_t$ by bilinearly sampling the image $\mathbf{F}_{t+k}$ at locations C_{t+k}.
$\mathbb{I}_4$	The 4×4 Identity matrix.
$\mathbf{I}_t$	An image taken at time t.
$\mathbf{I}_{t+k}$	An image taken at time t+ k.
$\hat{\mathbf{I}}_{t+k}$	The reconstructed image $\mathbf{I}_t$ by bilinearly sampling the image $\mathbf{I}_{t+k}$ at locations C_{t+k}.
k	Time offset for the supporting images.
$\mathcal{L}_{feat}$	The relational loss between two dense feature maps.
$\mathcal{L}_{photo}$	The photometric loss between two tensors.
$\mathcal{L}_{i-photo}$	The photometric loss between two images.
$\mathcal{L}_{f-photo}$	The photometric loss between two dense feature maps.
$\mathcal{L}_{smooth}$	The smoothness regularization constraint on a depth map $\mathbf{D}_t$.
$\mathbb{M}$	The automask filtering out static points in $\mathbf{I}_t$.
$\mathbf{p}_t$	A 2-D point in an image $\mathbf{I}_t$.
$\mathbf{p}_{t+k}$	A 2-D point in an image $\mathbf{I}_{t+k}$.
$\mathbf{P}_{t \to t+k}$	The predicted pose between images $\mathbf{I}_t$ & $\mathbf{I}_{t+k}$.
t	Time at which the target image was taken.
λ_{smooth}	Weight controlling the contribution of $\mathcal{L}_{smooth}$.
λ_{ssim}	Weight controlling the contribution the *SSIM* term in $\mathcal{L}_{photo}$.
$\Phi_{\hat{\mathbf{D}}}$	A CNN producing a disparity map $\hat{\mathbf{D}}_t$ from an input image $\mathbf{I}_t$.
$\Phi_{\mathbf{P}}$	A CNN producing a pose $\mathbf{P}_{t \to t+k}$ from input images $\mathbf{I}_t$ & $\mathbf{I}_{t+k}$.

Introduced in Chapter 6

$\mathbf{B}_s$	The set of features at various scales produced by the backbone $\Phi_{\mathbf{B}}$.
C_s	The number of backbone feature channels at a given scale s.
$\mathbf{F}_s^{\mathbb{T}}$	The task specific features filtered from the backbone features $\mathbf{B}_s$ via the spatial attention $SA_s^{\mathbb{T}}$.
$\bar{\mathbf{F}}_s^{\mathbb{T}}$	The refined task features.
$\mathbf{H}_s^{\mathbb{T}}$	The refined scale features used in the MSA head prior to concatenation.
$\mathbf{H}^{\mathbb{T}}$	The concatenated task features $\mathbf{H}_s^{\mathbb{T}}$ used to make the final prediction $\mathbf{O}^{\mathbb{T}}$.
$l^{\mathbb{T}}$	The label indicating whether a lower number represents an increase in performance for task $\mathbb{T}$.
$M_b^{\mathbb{T}}$	The single task baseline performance on task $\mathbb{T}$.
$M_m^{\mathbb{T}}$	The multi-task framework performance on task $\mathbb{T}$.
N_s	The total number of scales s.
$N_{\mathbb{T}}$	The total number of tasks $\mathbb{T}$.
$\mathbf{O}_s^{\mathbb{T}}$	The initial predictions made for each task based on the refined task features $\bar{\mathbf{F}}_s^{\mathbb{T}}$.
$\mathbf{O}^{\mathbb{T}}$	The initial predictions made for each task based on the refined task features $\mathbf{H}^{\mathbb{T}}$.
s	The scale representing the downsampling in a feature map or task prediction.
SA	A convolutional spatial attention block.
$SA_s^{\mathbb{T}}$	The spatial attention learnt to convert the backbone features $\mathbf{B}_s$ into task specific feature at each scale $\mathbf{F}_s^{\mathbb{T}}$.
$\mathbb{T}$	The target task.
Δ_m	The average increase or decrease in performance across all evaluation tasks $\mathbb{T}$.
σ	The sigmoid operation, scaling the input to the range $[0, 1]$
ϕ	A convolutional block with BatchNorm and a ReLU activation.
$\phi_s^{\mathbb{T}}$	The convolutional blocks used to make the initial predictions $\mathbf{O}_s^{\mathbb{T}}$ based on the refined task features $\bar{\mathbf{F}}_s^{\mathbb{T}}$.
$\phi^{\mathbb{T}}$	The final convolutional block mapping from the task head features $\mathbf{H}^{\mathbb{T}}$ to the task output $\mathbf{O}^{\mathbb{T}}$.
$\Phi_{\mathbf{B}}$	A CNN producing the shared backbone feature representation $\mathbf{B}_s$.
ψ	A residual convolutional block.

List of Figures

List of Tables

Declaration

The work presented in this thesis is also present in the following manuscripts:

- Jaime Spencer, Richard Bowden, and Simon Hadfield. Scale-adaptive neural dense features: Learning via hierarchical context aggregation. In *Conference on Computer Vision and Pattern Recognition*, pages 6193–6202. IEEE Computer Society, jun 2019
- Jaime Spencer, Richard Bowden, and Simon Hadfield. DeFeat-Net: General monocular depth via simultaneous unsupervised representation learning. In *Conference on Computer Vision and Pattern Recognition*, pages 14390–14401. IEEE Computer Society, jun 2020
- Jaime Spencer, Richard Bowden, and Simon Hadfield. Same features, different day: Weakly supervised feature learning for seasonal invariance. In *Conference on Computer Vision and Pattern Recognition*, pages 6458–6467. IEEE Computer Society, jun 2020

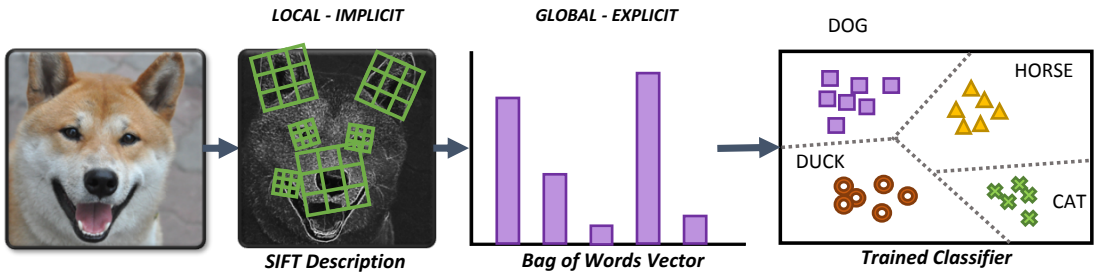
Chapter 1

Introduction

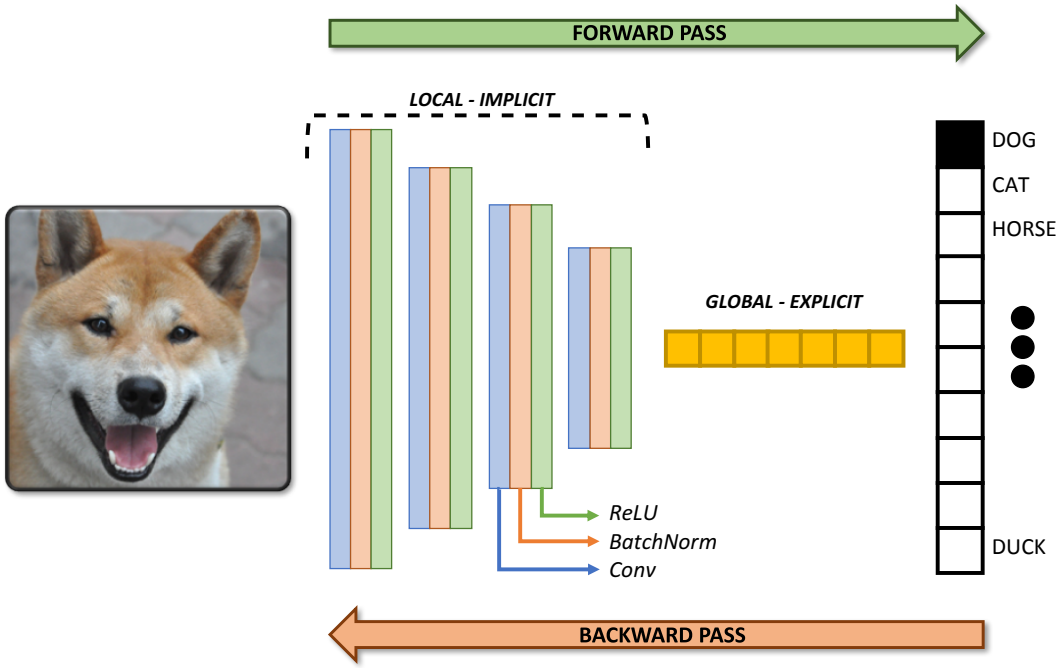
Computer vision is the process by which computers can see and understand the world surrounding them. It allows agents to make informed decisions about how to interact with the world and the consequences of their actions. But how can a computer convert an image, containing millions of measurements of reflected light intensity at different wavelengths, into useful information?

Traditionally, researchers developed hand-crafted heuristics to detect patterns in the image. For instance, SIFT [101] characterized each image patch based on its texture by analysing the distribution of spatial image gradients within it. To perform image classification, the features detected across the whole dataset were used to create a codebook or *Bag-of-Visual-Words* [170]. Based on the distribution of "words" within an image, it was then possible to train a linear classifier to distinguish the different classes and objects from one another. Unfortunately, the process of determining what features are relevant to each task is challenging and requires expert knowledge for each research area. Since the paradigm shift introduced by deep learning [151, 91, 84] the features have become part of the learning process. Given the network predictions and ground truth image labels, the error signal is propagated across all network layers and used to simultaneously update the parameters of both the convolutions generating the features and the linear layers classifying them. This is known as end-to-end training, which results in representations which would never have been developed heuristically, but provide much better performance and generalization capabilities.

Figure 1.1 helps to illustrate the various roles that features can have within a learning pipeline. *Explicit* features are those that can be directly applied to solve a target task. In the hand-crafted



(a) Hand-crafted Pipeline



(b) Deep Pipeline

Figure 1.1: Hand-crafted *vs*. Deep Pipelines. Previously, researchers had to decide what features to include in the pipeline based on the target task. With deep learning, features have become part of the learning process. We also make a distinction between features explicitly used to accomplish a tasks *vs*. those used to generate other features.

pipeline previously described, this would correspond to the resulting distribution of visual words for each image. In deep learning architectures, *explicit* features tend to appear at the final layer of the network and conceptually represent the highest level of distillation of the information from the original image. On the other hand, *implicit* features are used only by the following stages in the pipeline to generate new features. Once again, in the previous hand-crafted example this role falls to the SIFT features, which are used to generate the codebook and final word distributions, without directly being used by the classifier. It is worth noting that the same features can

have different roles depending on the target task and the given pipeline. For instance, SIFT features were used as *implicit* features in the image classification pipeline, but they can also be used as *explicit* features in tasks such as geometric correspondence estimation. Meanwhile, *implicit* features are more common in deep learning architectures, which rely on hierarchically aggregating previous layers and focus on learning features that perform well on only one task.

The *explicit* features in both traditional and deep pipelines can be further grouped based on the regions of the image they describe. *Global* features, such as those in *Bag-of-Visual-Words*, aim to summarize the whole image as a single feature vector. As such, they focus on describing the overall structure and content of the image. This is useful in coarse tasks such as image classification [84, 177, 29], landmark detection [131, 195], sketch-based image retrieval [140, 153] and coarse visual localization [5, 7, 155]. *Local* features, such as SIFT, instead target only a small patch within the image, providing additional information about the texture and local structure. These features are therefore more suited to fine-grained and dense prediction tasks, including object detection [144, 68, 20], semantic segmentation [100, 178], depth estimation [56, 62] and geometric correspondence estimation [120, 181, 41]. The methods presented in this thesis will focus exclusively on *local* features, as most real-world problems require the high level of detail provided by these features.

One final distinction can be made within *local* feature representations based on the nature of the task they are performing. Tasks such as geometric correspondence estimation are relational tasks. The objective is to capture relationships of similarity and dissimilarity between regions in different images. As such, a feature embedding space for this task cannot be identified as discriminative or accurate based on the encoding of a single point. It is only when we attempt to match features across different images that we can evaluate them. We therefore refer to these as *metric* features [32, 162, 127], useful in tasks such as disparity estimation [81, 24], optical flow [54, 78], Structure from Motion (SfM) [161, 97] or Simultaneous Localization and Mapping (SLAM) [126, 49, 215]. On the other hand, most computer vision tasks rely solely on the predictions made for a single image. For example, when evaluating a semantic segmentation model, the accuracy of one image does not influence the predictions or accuracy of the remaining images in the dataset. These are therefore absolute tasks, which make use of *non-metric* features. In other words, *metric* feature spaces are generally smooth and should encode a concept of locality and neighbourhood to support the computation of similarity metrics.

The difficulty of *metric* tasks is determined by the visual similarity between (non-)corresponding patches, as shown in Figure 1.2. If two images were taken only a few seconds apart or from very close viewpoints, the appearance of matching points will likely not change much. This is known as narrow baseline matching [40, 54, 78]. Knowing additional geometric constraints, such as the relative location of the cameras, can further eliminate a large number of points from the search space [81, 24]. On the other hand, large changes in viewpoint lead to appearance changes due to perspective transformations, scale changes and occlusions, while temporal changes (winter, night-time) cause drastic changes due to illumination conditions. This is know as wide baseline matching [161, 126, 155]. As humans, we are able to easily determine correspondences in these cases, since we have an intrinsic understanding of light and perspective and how it affects the world around us. However, it is very challenging to train a computer vision system to also encode this knowledge.

1.1 Motivation & Objectives

Metric and *non-metric* features have traditionally been framed as separate research problems. However, it is logical to theorize that *metric* features, which are required to be invariant to viewpoint and illumination changes, could also be applied to other tasks that are nowadays tackled by *non-metric* features within a deep neural network. Furthermore, since they are trained for general correspondence estimation, they should be able to generalize to a wider range of tasks more easily than single-task *non-metric* features.

One of the main challenges regarding the training of *metric* features is the data requirements. Supervised approaches rely on ground truth pixel-wise correspondences, typically obtained via Light Detection And Ranging (LiDAR) sensing or SfM reconstructions. These methods are not always feasible, since they can be expensive or computationally demanding. Furthermore, in the case of data taken with a wide temporal baseline, it is not always possible to align the data correctly due to sensor drift and inaccuracies. If the data collected is not accurate, the system will be trained to predict incorrect correspondences, reducing the accuracy and robustness of the resulting features.

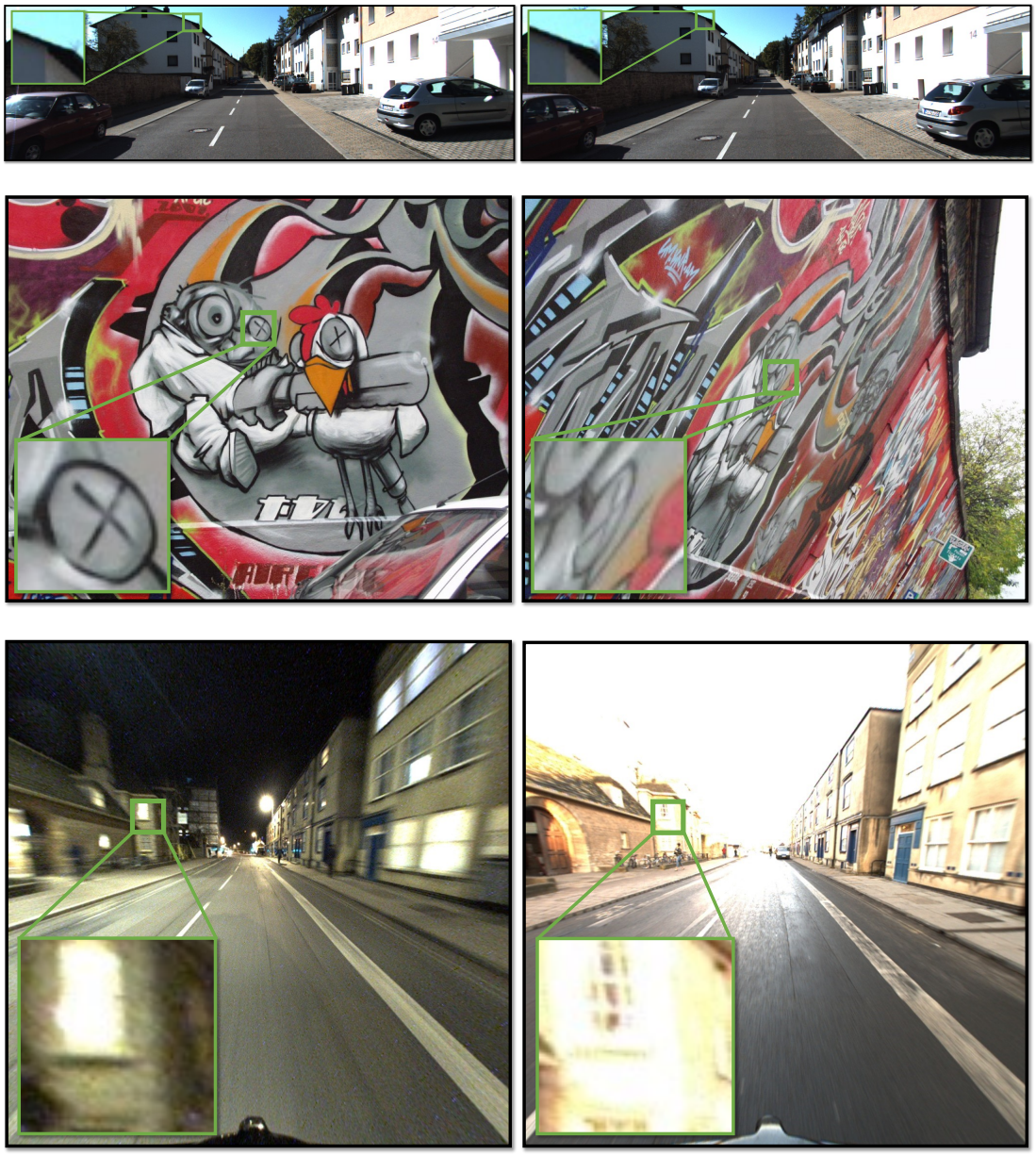


Figure 1.2: Challenges in Correspondence Estimation. The top row illustrates the case where images have a narrow baseline and therefore small changes in appearance. The middle row shows an increasing baseline in viewpoint, where corresponding patches are related by an affine transform. The bottom row shows a wide temporal baseline, resulting in large appearance changes due to illumination.

Meanwhile, self-supervised methods have relied on homographic and photometric augmentations. Whilst this overcomes the need for accurate ground truth data, it comes at the cost of not showing the networks real-world variation for viewpoint and lighting changes. Such approaches can also consist of multiple training and refining cycles, increasing overall training time.

The challenges described above pave the path for the overall aims of this thesis. Namely, these objectives can be summarized as:

1. To explore deep neural network solutions to dense metric feature learning and correspondence estimation.

2. To reduce the amount of labels required to train these representations whilst still showing real-world variation.

3. To explore the application of dense metric feature descriptors to a wide range of traditionally descriptive computer vision tasks.

4. To explore the simultaneous optimization of multiple tasks based on shared feature spaces.

1.2 Contributions

Chapter 3 introduces *Scale-Adaptive Neural Dense (SAND)* features [172], targeting objectives 1 & 3. When training dense feature descriptors, the set of positive examples is fixed, since each point has only one true correspondence. Meanwhile, there is a much larger pool of negative examples to choose from. We explore how the feature learning process and the downstream tasks are influenced by the selection of negative examples. By constraining the selection of negatives to a limited area around the original correspondence we are capable of training feature spaces more suitable for correspondence estimation as well as each of the target tasks. We therefore evaluate *SAND* on a variety of tasks: semantic segmentation, disparity estimation, visual localization and SLAM. In this case, semantic segmentation depends on the overall structure in order to classify each pixel. On the other hand, stereo disparity estimation may instead benefit from a higher discriminative power at a low level, focusing on the fine-grained structure and object boundaries.

Chapter 4 presents *Déjà-Vu* features [173]. As previously discussed, seasonal robustness is crucial for real-world feature learning. However, since most available training data consists of daytime conditions, previous approaches have instead relied on augmentations to introduce temporal variability. Even if seasonal data exists, it is typically not aligned well enough to provide pixel-perfect correspondences. We instead propose a weakly-supervised method for training seasonal features requiring only rough alignment and positioning, addressing objectives 1 & 2. This is based on the assumption that in each pair of images, each feature in the first image must match well with only one other feature in the second image. We introduce this constraint into a triplet loss framework, where positive images are drawn from the same location at a different season, whereas negatives are drawn from any location and any season. Despite having only image-level matching labels, we show that it its possible to train dense feature representations that can still be used for pixel-wise matching.

Chapter 5 presents *Depth & Feature Network (DeFeat-Net)* [174], a framework for simultaneously learning dense features, monocular depth and Visual Odometry (VO). This targets objectives 1, 2 & 4. Recently there has been an increased interest in monocular depth estimation. This is an inherently ill-posed problem, since it is geometrically impossible to distinguish between the scale and depth of an object given a single viewpoint. To humans this seems like a trivial task since we have priors on the size of certain objects and parallax from motion and stereo vision. Therefore, in order for a computer vision system to estimate depth from a single view, it is necessary to jointly learn a surrogate task and make assumptions about the appearance of the world. Unfortunately, some of these assumptions break down when dealing with low-light or night-time environments. We make the observation that, since feature descriptors such as those from the previous chapters are typically robust to lighting conditions, they should be able to overcome the photometric assumptions and provide a more stable supervision signal. Simultaneously, the predicted depth and motion allows us to obtain pseudo-ground truth correspondences to train the dense feature representation, further reducing the supervisory requirements. We show that it is possible to learn dense features, monocular depth and VO simultaneously in an entirely self-supervised manner without requiring any additional labels.

Finally, objective 4 is addressed in Chapter 6 with *Medusa. SAND* showed how it is possible to first train a dense feature learning network suitable for correspondence estimation and apply it to different downstream tasks. Meanwhile, *DeFeat-Net* simultaneously learned dense features

along with two highly correlated surrogate tasks. *Medusa* takes this as step further and aims to learn a universal feature space through Multi-Task Learning (MTL), with the objective of generalizing to a much wider range of tasks. Current approaches to MTL have focused on modelling task relationships at the decoder level by introducing dense connections between all possible task heads. Whilst this tends to improves performance, it comes at the cost of a quadratic parameter complexity. Furthermore, it becomes more challenging to decouple the different task heads and introduce new tasks without retraining. *Medusa* instead focuses on learning a shared feature space that contains useful features for all tasks. This is done via a combination of multiple spatial attention mechanisms, allowing each of the task heads to extract only the sub-feature space they require and alleviate *negative transfer* between tasks. This results in a lightweight architecture that performs on par with the current MTL State-of-the-Art (SOTA) on the training task set and provides greatly improved generalization to new unseen tasks.

1.3 Summary

To summarize, the main aim of this thesis is to learn feature spaces that can be applied to a wide range of computer vision tasks in real-world situations. This includes traditional *metric* uses for hand-crafted features (such as geometrical correspondence estimation) and as *non-metric* features in deep neural networks (for semantic segmentation, monocular depth, stereo disparity estimation or pose regression).

To provide context for the contributions of this thesis, Chapter 2 reviews existing related work done in the areas of correspondence estimation and dense metric feature learning. Chapter 3 focuses on objectives 1 & 3, learning a dense feature representation that can be used in a variety of hand-crafted and deep learning tasks. Objectives 1 & 2 are further explored in Chapter 4 via dense feature learning in the context of seasonal robustness and weak supervision. Chapter 5 presents *DeFeat-Net*, simultaneously learning dense features and monocular depth estimation in a fully unsupervised pipeline, addressing objectives 1, 2 & 4. Chapter 6 satisfies objective 4, learning shared features for a much wider range of computer vision tasks. Finally, Chapter 7 summarizes the findings of this thesis and discusses avenues for future work and directions for the field.

Chapter 2

Literature Review

This chapter will discuss the existing methods related to the topics in this thesis. Namely, feature detection and description. This includes hand-crafted and deep pipelines, sparse and dense representations, and joint detection and description. Whilst this provides an overview of the required knowledge to underpin the thesis, some chapters contain a specialised review of their relevant topics.

2.1 Feature Detection

This thesis primarily focuses on learning feature descriptions. However, a topic intrinsically related to it since its inception is that of feature detection. As previously mentioned, images contain millions of pixels, making it infeasible to run most description algorithms on each of them. Therefore, the first step in the majority of correspondence estimation pipelines is to determine the subset of points in the image that are most likely to contain useful and discriminative information. This is known as interest point or keypoint detection. Unfortunately, this is a semantically ill-posed problem. Even for humans, it can be complicated to determine which points will be suitable for tracking. Furthermore, downstream tasks may also have vastly different requirements. In practice, hand-crafted approaches have focused on detecting the presence of corners, blobs and regions in the image.

2.1.1 Corner Detectors

Moravec [125] defined interest points as those that are discriminative in their local neighbourhood. In practice, this was measured by computing the Sum of Squared Differences (SSD) between a small window centred around the given point and that window offset by a few pixels in each of the possible directions (vertical, horizontal & diagonals). The final locations were given by applying Non-Maxima Suppression (NMS) to the score map for the whole image. However, taking the interest point as the centre of the window is not always optimal. Förster [55] proposed a procedure for estimating keypoints with sub-pixel accuracy based on the intersection of lines tangent to those forming the corner.

These methods were sped up by approximating the SSD over sliding windows with the second moment matrix computed from the image gradients. The extremely popular Harris detector [65] showed how the eigenvalues of the matrix contain information about the local patch and can be used to determine the presence of corners. Namely, it defined the selection criteria based on the ratio between eigenvalues, given by the determinant and trace of the matrix. The Shi-Tomasi detector [165] simplified this by thresholding based on the minimum eigenvalue magnitude. Meanwhile, scale and affine invariance was introduced by Mikolajczyk and Schmid [118] through the Harris-Laplace and -Affine detectors. Scale invariance was achieved by computing the Harris response over an image pyramid consisting of progressively more downsampled and smoothed versions of the image. The corner scale was determined by finding the maxima using a Laplace operator. Meanwhile, the second moment matrix eigenvalues were used to estimate an elliptical affine region using the Lindeberg algorithm [96]. The estimated shape was further used to normalize the detected patch to a circular region.

SUSAN [171] instead considered a circular region around each point. The central point defined the intensity reference and was used to classify the remaining points as similar or different. Corners were detected as regions where there is an overall low similarity. FAST [147] greatly sped this process up by considering only the 16 edge pixels of the circular region. Corners were detected based on the number of contiguous pixels brighter or darker than the center reference. As such, points can be quickly rejected by testing the four vertical and horizontal points of the circle. This work was extended by approaches such as AGAST [110] and BRISK [93] in the form of adaptive point querying and scale-space selection, respectively. The detection

component of ORB [148] (oriented-FAST) further incorporated rotation invariance by finding the "centre of gravity" of the patch. This was based on the fact that the overall intensity is offset from the centre if the patch represents a corner.

2.1.2 Blob & Region Detectors

Corner locations are typically very well defined. However, their appearance does not vary much over changing scales. Blob and region detectors can provide complimentary information, given that their scale is well defined.

In the same vein as the second moment matix, Beaudet [14] showed how the Hessian matrix provides information about the image structure. Here, local maxima of both trace and determinant were used to detect blobs in the image. Scale and affine invariant versions were obtained in a similar manner to the procedure described for the Harris detector [119]. Meanwhile, SURF [13] drastically sped up computation time by approximating the Hessian matrix using box filters and integral images [193].

Similarly to image spatial gradients, several works [36, 59, 101, 102] noted that the scale-space Laplacian operator could instead be approximated by the difference between images of different scales. This lead to the Difference of Gaussians (DoG) operator by applying increasing levels of Gaussian blurring to the image. Keypoints were found as local maxima in both location and scale, and further refined via quadratic interpolation.

IBR [185] aimed to detect arbitrary shapes by casting rays from intensity extrema and analysing the change in intensity throughout them. The resulting regions could be replaced with an ellipse with a matching second moment matrix to make these compatible with the detections from previously described algorithms. Meanwhile, MSER [112] performed interest region detection by targeting light and dark regions stable over a large range of intensity thresholds. Parting from intensity extrema, a series of connected components were found by thresholding at various intensity levels. The final regions were those where the relative change in area over these thresholds is low. Several extension have been made to improve the efficiency of this algorithm [43, 130].

2.1.3 Learnt Detectors

TILDE [192] was one of the first machine learning approaches to tackle keypoint detection, using piece-wise linear functions to regress keypoint scores given the image intensity and gradients. Training ground truth was gathered from SfM data across seasons using SIFT keypoints, which were checked for geometric consistency and propagated to images where they were previously not detected. As previously discussed, keypoint detection is semantically ill-conceived. As such, rather than biasing learned approaches to these hand-crafted heuristics, it maybe be beneficial to let the framework discover by itself what constitutes a good keypoint. One such framework is DetNet [92], which aimed to regress stable detection anchors by introducing a covariant constraint. This prediction was done in a dense manner for the whole image, with the final detections given by a voting process between overlapping regions.

Key.Net [88] made use of hand-crafted filters to provide soft anchors for learned filters. This included first- and second-order derivatives such as those used by the Harris and Hessian detectors, respectively. The input was processed as an image pyramid, from which the detections were upsampled, combined and processed in a final convolution block. To make the learning end-to-end differentiable, they introduced an Index Proposal layer that computed the expected keypoint location based on a softmax over a small window. On the other hand, D2D [179] did not strictly learn keypoint detection, but instead made use of learned deep descriptors in the *describe-then-detect* paradigm introduced by [131, 45]. Keypoints were detected based on the absolute and relative saliency of descriptors in the dense feature map. Absolute saliency was approximated by the standard deviation of the descriptor, while relative saliency was equivalent to the Moravec detector [125] computing the L_2 distance between descriptors.

Most of the approaches detailed above follow the *detect-then-describe* paradigm. These detectors are designed to run on dense input images to produce a sparse set of interest points, which are then processed to describe their context region. Recently, several authors have begun to propose the *describe-then-detect* paradigm, where sparse interest points are extracted from dense feature maps rather than the input images. We will explore these approaches and their effectiveness in Section 3.2.1.

2.2 Feature Description

Feature description aims to summarize the information contained in a patch within an image. This is typically done with the objective of establishing geometric correspondences between images. The resulting vector must therefore be capable of encoding each point in a unique, distinctive manner, irrespective of the current lighting and viewpoint conditions. Hand-crafted approaches traditionally focused on summarizing the textures and spatial gradients within the patch. To avoid unnecessary computation in textureless regions, these methods were combined with the interest point detectors described in Section 2.1.

2.2.1 Histogram Descriptors

Schmid and Mohr [159] built an image retrieval system based on descriptors at sparse keypoints. The images were convolved with increasing Gaussian scales, from which invariants such as average luminance, gradient magnitude and the Laplacian were computed. These invariants were stacked to form the final descriptor vector for each point. Approaches that followed typically relied on summarizing the information contained in an image patch by creating a histogram from its spatial gradients. Perhaps the most well known of these approaches is SIFT [101, 102]. The descriptor received affine regions detected though DoG and computed their spatial gradient orientation and magnitude. These gradients were quantized into eight possible orientations, aggregated into a histogram over a 4×4 window and concatenated to form the final descriptor. These descriptors were made more robust to illumination changes by L_2 normalizing, clipping the values and re-normalizing. Meanwhile, matching performance was improved by analysing the ratio between the first and second closest descriptors to a query point, know as Second Nearest Neighbour (SNN) matching.

Future work focused on improving either the robustness or the speed of SIFT descriptors. For instance, Arandjelovic and Zisserman [7] made the observation that there are better ways of comparing histograms, namely through the Hellinger distance. However, since this metric is not as well optimized as matrix multiplication, it would be beneficial to instead modify the descriptors such that the standard L_2 distance could be used. This was achieved by L_1 normalizing the descriptor and taking its square root, leading to RootSIFT. On the other hand,

DSP-SIFT [42] improved robustness to scale changes by pooling gradients from multiple scales, while GLOH [120] used a polar grid with varying radial and angular directions. Fan *et al*. [50] noted that the descriptor depends on the orientation estimated by the keypoint, which may be inaccurate. In order to introduce rotation invariance they proposed to pool the gradients based on the order of their magnitude, which additionally incorporates information about the patch structure. In a similar vein, Hassner *et al*. [67] argue that the keypoint scale selection may also be flawed and instead represent each keypoint as a set of descriptors from multiple scales. Since increasing scales contain information from the previous scales, the set representation can be approximated as a single point, allowing for efficient comparison. Finally, ASIFT [206] improved robustness by augmenting the patches with multiple affine transforms, which could be compared using vanilla SIFT and checked for geometric consistency.

In terms of increasing efficiency, CHoG [23] aggregated gradient orientations over more complex patterns and quantized the resulting histograms to improve their bit-rate. PCA-SIFT [79] instead concatenated all horizontal and vertical gradients into a 3042-D vector, which is L_2 normalized and reduced to 20-D via Principal Components Analysis (PCA). Similarly to its detector, SURF [13] approximated the Gaussian derivatives using box filters and integral images, allowing for an efficient a filter pyramid. KAZE [3] improved scale selection by replacing Gaussian blurring with non-linear diffusion filtering, which preserved edge structure by adaptively blurring based on the gradient magnitudes in a small window. Meanwhile, DAISY [183, 184] provided an efficient SIFT-like descriptor which could be applied in a dense manner. SIFT's weighted sum of gradient orientations was replaced with various oriented derivatives of Gaussian filters. These values were then pooled from overlapping polar regions with increasing scales to form the final descriptor.

Brown *et al*. [15] introduced one of the first procedures for automatically learning robust feature descriptors. They created a network based on various building blocks, including Gaussian blurring, non-linear transformations, spatial pooling and normalization. The parameters for these blocks were jointly optimized using Powell's multi-dimensional set method [139] to maximise the Receiver Operating Characteristic (ROC) curve performance when classifying pairs of patches. However, this optimization process is non-convex and can lead to suboptimal representations. Simonyan *et al*. [168] instead formulated the learning of pooling regions and dimensionality reduction as a convex optimization problem. By computing a normalization

factor based on the statistics of the gradient magnitudes, a non-linear normalization step can be removed from the pipeline. The gradients within the patch were then aggregated using various learnt pooling methods and mapped to a lower dimensional space via a learnt projection matrix.

2.2.2 Binary Descriptors

To further improve the efficiency and storage requirements for descriptors, it is possible to represent a descriptor simply as a short binary string. LDAHash [175] used locality sensitive hashing to encode SIFT descriptor similarity based on the collision probability of binary codes. The covariance across classes was maximized by learning a projection matrix using Linear Discriminant Analysis. The sign of each channel in the mapped descriptor was used to obtain the final binary string. However, this still required computing the original SIFT descriptor.

Ojala *et al*. [132] introduced a method for directly computing a binary string from an image patch based on performing binary comparisons. In a method reminiscent of the SUSAN detector [171], the centre intensity was used as a reference and compared against all other intensities in the patch, which were marked as darker or brighter and flattened to form a texture unit. CS-LBP [72] simplified this by instead taking the sign of the gradient in each of the possible directions. This drastically reduced the dimensionality required, while improving robustness to noise.

BRIEF [18] expanded on LBP [132] by considering arbitrary comparisons within the patch, rather that always using the centre point as reference. The comparison kernels were obtained by randomly sampling with different strategies, such as uniform, Gaussian or polar weighting. Meanwhile, BRISK [93] performed the binary comparisons in a polar grid with increasing scale. This grid was rotated according to the gradient orientation around the keypoint to introduce additional invariance. ORB [148] instead obtained rotation invariance by rotating the BRIEF grid to match the estimated orientation using the “center of gravity”. This was done efficiently by discretizing the rotation angle and creating lookup tables. Furthermore, a training procedure was introduced to increase the overall robustness and reduce the dimensionality. The first dimension of the descriptor was taken as that with a mean of 0.5 and a large variance, while subsequent dimensions were chosen in a greedy fashion as those with low correlation w.r.t. the existing dimensions.

On the other hand, FREAK [1] proposed a human-inspired sampling pattern, arranged in

a polar grid and more densely sampled near the centre. The descriptor was constructed by performing comparisons between items with the same Gaussian scale and reduced using the training procedure introduced by ORB. Finally, A-KAZE [2] sped up the diffusion solution computation of its counterpart by using Fast Explicit Diffusion. In this case, the descriptor was obtained by rotating LDB [202] grids based on the estimated orientation and subsampling within the scale-space.

2.2.3 CNN Descriptors

Instead of hand-crafting the properties we want to encode into the feature descriptor, it is possible to let a neural network learn the most discriminative encoding for the given image patch. As with hand-crafted methods, early deep descriptors processed image patches individually, after the feature detection stage. However, the increased computing power and parallelism provided by modern GPUs has allowed for efficient architectures that can process all pixels in the image in a single forward pass. This has led to the emergence of dense feature descriptors.

Sparse. Based on the recent advances in deep learning, Fischer *et al*. [52] showed how SIFT descriptors could be replaced with intermediate features from a Convolutional Neural Network (CNN) trained on ImageNet [39] in both a supervised [84] and unsupervised [44] manner. Future work focused on learning the similarity function to predict if two patches are (non-)matching. MatchNet [64] trained this network using a cross-entropy loss based on the correct classification, while DeepCompare [207] introduced the Centre-Stream network where the centre and border of patches are processed by separate networks. However, given that these approaches were primarily learning the matching function, the feature embeddings were not explicitly optimized. This also meant that matching could only be done by in a pairwise fashion, which results in a large amount of duplicated computation.

DeepDesc [167] instead optimized the L_2 distance between descriptors directly based on a hinge loss, without the need for a matching network. They made the observation that it is not feasible to use all possible training pairs and that once most pairs are correctly classified the network stops learning. To counter this, they proposed a form of hard negative mining where only the top-k positive and negative pairs with the highest loss were backpropagated. T-Net [85] made a similar observation and instead countered it by introducing a global loss based on the

distribution of positive and negative distances. This aimed to minimize the variance within each distribution and separate their means by at least a target margin. PN-Net [8] and T-Feat [10] make use of triplet networks, extending hard negative mining by using both possible negative pairs within a triplet. This introduced the concept of "anchor swapping", where the smallest of the negative distances is used in either a softmax ratio [74] or triplet loss.

In order to greatly increase the number of negative examples seen by the network, L2-Net [181] introduced the concept of pairwise sampling, where only positive pairs are sampled and all other items in the batch are used as negatives. Their loss was inspired by Nearest Neighbour (NN) matching, where only the relative distance between pairs is important. This is enforced by taking the softmax over rows and columns and computing the loss based on the negative log likelihood. This loss was further applied to the first and last feature maps in the network, along with a compactness loss to de-correlate descriptor dimensions and prevent overfitting. On the other hand, HardNet [121] introduced a hard negative mining scheme inspired by the SNN matching criterion from SIFT. Instead of using the average loss from all negative samples, they claim that it is more beneficial to use only the hardest negative sample w.r.t. each anchor/positive. This forms a quadruplet, reduced to a triplet via anchor swapping. AffNet [123] further improved on this by making the loss constant w.r.t. the hardest sample by setting its gradient to zero. This favoured the distance between positive examples and helps to avoid local minima in the loss.

Later work focused on improving the descriptors by introducing additional constraints into the loss. For example, GeoDesc [105] integrated geometric constraints by classifying the hardness of training pairs based on the affine changes caused by the difference in viewpoint. This allowed them to sample hard pairs and exclude pairs that would not contribute to the training. Furthermore, they applied a margin relative to the positive distance rather than a fixed margin, reminiscent of SNN matching constraints. Other approaches instead focused on good utilization of the feature embedding space. Zhang *et al*. [211] introduced a regularization constraint based on the observation that two points randomly sampled from the unit hypersphere have a high probability of being orthogonal. SOSNet [182] instead used second-order similarity, stating that matching pairs of descriptors should also exhibit similar distances to other points in the embedded space. However, using all possible pairs can lead to worse results due to the fact that most of them are easily classified. As such, they used k-NN to select only the hardest pairs. Meanwhile, Tian *et al*. [180] and Zhang *et al*. [210] observe that, when using L_2 normalized

descriptors, both cosine similarity and L_2 distance losses result only in gradients perpendicular to the descriptors. To improve convergence speed they introduced regularizations pushing descriptor magnitudes to be similar to the matching pair or the mean, respectively. HyNet [180] made the additional observation that the cosine similarity favours positive distances, while L_2 distance favours negative samples. They therefore introduce a hybrid similarity measure to take this into account.

Dense. All the approaches discussed so far learn representations based on a single image patch. However, with the use of Fully Convolutional Networks (FCNs) [100] it is possible to efficiently produce a descriptor for every single pixel in the input image, resulting in dense feature maps. UCN [33] introduced a pixel-wise version of the contrastive loss [32]. Given a sampled descriptor in the first image, its nearest neighbour was found in the second image. If this was far from the ground truth reprojection, it was used as a negative sample. To increase robustness to affine warps they used Spatial Transformers [76] in a local manner, applying an independent transform to each point. Schidmt *et al*. [160] instead made use of KinectFusion [129] and DynamicFusion [128] models to obtain pixel-wise correspondences that could be propagated throughout a video.

To make better use of CNN architecture representations, HiLM [51] extracted features from both shallow and deep layers within the network. Matching is done in a coarse-to-fine manner, since deeper layers are better for large distance thresholds (*i.e*. recall), but are not as precise. The final location was refined by matching shallow features in a 32×32 window around the initial estimated location. SDC-Net [163] focused on the network architecture design, increasing the receptive field by stacking convolutions with increasing levels of dilation. Meanwhile, CAPS [194] proposed a weakly supervised approach requiring only relative pose between images. They exploit epipolar geometry and define the base loss as the distance between the matched point and the epipolar line. Since this is not enough to guarantee accurate correspondences, they introduced a cycle consistency loss by matching back to the first image.

2.3 Joint Detection & Description

Finally, given the intrinsic relationship between the two tasks discussed in this chapter, it is natural to attempt to learn them simultaneously. This leads to a simple form of multi-tasking

where the computation requirements can be shared and reduced. In the context of this thesis, the primary focus lies on the feature description component of these pipelines. However, it is necessary to discuss these approaches in full to understand the design choices and losses driving these algorithms. Furthermore, these provide the baselines against which we will compare our methods in future chapters.

LIFT [204] learned detection and description in a sequential manner, inspired by independent blocks for detection [192], orientation [205] and description [167]. Since each block was optimizing different objectives, these were trained in stages in reverse order. The descriptor network was trained using ground truth patches extracted from photo-tourism datasets, while the orientation estimator predicted the orientation that minimized the distance between positive descriptor pairs. Finally, the detector was trained to produce a keypoint score for every pixel in a small patch. This was trained by freezing the rest of the pipeline and maximizing the classification score. LF-Net [133] merged the detection and orientation estimation into a single network that additionally predicted keypoint scale. These were used to extract patches from the predicted location and warp them for use in the descriptor network. AffNet [123] instead used keypoint locations detected by the determinant of the Hessian and predicted their affine shape parameters. Similarly to [205, 204], these parameters were not directly optimized and were instead trained in conjunction with the descriptors. The estimated parameters were used to unwarp the patch using Spatial Transformers, which was forwarded to the descriptor network to minimize the loss previously described.

SuperPoint [41] introduced the first framework capable of simultaneously predicting keypoint locations and descriptors in a dense manner. The network was composed of a VGG [169] backbone, which split into two heads without deconvolution layers. The detection head represented the locations as an 8×8 cell in the original image, from which the exact location was extracted via softmax. Despite producing both outputs simultaneously, training still had to be done in multiple stages. The detector was initially trained using a synthetic dataset of shapes with keypoints labelled at corners, T-junctions, line segment ends, *etc*. To obtain a pseudo-ground truth for real images, the detector was applied to multiple versions of the image augmented with homographies, and aggregated to form a superset of points. SuperPointVO [40] introduced a reliability head to predict if a point is reliable to track based on its average track length and reprojection error when performing VO. UnSuperPoint [34] aimed to remove iterative

training without requiring pseudo-ground truth. This was achieved by introducing a consistency loss between the predictions generated for the original and augmented images. Furthermore, keypoints were regressed by predicting a horizontal and vertical offset for each point, allowing sub-pixel precision.

Similarly to [123, 40], R2D2 [145] argued that repeatability is not sufficient to guarantee that a point will be matchable. The detector loss attempted to maximise the cosine similarity between the saliency maps in corresponding patches in pairs of images. Meanwhile, descriptor matching was framed as a ranking optimization problem using a differentiable version of Average Precision (AP) [71] and weighted according to the predicted point reliability. NC-Net [146] targeted matching performance by learning dense correspondences while enforcing local geometric constraints. This was done by processing the volume of all possible pair similarities in a 4-D CNN. However, since NN matching is non-differentiable, they introduced a soft version where matches are weighted based on the highest similarity score within each row and column. Meanwhile, HF-Net [155] introduced a coarse-to-fine localization pipeline by predicting a global descriptor as well as dense detection and description. To make this mobile compatible, they opted for multi-task distillation from expert networks [6, 41]. DISK [186] instead trained a CNN for joint detection and description through reinforcement learning by optimizing for a high number of correct matches. The detection scores are divided into a grid, from which a single location was picked by sampling based on the softmax. Similarly, the matching distribution was given by the pairwise distance between descriptors at the selected keypoints. If both the forward and reverse matches are sampled, *i.e*. Mutual Nearest Neighbour (MNN) matching, the pair was considered a valid match and evaluated according to the ground truth data.

One common aspect of the approaches described in this section and in traditional pipelines is that the keypoint detections define the locations from which descriptors are extracted. An alternative option is to use the dense descriptors to define the locations which should be used as keypoints. This is know as the *describe-then-detect* paradigm. DELF [131] did this by pooling dense features with a weighted sum, where the weights were predicted by an attention network. However, since their target application is global image retrieval, the loss was based on the correct classification of the landmark represented by the image. As such, the descriptors were not explicitly optimized for correspondence estimation. D2-Net [45] instead detected keypoints based on the local maxima within the spatial response of each of the descriptor's

channels. During training this was softened by describing each point as the ratio to the channel with the highest score. This allowed for a loss that jointly optimized detection and description, weighted by the detection scores at each pixel. ASLFeat [106] expanded on this by introducing deformable convolutions constrained to affine warps. They further proposed a multi-level detection pyramid, where the D2-Net detection procedure was additionally applied to shallower feature maps. Finally, UR2KiD [201] jointly learned global descriptors with dense keypoints and descriptors while using only image-level correspondences. The dense descriptors were trained by taking the pairwise similarity and finding the maximum over rows and columns. The final loss was a triplet loss based on the average similarity scores between pairs of images.

2.4 Summary

This chapter has introduced a wide variety of existing approaches to both feature detection and description. While hand-crafted descriptors still provide good baselines for correspondence estimation tasks, the field has recently been dominated by learning based approaches. Furthermore, learning both components of the pipeline simultaneously has led to more efficient approaches where the network can jointly reason about what an interest point is and how to best describe it.

However, none of the approaches described have attempted to use the learnt features outside of geometric correspondence estimation. Even the dense feature learning approaches are still primarily used only in sparse downstream tasks. We aim to bridge the gap between *metric* and *non-metric* features, applying the learnt dense features to tasks such as depth/disparity estimation and semantic segmentation. Furthermore, we also address the existing heavy bias towards ideal sunny daytime conditions in current computer vision approaches. By introducing dense feature descriptors, we are capable of improving the robustness to low-light conditions and overcome some of the assumptions made to train these algorithms.

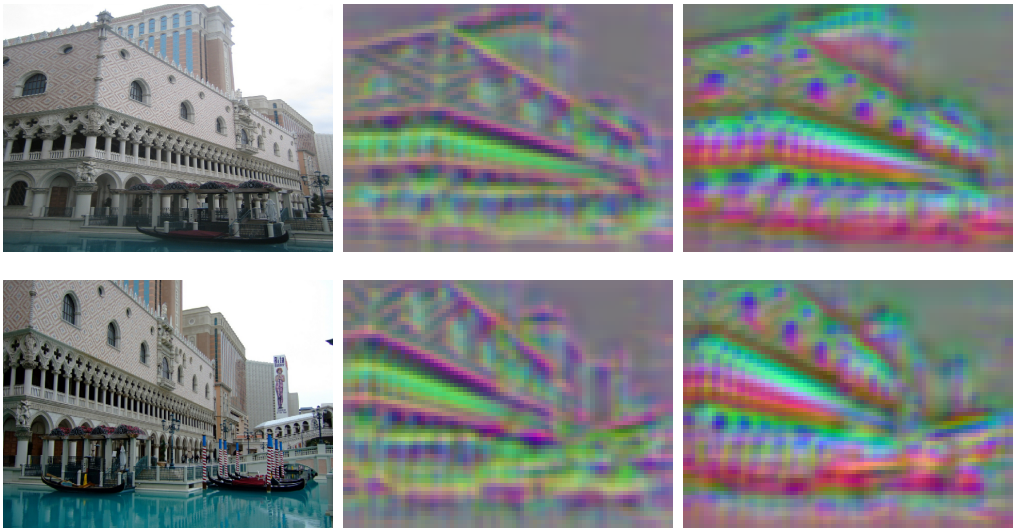
Chapter 3

Scale-Adaptive Neural Dense Features

The methods reviewed in the previous chapter provide feature representations that are suitable for geometric correspondence estimation. Hand-crafted approaches typically summarize the texture and spatial gradients within the given image patch. Meanwhile, deep learning approaches learn an abstract representation encoding whether two pixels are likely to correspond to the same point. Despite being generic representations, these features have been exclusively used to establish correspondences. We argue that such representations should also be applicable to a wider range of computer vision problems, *i.e*. both *metric* and *non-metric* tasks.

As a solution to this, we present *SAND* [172] features. In the context of this thesis, this tackles objectives 1 & 3. The main objective is to learn a dense feature representation capable of estimating geometric correspondences, as well as supporting a wide range of computer vision tasks. Designing such a system is highly challenging due to the fact that different tasks may require very different properties from a features representation. For instance, tasks such as optical flow, object tracking or VO may favour locally unique features, since they tend to require iterative processes over narrow spatio-temporal baselines. On the other hand, SLAM and visual localization require globally consistent features, since these are matched over much wider baselines with large viewpoint or seasonal appearance changes. Similarly, segmentation focuses on the overall classification for each pixel in the image, without making a distinction between different instances within that class.

Tasks currently performed using end-to-end deep learning, such as optical flow or semantic segmentation, assume the required features are implicitly learnt by the network. As a result,



(a) Image Pair **(b)** Global Features **(c)** Local Features

Figure 3.1: Dense Features Produced by *SAND*. By drawing negatives from different *scales* we train dense features with different properties for each task. Furthermore, we show how we can combine and train multiple scales to improve performance.

these features become specific to each task and may be challenging to reuse in other systems. Furthermore, they are not suitable for correspondence estimation or tasks such as localization and VO. We instead frame the problem as a two step process, where a collection of generic feature descriptors with various properties are pre-trained and then adapted to various geometric and non-geometric problems. The initial feature training makes use of large-scale SfM [95] or driving [60] data to learn representations capable of estimating correspondences. Many current feature learning approaches target negative pairs that are challenging for the network. We instead introduce constraints based on the geometric relationship between positive and negative pairs. We show how targeting specific regions around the original correspondence leads to different behaviour and how these regions can be combined into a more robust multi-scale representation.

The learnt features are then applied to different downstream tasks. In the case of disparity estimation and semantic segmentation, the encoder portion of the network is replaced with the learnt *SAND* features. From this, a matching cost volume is built and further processed in a 3-D stacked hourglass network. Visual localization is instead performed by directly replacing the input image to the network with its dense feature counterpart. Finally, SLAM requires no additional training and *SAND* can instead simply replace existing sparse feature descriptors.

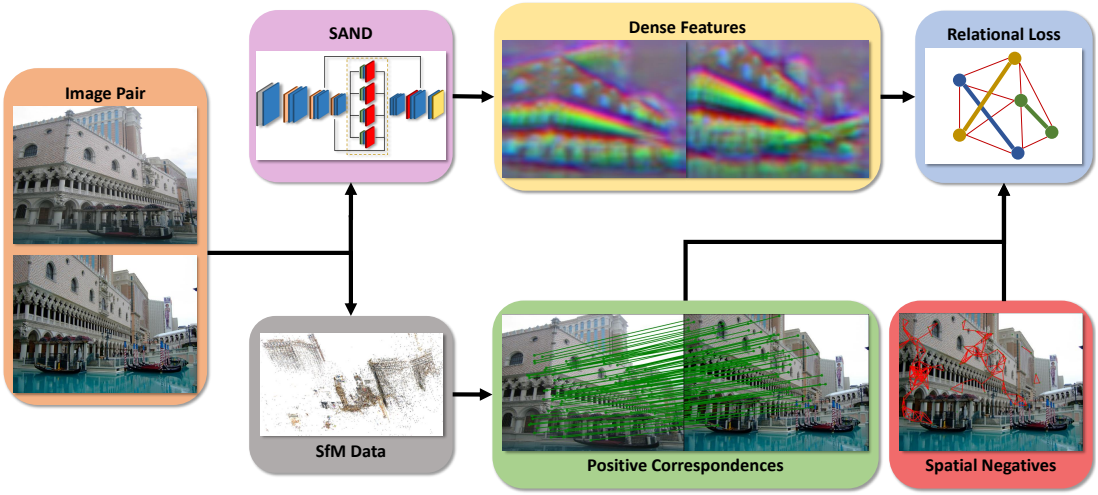


Figure 3.2: Proposed *SAND* Overview. From large-scale SfM data we obtain ground truth correspondences to train dense features. We propose a spatial negative mining scheme that can be combined with any given relational embedding loss to train features to be unique within the chosen scale.

3.1 Methodology

In this section we describe the details of the proposed framework, shown in Figure 3.1. Once again, the objective is to learn a high-dimensional descriptor for each pixel in the input image. These descriptors should contain information useful across a range of tasks. To achieve this, a set of ground truth correspondences between pairs of images is provided via SfM reconstructions. We sample negative pairs based on the spatial negative mining scheme outlined in Section 3.1.3. In Section 3.1.4 we further show how to combine multiple mining strategies to simultaneously train dense descriptors containing a wide range of desirable properties.

3.1.1 Network

Contrary to approaches learning feature descriptors from image patches [8, 181, 121, 105, 182, 180], our objective is to produce a feature descriptor for every single pixel in the input image in a single forward pass [33, 160, 41, 45, 194]. To allow for an arbitrary input image size we opt for an encoder-decoder FCN [100] architecture. The encoder provides the initial feature extraction, defined as

$$E = \Phi_{\mathbf{F}}^{enc}\left(\mathbf{I}\right), \tag{3.1}$$

where $\mathbf{I}$ is the input image, $\Phi_{\mathbf{F}}^{enc}$ the encoder portion of the feature network and E the set of features from each encoder block. The encoder is typically formed by five downsampling stages, using building blocks such as ResNet [70], VGG [169] or MobileNet-v2 [154]. The final stage of the encoder can be replaced with a Spatial Pooling Pyramid (SPP) block [69] to incorporate features from multiple scales.

This is followed by a custom dense decoder with convolutional blocks that upsample the encoder representation. Skip connections are included between corresponding sized layers to combine deeper decoder features with early low-level features suitable for dense predictions. This is given by

$$\mathbf{F} = \Phi_{\mathbf{F}}^{dec}\left(E\right), \tag{3.2}$$

with $\mathbf{F}$ as the final dense feature representation. As such, the feature network provides a mapping $\Phi_{\mathbf{F}} : \mathbb{N}^{h \times w \times 3} \mapsto \mathbb{R}^{h/f \times w/f \times n_{dim}}$, where n_{dim} is the desired descriptor dimensionality. In practice, the final representation is downsampled by a factor f to reduce the memory requirements and improve the global consistency of the feature maps.

3.1.2 Losses

Feature description is a *metric* task, where the system can only be evaluated using the relationship between different images and points in the dataset. Furthermore, since the objective is to learn the feature representation, there are no ground truth embeddings for use during training. The only ground truth available is the pairs of samples that should have a similar representation.

Unfortunately, we cannot rely on a simple attractive loss, such as L_2 distance, due to the degenerate solution where every point has a constant embedding. We therefore need to incorporate the concept of negative samples, *i.e*. those that should be distant from each other in embedding space. As such, these losses are known as relational losses. *SAND* can be trained with any relational loss, but in this work we will focus on the Pixel-wise Contrastive (PixCon) and Normalized Temperature-scaled Cross-entropy (NT-Xent) losses.

PixCon is an adaptation of a well established contrastive learning loss [32, 33, 160] for dense pixel-wise problems. Given a pair of 2-D points $\mathbf{p}_1$ & $\mathbf{p}_2$ from two different images $\mathbf{I}_1$ & $\mathbf{I}_2$,

this loss is defined as

$$\ell_{con}\left(y, \mathbf{p}_1, \mathbf{p}_2\right) = \begin{cases} \left\|\mathbf{f}_1 \text{-} \mathbf{f}_2\right\|_2^2 & \text{if } y = 1 \\ {}^{+}(m \text{-} \left\|\mathbf{f}_1 \text{-} \mathbf{f}_2\right\|_2)^2 & \text{if } y = 0 \\ 0 & \text{otherwise} \end{cases}, \tag{3.3}$$

where y is the label indicating if the pair is (non-)matching, m is the margin, $\mathbf{f} = \mathbf{F}\left\langle\mathbf{p}\right\rangle$ is a feature descriptor bilinearly sampled from the dense feature map, $\left\|\mathbf{f}_1 \text{-} \mathbf{f}_2\right\|_2$ is the L_2 distance and ${}^{+}(a) \equiv \max\left(0, a\right)$, *i.e.* the positive clipping function. Intuitively, the objective of this loss is to minimize the distance between matching descriptors, while separating non-matching pairs by at least a target margin m.

To extend this loss to a dense scenario, it is necessary to create a dense label map $\mathbf{Y}$ describing the relationship between each possible pair of pixels. The loss can then be aggregated over all pairs through

$$\mathcal{L}_{con}\left(\mathbf{Y}, \mathbf{F}_1, \mathbf{F}_2\right) = \mathop{\sum\!\!\!\!\!\!\int}_{\mathbf{p}_1} \mathop{\sum\!\!\!\!\!\!\int}_{\mathbf{p}_2} \ell_{con}\left(\mathbf{Y}\left(\mathbf{p}_1, \mathbf{p}_2\right), \mathbf{p}_1, \mathbf{p}_2\right), \tag{3.4}$$

where $\sum\!\!\!\!\!\!\int_{i=0}^{N} i \equiv \frac{1}{N}\sum_{i=0}^{N} i$ represents the average summation.

As has been pointed out by previous works [167, 85, 121], it is infeasible to use all available negative examples due to computational requirements. Furthermore, if most examples are already correctly classified, the network will not learn effectively. NT-Xent [197, 188, 29] aims to solve this by framing the learning of a feature embedding as a classification problem. The classification prediction is obtained based on the softmax over the distance between one point and all available pairs in the second image. The loss is therefore defined as

$$\ell_{xent}\left(\mathbf{Y}, \mathbf{p}_1, \mathbf{F}_2\right) = \frac{\sum_{\mathbf{p}_2} [\![\mathbf{Y}\left(\mathbf{p}_1, \mathbf{p}_2\right) = 1]\!] \exp\left(\frac{\mathbf{f}_1^\top \mathbf{f}_2}{\left\|\mathbf{f}_1\right\| \left\|\mathbf{f}_2\right\|} \frac{1}{\tau}\right)}{\sum_{\mathbf{p}_2} [\![\mathbf{Y}\left(\mathbf{p}_1, \mathbf{p}_2\right) \in \{0, 1\}]\!] \exp\left(\frac{\mathbf{f}_1^\top \mathbf{f}_2}{\left\|\mathbf{f}_1\right\| \left\|\mathbf{f}_2\right\|} \frac{1}{\tau}\right)}, \tag{3.5}$$

where $[\![]\!]$ represents the Iverson bracket, τ is the softmax temperature and $\frac{a^\top b}{\|a\|\|b\|}$ is the cosine similarity between descriptors. Note that, since there is only ever one positive correspondence per point, the condition $\mathbf{Y}\left(\mathbf{p}_1, \mathbf{p}_2\right) = 1$ is only met once. This loss is advantageous because each negative sample is automatically weighted according to its difficulty w.r.t. the positive pair. Furthermore, this encourages a uniform separation between all negative pairs, leading to a well

distributed embedding space. Once again, this loss can be extended to instead account for two dense feature descriptor maps through

$$\mathcal{L}_{xent}\left(\mathbf{Y}, \mathbf{F}_1, \mathbf{F}_2\right) = \sum_{\mathbf{p}_1} \ell_{xent}\left(\mathbf{Y}, \mathbf{p}_1, \mathbf{F}_2\right). \tag{3.6}$$

3.1.3 Spatial Negative Mining

The losses discussed in the previous section require ground truth positive and negative descriptor pairs. This ground truth can be obtained from multiple sources, including optical flow [33, 163], homographic augmentations [41, 34] or SfM reconstructions [15, 181, 45]. Despite being easier to obtain, optical flow data is usually restricted to narrow baselines without large viewpoint changes. Similarly, synthetic homography augmentations do not show the network real world viewpoint variation, which can result in lower robustness upon deployment. We therefore opt for real world correspondences obtained via SfM reconstructions.

In this case, given the 2-D point in the image $\mathbf{p}$, its 3-D position in the world $\mathbf{q}$ is obtained via

$$\mathbf{q} = \pi^{-1}\left(\mathbf{p}|\mathbf{P}, \mathbf{K}, \mathbf{D}\right) = \mathbf{P}^{-1}\,\mathbf{K}^{-1}\,\dot{\mathbf{p}}\,\mathbf{D}\left\langle\mathbf{p}\right\rangle, \tag{3.7}$$

where π^{-1} is the backprojection function, $\mathbf{K}$ are the camera's intrinsic parameters, $\mathbf{P}$ are the camera's extrinsic parameters (*i.e.* its pose in the world), $\mathbf{D}$ a dense depth map and $\dot{\mathbf{p}}$ is the homogenous version of $\mathbf{p}$. This point can then be projected onto any of the other available images through

$$\mathbf{p} = \pi\left(\mathbf{q}|\mathbf{K}, \mathbf{P}\right) = \mathbf{K}\,\mathbf{P}\,\dot{\mathbf{q}}. \tag{3.8}$$

Any point can therefore be reprojected onto a different image by combining both functions as

$$\dot{\mathbf{p}}' = \Pi\left(\mathbf{p}\middle|\mathbf{K}', \mathbf{P}', \mathbf{P}, \mathbf{K}, \mathbf{D}\right) = \mathbf{K}'\,\mathbf{P}'\,\mathbf{P}^{-1}\,\mathbf{K}^{-1}\,\dot{\mathbf{p}}\,\mathbf{D}\left\langle\mathbf{p}\right\rangle. \tag{3.9}$$

In the naïve case, the label map $\mathbf{Y}$ containing the pairs used for training could simply be defined as $\mathbf{Y}\left(\mathbf{p}_1, \mathbf{p}_2\right) = \left[\!\left[\Pi\left(\mathbf{p}_1|\mathbf{K}_2, \mathbf{P}_2, \mathbf{P}_1, \mathbf{K}_1, \mathbf{D}_1\right) = \mathbf{p}_2\right]\!\right]$. While the set of positive examples is limited by the number of available SfM correspondences, these can be combined into an almost infinite amount of negative pairs. The simplest solution to make this tractable would be to uniformly sample a fixed number of negatives for each positive pair [160]. Another option is to instead sample only positive pairs and use the remaining samples in a training batch as

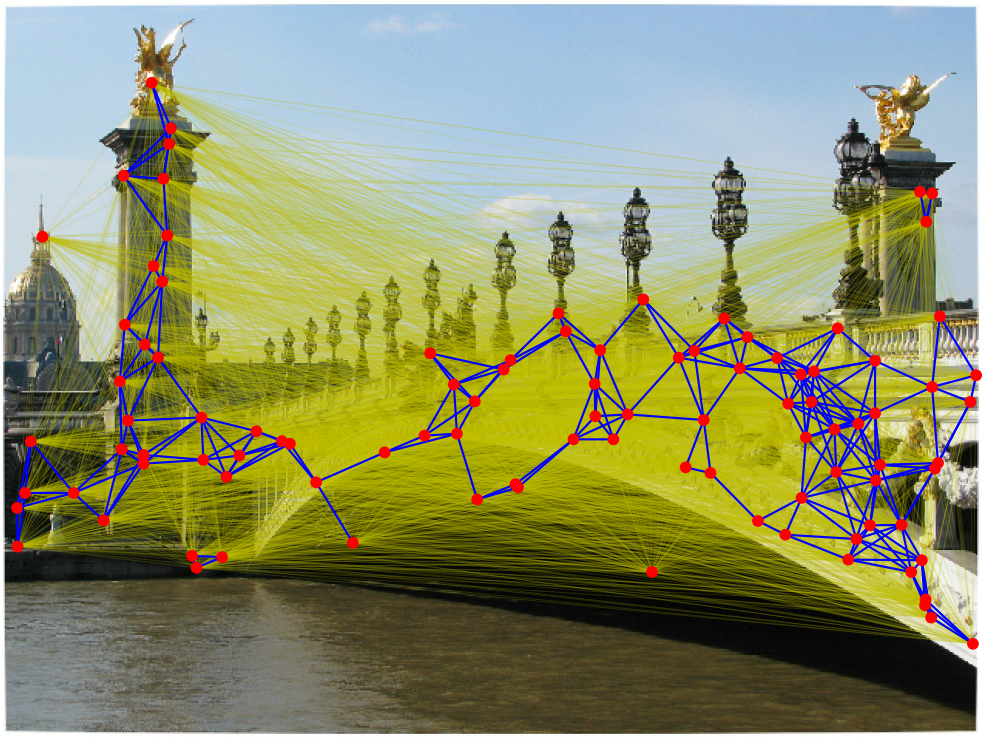


Figure 3.3: Spatial Negative Mining. Given the set of original image locations used as positive correspondences (red), we visualize the potential negative candidates for each of those points. Yellow lines represent negatives chosen when using a **G**lobal scheme ($\kappa_{min} = 50, \kappa_{max} = \infty$), whereas blue is the **L**ocal scheme ($\kappa_{min} = 1, \kappa_{max} = 50$).

the negatives w.r.t. each other [181]. This leads to an effective way of scaling the number of negative samples, providing a NN-style loss where only the relative distance between pairs is important. Recent approaches [121, 105, 45] have additionally focused on selecting examples that are challenging for the network, *i.e*. hard negatives.

However, when applying feature descriptors to *non-metric* tasks, hard negative mining does not always reflect the requirements of these tasks. For instance, narrow baseline problems that depend on the find-grained representations do not need to be robust to changes in distant regions of the image. Furthermore, repeating patterns or untextured regions could lead to the training set being dominated by trivial negative examples that are impossible to distinguish from each other.

We therefore instead select negative samples based on their geometric relationship w.r.t. the positive example. We refer to this strategy as *spatial negative mining*. This allows us to introduce invariance into the features based on the desired spatial properties in the downstream tasks.

We formalize this as

$$\mathbf{Y}\left(\mathbf{p}_1, \mathbf{p}_2\right) = \begin{cases} 1 & \text{if } \Pi\left(\mathbf{p}_1 | \mathbf{K}_2, \mathbf{P}_2, \mathbf{P}_1, \mathbf{K}_1, \mathbf{D}_1\right) = \mathbf{p}_2 \\ 0 & \text{if } \kappa_{min} < \left\| \Pi\left(\mathbf{p}_1 | \mathbf{K}_2, \mathbf{P}_2, \mathbf{P}_1, \mathbf{K}_1, \mathbf{D}_1\right) - \mathbf{p}_2 \right\|_2 < \kappa_{max} \\ -1 & \text{otherwise} \end{cases}, \tag{3.10}$$

where κ_{min} and κ_{max} represent the minimum and maximum distance thresholds, respectively. As shown in Figure 3.3, this means that we are restricting negative samples to belong to a certain circular region around the original ground truth correspondence. This defines the context region in which we expect that descriptor to be unique in.

As has been discussed, narrow baseline matching requires locally discriminative features. Distant regions do not need to be unique, as long a fine details cause a response change in the embedding space. To encourage this, we can sample exclusively from a small neighbourhood by setting a small value for κ_{min} and κ_{max}. Globally consistent features can instead be obtained by excluding the immediate neighbourhood and sampling from all distant regions in the image.

3.1.4 Hierarchical Context Aggregation

Complex tasks may require more than one set of properties from its features. For instance, SLAM performs both narrow (VO) and wide (localization) baseline matching. As such, features that only satisfy one of these properties will still lead to non-optimal results. To benefit from multiple spatial negative mining strategies simultaneously, the dense feature descriptor can be split into multiple "sub-descriptors" as $\mathbf{f}^i = \mathbf{f}\left(\left[(i\text{-}1)\,\frac{n_{dim}}{N_{hca}} \,..\, i\frac{n_{dim}}{N_{hca}}\right]\right)$, where N_{hca} is the number of spatial negative mining strategies and $[a\,..\,b]$ represents slicing across the given channels. Each of these sub-descriptors can be provided with samples from different spatial negative scales, as illustrated in Figure 3.4

Using a a slight abuse of notation, we can redefine the final dense losses to be

$$\mathcal{L}_{con}\left(\mathbf{Y}, \mathbf{F}_1, \mathbf{F}_2\right) = \sum_{i=1}^{N_{hca}} \mathcal{L}_{con}\left(\mathbf{Y}^i, \mathbf{F}_1^i, \mathbf{F}_2^i\right), \tag{3.11}$$

$$\mathcal{L}_{xent}\left(\mathbf{Y}, \mathbf{F}_1, \mathbf{F}_2\right) = \sum_{i=1}^{N_{hca}} \mathcal{L}_{xent}\left(\mathbf{Y}^i, \mathbf{F}_1^i, \mathbf{F}_2^i\right). \tag{3.12}$$

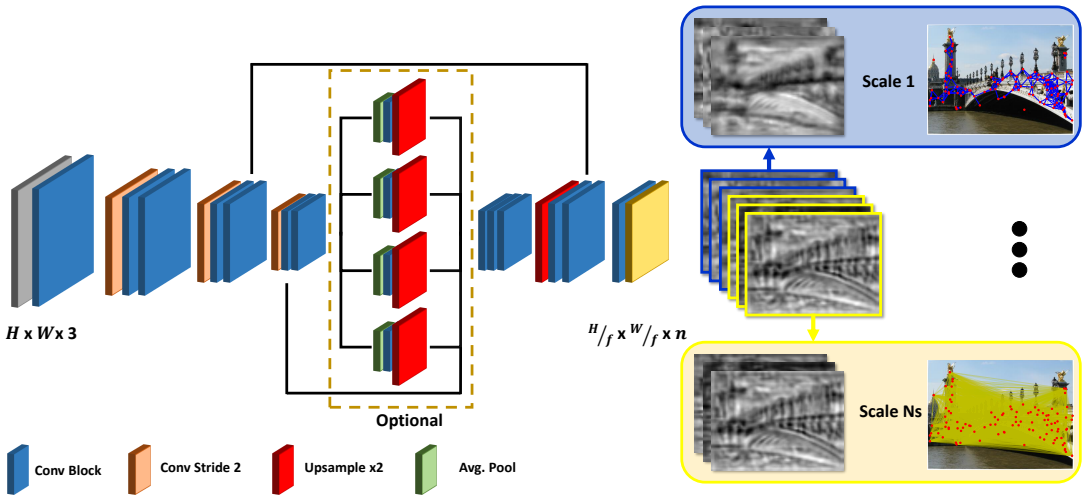


Figure 3.4: Hierarchical Context Aggregation. Given a dense feature map, it can be split in multiple sections. Each of these is trained with a different set of negatives and scales. As a result, these features combine information and properties from all these scales.

Here each set of labels $\mathbf{Y}^i$ is generated using different spatial negative mining thresholds, as described in Section 3.1.3. In this work we consider three mining scales: **G**lobal ($\kappa_{min} = 50, \kappa_{max} = \infty$), **L**ocal ($\kappa_{min} = 1, \kappa_{max} = 50$) and the hierarchical combination of both (**GL**). These values were chosen empirically to provide balanced results, as discussed further on in Section 3.2.2.

3.2 Feature Descriptor Evaluation

Training & implementation details. We train *SAND* using 116 scenes from the MegaDepth dataset [95]. When selecting images as training pairs we enforce an overlap between image pairs in the range $[0.2, 0.8]$ with no restrictions in scale change. The number of training samples is balanced by sampling 200 images per scene. For each image pair, ground truth correspondences are obtained by reprojecting the depth maps per (3.9) and applying z-buffering. From these, 2000 different correspondences are sampled from each image pair. The images are cropped around these correspondences and resized. This leads to a total of 44.6M positive pixel correspondences for training. This is then expanded by randomly applying photometric augmentations (colour jitter, blur, grayscale) to both images and homographic augmentations to the second image.

Models are trained for 10 epochs using Stochastic Gradient Descent (SGD), with base learning

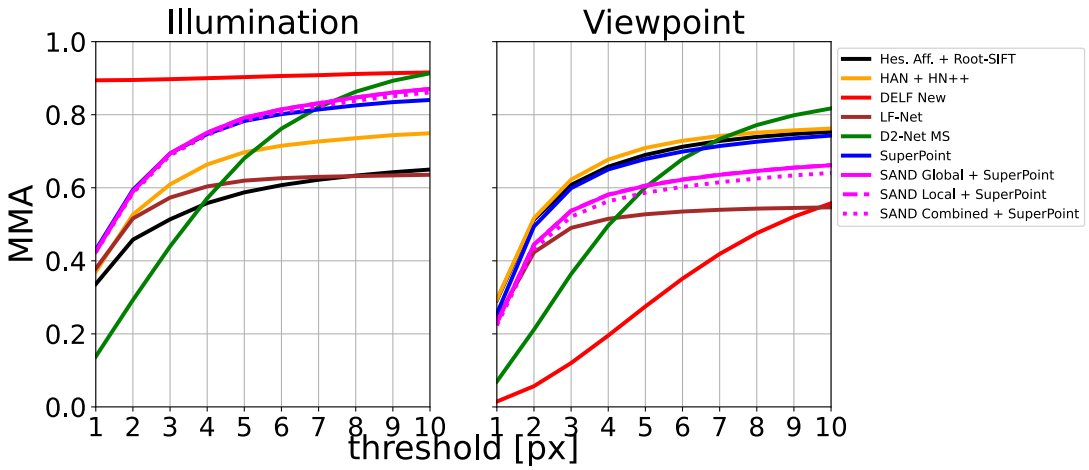


Figure 3.5: HPatches Image Matching Evaluation. We evaluate Mean Matching Accuracy when reprojecting predicted matches using different correctness thresholds. The proposed *SAND* features are robust to illumination changes, outperforming most approaches after 6 px. Training data, however, restricts viewpoint invariance.

rate 10^{-3}, a momentum of 0.9 and 10^{-5} weight decay. We find VGG [169] pretrained on ImageNet [39] to perform best, with descriptor dimensionality $n_{dim} = 128$ and a downsampling factor $f = 8$. As is common practice, descriptors are L_2 normalized during both training and evaluation. PixCon typically uses a margin $m = 0.5$ and NT-Xent a temperature $\tau = 0.1$. Finally, we use the spatial negative mining strategies **G**, **L** & **GL** as defined previously.

3.2.1 Image Matching

We first perform a feature centric evaluation to determine the quality of the learnt representations. This focuses on matching uniqueness across lighting and viewpoint changes, specifically in the task of correspondence estimation. To do this, we follow the procedure from D2-Net [45] by evaluating Mean Matching Accuracy (MMA) on the HPatches dataset [9]. The dataset consists of multiple sequences with either viewpoint or illumination changes, where the features detected in the first image are progressively matched to each frame in the sequence. The MMA is computed as the percentage of correct matches given a varying threshold for the reprojection error for each keypoint. Since *SAND* does not provide keypoints, we make use of those detected by SuperPoint [41].

As seen in Figure 3.5, the proposed features are robust to illumination changes, outperforming most other approaches after the 6 pixel mark. Unfortunately, due to the lack of extreme viewpoint

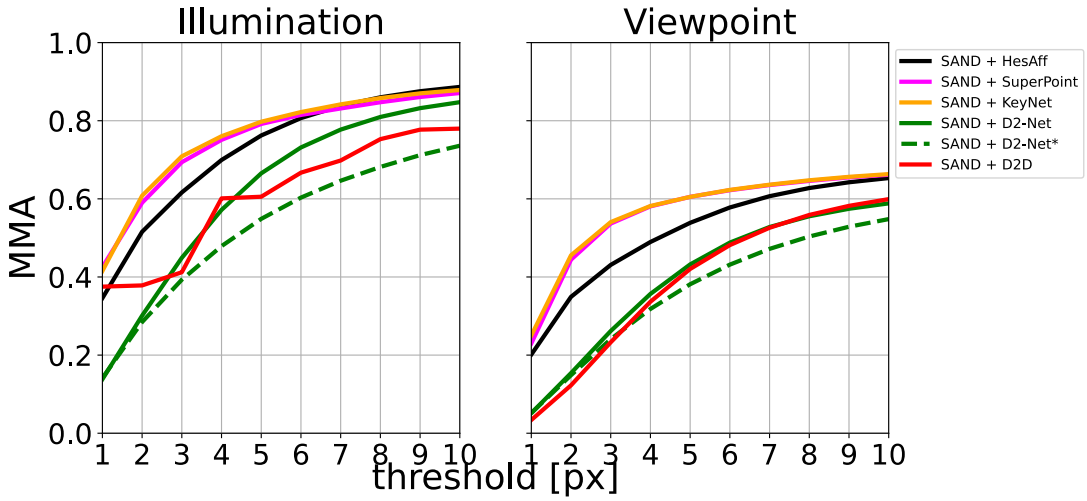


Figure 3.6: HPatches Keypoint Ablation. We combine *SAND* with various keypoint detectors and show the impact on the performance. Learned detectors (SuperPoint, Key.Net) outperform hand-crafted (HesAff) and *describe-then-detect* (D2-Net, D2D) detectors. *SAND* + D2-Net* represents the case where we apply the D2-Net algorithm to the learnt *SAND* features.

Table 3.1: HPatches Keypoint Ablation. We report the average number of matches obtained per Illumination/Viewpoint sequence in Figure 3.6. Key.Net produces less matches than SuperPoint, but has a similar performance. On the other hand, D2-Net produces a much larger number of matches, but they tend to be less accurate.

	Illumination	Viewpoint
SAND + HesAff [118]	1609.55	2987.45
SAND + SuperPoint [41]	703.70	1093.32
SAND + Key.Net [88]	619.09	482.60
SAND + D2-Net [45]	2429.36	3793.72
SAND + D2-Net* [45]	1348.43	1873.15
SAND + D2D [179]	272.04	458.28

changes in the training dataset, *SAND* is not as robust to viewpoint variation. It is also worth noting that this evaluation is carried out using MNN. In practice, downstream tasks such as those in Section 3.3.4 can use more advanced matching [122, 156] or outlier rejection [53, 12] schemes to filter out many of the incorrectly predicted matches.

Keypoint Ablation. Here we combine *SAND* with various keypoint detection algorithms. As

shown by the results in Figure 3.6, the choice of keypoint has a great impact on the final performance. SuperPoint [41] and Key.Net [88] provide the best MMA, especially at lower reprojection error thresholds. However, as shown by Table 3.1, Key.Net produces a lower number of matches.

We also explore various *describe-then-detect* approaches. In the case of D2-Net [45], we show two variants. The first (*SAND* + D2-Net) uses the keypoints detected by the trained features from D2-Net. The second (*SAND* + D2-Net*) instead applies the fixed detection algorithm proposed by D2-Net to the trained *SAND* features. We also use D2D [179] based on the absolute and relative metrics of saliency for each descriptor. In all cases, we find these approaches underperform traditional keypoint detection methods. This is partially due to the fact that *describe-then-detect* approaches operate on the downsampled feature maps, resulting in keypoints that are not accurately localised. As shown by the drop in performance in *SAND* + D2-Net*, training the features along with the keypoint detection is also likely to provide more stable keypoints that improve performance.

3.2.2 Ablation Study

To validate our choice of parameters we show an ablation study performed on the Kitti Odometry [60] dataset. In this case, correspondences are obtained via ground truth LiDAR and odometry data. We use sequence 00 for training and 03 for evaluation. Similarly to the previous section, we evaluate performance through the MMA at varying thresholds.

Losses. We first explore two different negative sampling strategies: *sample* and *pairwise*. *Sample* is akin to the strategy from [160], where each positive correspondence samples a fixed number of new random negative points. This allows sampling from regions regions of the image without ground truth LiDAR data. However, it leads to a large increase in memory requirements. The *pairwise* sampling strategy instead resembles that used by [181, 121]. In this case, only positive pairs are sampled and the remaining pairs in the image are used to generate the negative pairs. This provides an effective way of scaling the number of negatives as $\binom{N}{2}$. We evaluate both sampling modes with both losses introduced in Section 3.1.2, PixCon and NT-Xent. As shown in Figure 3.7a, the *Pairwise*-NT-Xent combination performs best, since it provides an effective way of both sampling and weighting negative pairs.

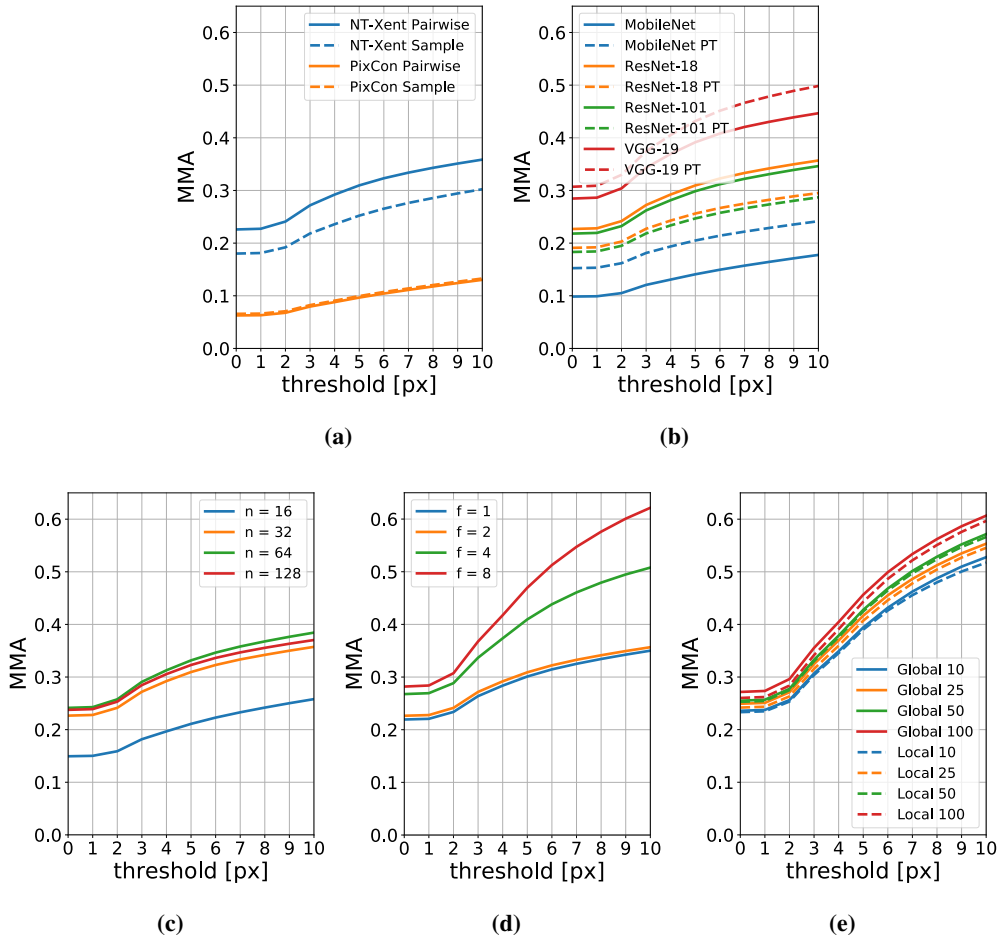


Figure 3.7: Kitti Ablation Study. We study the components of the proposed approach, namely losses, architecture, feature dimensionality and feature resolution. Interestingly, we find feature resolution to be the component that most affects performance. The proposed loss and negative training method also provide a large boost in performance.

Architecture. Regarding the feature network architecture, we test encoders corresponding to ResNet [70], VGG [169] and MobileNet-v2 [154]. We optionally make use of ImageNet [39] pretrained models (PT). Figure 3.7b shows the results of these tests. As expected, increasing the network size typically results in improved performance. However, ImageNet pretraining leads to mixed results, with ResNet models performing significantly worse than their counterparts trained from scratch.

Descriptor dimensionality. When increasing the dimensionality of the learned descriptor, we notice a large performance gap at the lower levels. However, Figure 3.7c suggests that this effect seems to plateau rather rapidly as the dimensionality increases.

Dense descriptor resolution. This controls the downsampling factor of the output dense descriptor maps w.r.t. the input image. The results can be found in Figure 3.7d. Surprisingly, this is the factor that contributes most to the performance of the descriptors. Furthermore, performance is improved at increased levels of downsampling. These versions tend to produce a slightly lower number of matches, but of better quality. As shown in [117], larger downsampling factors lead to an improved global consistency due to the overall increased receptive field, in exchange for a loss of finer detail. In the context of correspondence estimation, we theorize that as long as the keypoints are in the correct location, the global consistency has sufficient discriminative power.

Spatial negative scale. Here we regulate the scale used to select negative samples for each point. The results in Figure 3.7e use two different configurations: Global and Local. In the Global case, we are setting the minimum distance κ_{min} from the original correspondence, whereas Local defines the maximum distance κ_{max}. It is worth noting that neighbouring pixels have highly overlapping context regions due to the receptive field of the network. This can lead to very similar descriptors that are hard to differentiate. As such, when using Local strategies it is important to select large enough thresholds that account for this. However, we find that Global strategies tend to perform better in the task of correspondence estimation. We believe this is due to the fact that this strategy provides a broader context to the features that allows them to be more discriminative when matching points across the whole image.

3.3 Downstream Tasks

The original motivation for *SAND* was to develop generic deep features useful in a wide range of tasks. The previous evaluation focused on correspondence estimation, whereas now we will directly evaluate *SAND* features across a range of different tasks without additional training at the feature level. We further show how the different spatial negative mining schemes can be leveraged by different tasks.

The target tasks are chosen to cover many common computer vision applications, namely disparity estimation, semantic segmentation, visual localization and SLAM. Disparity estimation and semantic segmentation represent classic deep learning tasks, where we replace intermediate

deep features, process them in a feature matching cost volume and produce a dense regression or classification, respectively. Visual localization is performed in an end-to-end manner by replacing the input image to a network with its dense feature representation. In this case, the objective is to regress holistic translation and rotation. Finally, SLAM is another classic use of correspondence estimation, where we directly replace sparse hand-crafted features.

It is worth noting that once again the evaluation for these tasks takes place on a completely different dataset to that used during training. Visual localization is performed on the Cambridge Landmarks dataset [80], while the rest make use of the various Kitti [60] subsets.

In the following sections we show results corresponding to two different variants. The original models published in *SAND-v1* [172] correspond to models trained using ResNet-18 with an SPP, $n_{dim} = 32$, $f = 1$ and the PixCon loss. Meanwhile, the updated version *SAND-v2* is trained with VGG-19, $n_{dim} = 128$, $f = 8$ and NT-Xent.

3.3.1 Feature Matching Cost Volume

Inspired by [24], disparity estimation and semantic segmentation uses a cost volume $\mathbf{V}$ to combine the dense features from stereo pairs. The features are concatenated across each disparity level $\hat{d}$ through

$$\mathbf{V}\left(x, y, \hat{d}, z\right) = \begin{cases} \mathbf{F}_1\left(x, y, z\right) & \text{if } z \leq n_{dim} \\ \mathbf{F}_2\left(x + \hat{d}, y, z\text{-}n_{dim}\right) & \text{otherwise} \end{cases}, \tag{3.13}$$

resulting in a cost volume of size $h \times w \times N_{\hat{d}} \times 2n_{dim}$ which maps from a 4-D index to a single value, $\mathbf{V} : \mathbb{N}^4 \mapsto \mathbb{R}$. This represents an application agnostic extension to traditional cost volumes [54]. The following layers in the network can either compute traditional pixel-wise distances or aggregate multi-scale information to improve robustness to viewpoint changes.

The cost volume is fed to a 3-D stacked hourglass network with skip connections, including connections from the early feature extraction layers in the dense descriptor network. To demonstrate the generality of this approach we use the cost volume for two very different tasks. Stereo disparity estimation represents a classical use of these techniques, while semantic segmentation usually makes use of a single image. We adapt the network to this purpose by modifying the final layer to produce the desired number of segmentation classes.

Table 3.2: Kitti Stereo 2015 Disparity Estimation Evaluation. We report the percentage of incorrect pixels (3-px abs. and 5% rel. error). *SAND-v1* variants trained on the full image resolution outperform the baseline consistently.

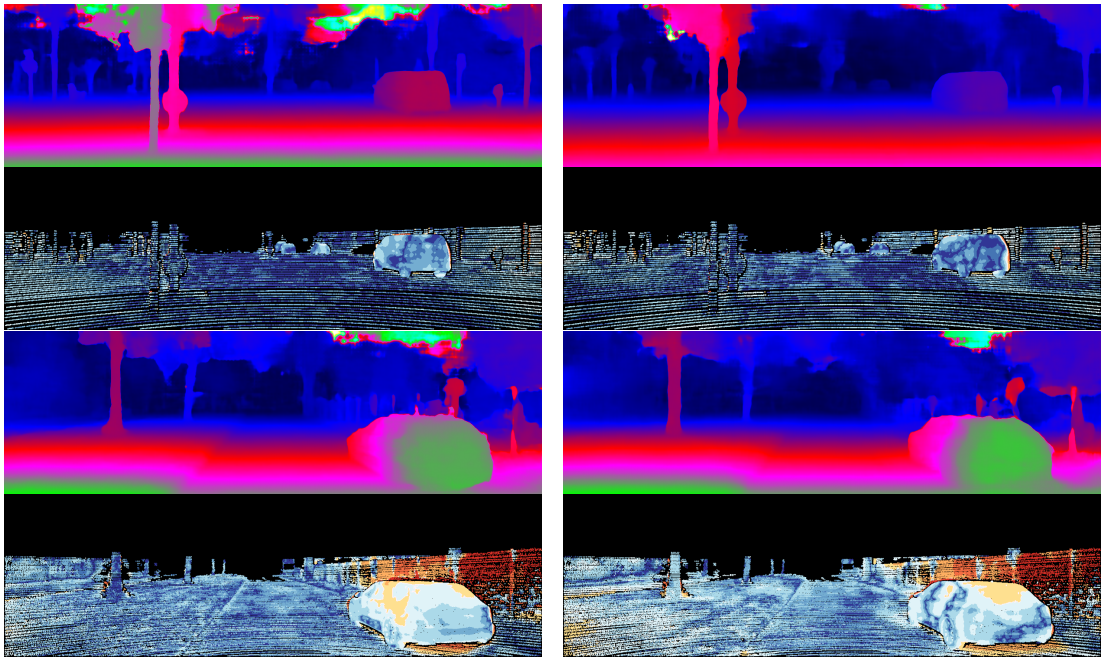
	Train (%) ↓	Eval (%) ↓
Baseline [24]	1.49	2.87
SAND-v1-G	**1.05**	**2.65**
SAND-v1-L	1.09	2.85
SAND-v1-GL	1.06	2.79
SAND-v2-G	2.44	5.41
SAND-v2-L	2.41	5.35
SAND-v2-GL	2.46	5.53

3.3.2 Disparity Estimation

In this experiment we compare our disparity estimation procedure to that from PSMNet [24], which is trained on Kitti Stereo 2015 for 600 epochs. As previously mentioned, the proposed approach replaces the intermediate feature extraction network with a dense *SAND* model and is trained for 450 epochs. *SAND-v1* variants include descriptor finetuning at a lower learning rate for the last 250 epochs.

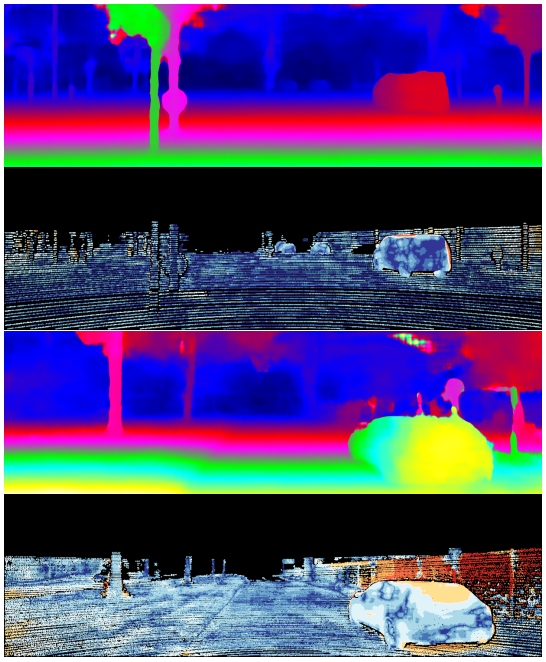
Both approaches are evaluated based on the original Kitti evaluation scripts, shown in Table 3.2. This reports the percentage of incorrect pixels based on relative and absolute errors of 3 pixels and 5%, respectively. As seen, the *SAND-v1* variants outperform the baseline. However, we find a decrease in performance when using *SAND-v2*.

This demonstrates the fact that different tasks have different feature requirements. As previously discussed, feature resolution represents a trade-off between global consistency and local detail. In a task such as disparity estimation, where the fine detail is more discriminative, it is natural for the original version from *SAND-v1* to outperform the downsampled *SAND-v2* variants. From the visualizations in Figure 3.8, it is apparent that most of the error is accumulated on borders and thin objects.

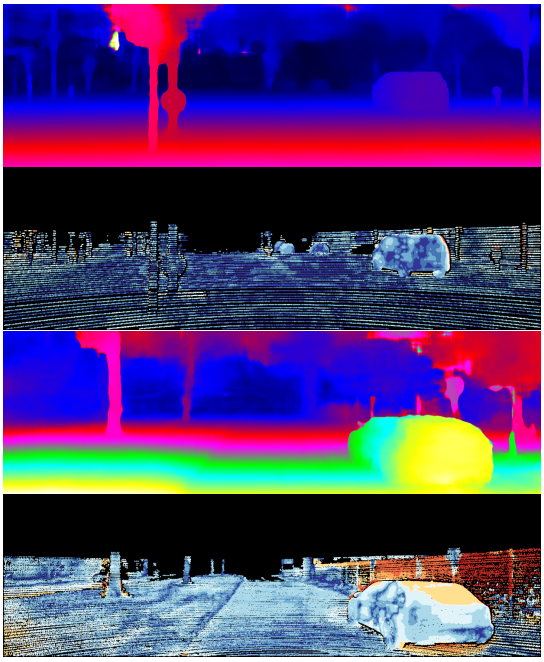


(a) Baseline

(b) *SAND-v2* G



(c) *SAND-v2* L



(d) *SAND-v2* GL

Figure 3.8: Kitti Stereo 2015 Disparity Estimation Visualization. Despite the difference in metrics (Table 3.2), visualizations show the performance to be comparable. As seen, most of the error occurs around object borders, likely caused by the decreased resolution of *SAND-v2* features.

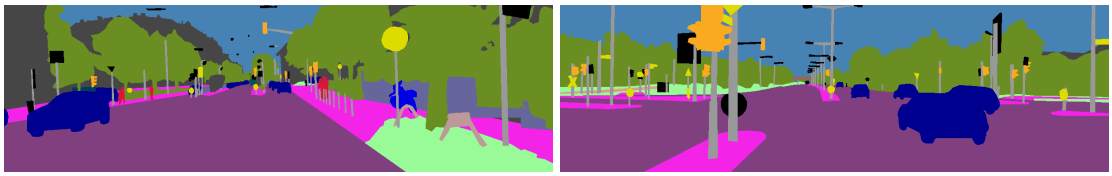
Table 3.3: Kitti Semantic Segmentation Evaluation. The proposed changes from *SAND-v2* improve on both the baseline and our previous results from *SAND-v1* [172] This is particularly noticeable in complex categories such as Human and Vehicle.

	Class ↑	Cat. ↑	Flat	Nature	Object	Sky	Const.	Human	Vehicle
Baseline	29.3	53.8	87.1	78.1	30.1	63.3	54.4	1.6	62.1
SAND-v1-G	31.1	55.8	**87.3**	**78.5**	36.0	59.8	57.5	6.7	66.8
SAND-v1-GL	29.4	51.7	85.1	76.6	33.8	51.8	54.4	4.3	56.3
SAND-v2-G	31.4	**60.0**	85.9	74.7	30.6	85.5	**62.2**	**9.0**	**71.8**
SAND-v2-L	31.5	58.3	85.3	74.5	27.5	**88.8**	59.5	4.0	68.3
SAND-v2-GL	**31.8**	58.8	84.5	73.4	30.9	88.3	59.3	6.2	69.3

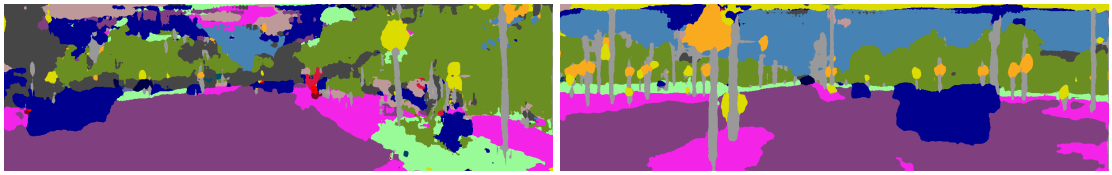
3.3.3 Semantic Segmentation

Once again, this approach is based on the cost volume presented in Section 3.3.1, with the final layer predicting a 19-class segmentation. All models are trained on the Kitti pixel-level semantic segmentation data for 600 epochs. As with disparity estimation, the baseline trains the full network from scratch, while *SAND* variants use their respective pretrained model.

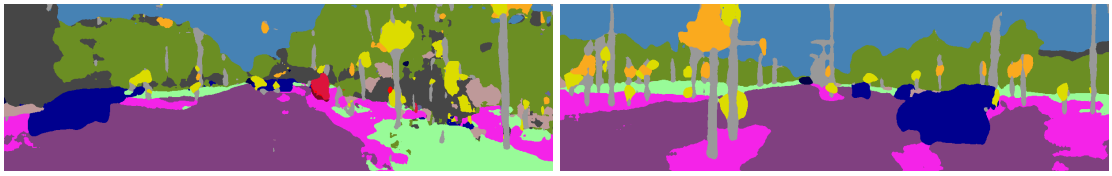
Results are obtained from the Kitti evaluation script, which reports the mean-Intersection over Union (m-IoU) for each class and category. For the sake of brevity, Table 3.3 shows only the breakdown at the category level. Both *SAND* variants lead to improvements over the baseline, with *SAND-v2* being better overall. The additional changes from this version improve performance especially in complex foreground categories such as Vehicle and Human. The visualizations in Figure 3.9 also show how the consistency in predictions for large instances is also improved.



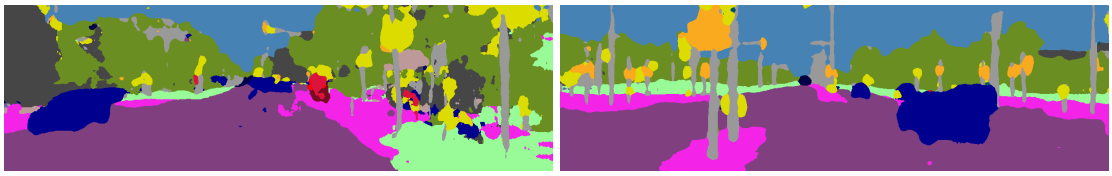
(a) Ground Truth



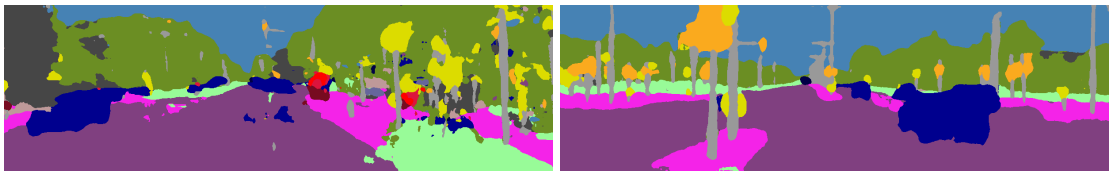
(b) Baseline



(c) *SAND-v2* G

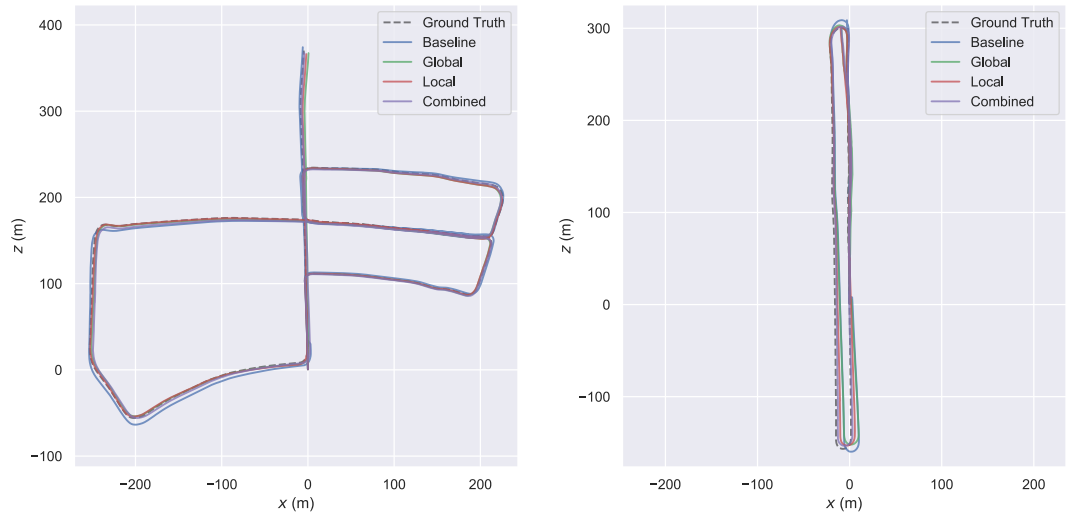


(d) *SAND-v2* L



(e) *SAND-v2* GL

Figure 3.9: Kitti Semantic Segmentation Visualization. *SAND* improves prediction consistency, as evidenced by the sky region in the shown images. As with disparity estimation, however, downsampling affects edges and thin objects.



(a) 05 **(b)** 06

Figure 3.10: Kitti Odometry SLAM Trajectory Visualizations. Variants using *SAND* features drift less and provide more robust trajectories overall. This can be seen in the bottom region of both trajectories shown.

3.3.4 SLAM

The next target task is SLAM. This represents another traditional use for feature descriptors, where correspondences are used to estimate the change in pose between consecutive frames, as well as triangulate the overall position in a map. As such, this is another case where hand-crafted approaches still dominate. We make use of S-PTAM [138] a stereo SLAM algorithm using ORB [148] descriptors. We can therefore simply replace those with different *SAND* versions, keeping the same keypoint detector and backend optimizer. Matching is performed via MNN between L_2 normalized descriptors.

We use the whole of Kitti Odometry and report Absolute and Relative Pose Error in Table 3.4. The variants using *SAND-v1* tend to outperform the baseline, with the other trajectories being comparable. This can also be seen in the accuracy of the predicted trajectories shown in Figure 3.10. Unfortunately, *SAND-v2* suffers from viewpoint changes, which results in undetected loop closures and accumulated drift errors. However, it can be seen how the hierarchical version performs best in both cases. This is due to the fact that it is able to effectively combine information from multiple scales, improving both VO and loop closure.

Table 3.4: Kitti Odometry SLAM Evaluation. We report Absolute and Relative Pose Error for each trajectory, excluding 01 due to lack of alignment for all approaches. In this case, *SAND-v1* provides larger improvements in trajectories containing loop closures, whereas *SAND-v2* improves the remaining trajectories.

	00		02		03		04		05		06		07		08		09		10	
	APE ↓	RPE ↓	APE	RPE	APE	RPE	APE	RPE	APE	RPE	APE	RPE	APE	RPE	APE	RPE	APE	RPE	APE	RPE
Baseline [138]	5.63	0.21	**8.99**	**0.28**	6.39	0.05	**0.69**	**0.04**	2.35	0.12	3.78	0.09	1.10	0.19	**4.19**	0.13	5.77	0.43	2.06	**0.28**
SAND-v1-G	13.09	0.21	41.65	0.36	6.00	0.08	6.43	0.13	6.59	0.16	9.10	0.13	2.05	0.21	15.40	0.17	11.50	0.45	18.25	0.35
SAND-v1-L	5.99	0.21	9.83	0.29	4.40	0.04	1.13	0.05	2.37	0.12	2.54	0.09	0.88	0.19	5.26	0.13	6.25	0.42	2.03	0.30
SAND-v1-GL	**4.84**	0.20	9.66	0.29	3.69	0.04	1.35	0.05	1.93	**0.11**	2.00	**0.08**	0.96	0.19	6.00	0.13	5.48	0.42	**1.36**	0.29
SAND-v2-G	7.87	0.20	18.34	0.33	2.15	0.06	2.99	0.07	2.34	0.13	2.66	0.11	0.68	0.20	8.50	0.15	**5.18**	0.42	4.16	0.31
SAND-v2-L	7.70	0.20	21.22	0.34	**1.88**	0.05	3.00	0.07	**1.53**	0.13	2.12	0.10	**0.57**	0.20	9.36	0.15	6.33	0.42	3.83	0.32
SAND-v2-GL	6.54	0.20	16.81	0.32	1.96	0.06	2.78	0.07	1.95	0.13	**1.81**	0.10	0.83	0.21	10.61	0.15	6.56	0.42	3.93	0.31

Table 3.5: Cambridge Landmarks Localization Evaluation. Results show the MSE for both **T**ranslation(meters) and **R**otation (radians). The feature representations learnt by *SAND* help to improve localization accuracy, most notably in term of position. We improve our original results from [172].

	GreatCourt		KingsCollege		OldHospital		ShopFacade		StMarysChurch		Street	
	$T \downarrow$	$R \downarrow$	T	R	T	R	T	R	T	R	T	R
Baseline [80]	10.30	0.35	1.54	0.09	**3.14**	0.10	2.22	0.19	**2.77**	0.22	**22.60**	**1.01**
SAND-v1-G	11.46	0.30	1.62	0.09	3.30	0.11	2.20	0.25	3.67	0.23	31.92	1.24
SAND-v2-G	6.79	0.23	1.44	**0.08**	3.27	**0.09**	1.34	0.18	3.14	**0.21**	26.35	1.06
SAND-v2-L	**5.43**	0.24	1.28	**0.08**	3.75	0.10	1.18	**0.17**	3.19	0.22	25.74	1.12
SAND-v2-GL	6.52	**0.21**	**1.24**	**0.08**	3.24	**0.09**	**1.09**	0.21	3.02	0.23	29.71	1.11

3.3.5 Visual Localization

The final downstream task is visual localization. This is a task still currently dominated by hand-crafted approaches. However, there have recently been some efforts to perform this in an end-to-end fashion [80]. These approaches represent a different type of task to those presented so far. In this case, the objective is to perform a holistic regression of 6-Degrees of Freedom (DoF) pose given an input image. The baseline represents the original approach trained from scratch using the Cambridge Landmarks [80] dataset. Meanwhile, *SAND* variants replace the input image to the network with its learned dense representation. This is upsampled to the original image resolution, with the first network layer modified to accept the required number of channels. Each scene in the dataset is trained for 100 epochs with a constant learning rate. It is worth noting that none of the *SAND* versions have additional training at the feature level.

The results in Table 3.5 show the median Mean Squared Error (MSE) for each scene for both **T**ranslation (meters) and **R**otation (radians). As seen, *SAND-v1* produces comparable results to the baseline without feature finetuning. However, the various improvements introduced by *SAND-v2* result in a drastic error reduction. This is most notable in sequences such as GreatCourt and ShopFacade, where the translation error is approximately halved. We find the multi-scale **GL** descriptors to slightly outperform the remaining *SAND* variants. Once again, this illustrates the benefits of incorporating multiple negative mining scales, allowing the downstream task

to incorporate both global and local features into its predictions. One of the main limitations of PoseNet + *SAND* is the decrease in performance in sequences with high self-similarity. In this case, the global feature representation cannot be reliably correlated with the exact position. Future work could explore the incorporation of additional temporal constraints to refine the predicted locations.

3.4 Conclusion

In this chapter we targeted thesis objectives 1 & 3, exploring dense feature learning in the context of both correspondence estimation and a wider range of computer vision tasks. To this end, we proposed *SAND* and introduced the concept of spatial negative mining. Complementary to hard negative mining, spatial mining can result in a hierarchical aggregation of the context information visible to each pixel during the training process. This allowed us to embed different properties into the feature space which might be useful for different target downstream tasks.

We evaluated these features in different computer vision tasks, each requiring different properties, including dense regression and classification, holistic regression and correspondence estimation. We showed how in each of these cases, the proposed solution outperforms the baselines and provides consistent improvements.

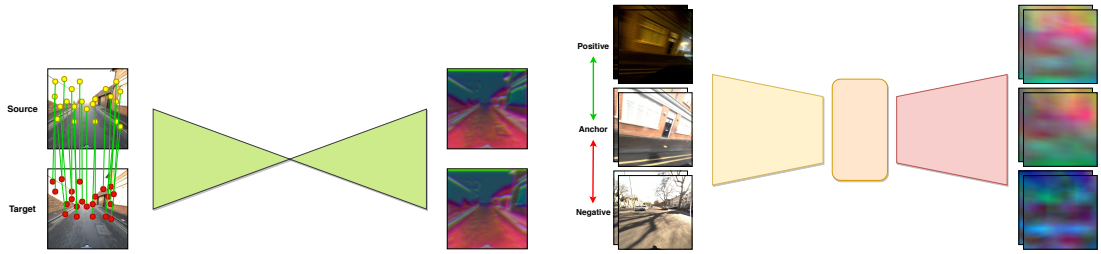
However, a common limitation seen in *SAND* and other methods present in the evaluation is the overwhelming bias towards ideal daytime conditions. Most of the commonly used datasets for training and evaluating lack data for adverse and challenging conditions such as night-time, rain or snow. Even if this data is present, training a supervised feature learning network is non-trivial due to the difficulties in obtaining accurate ground truth. We will discuss and overcome these limitations in the next chapter.

Chapter 4

Weakly Supervised Seasonal Features

As we saw in the previous chapter, generic dense features can be adapted to work in *non-metric* deep learning tasks. However, the limitations in the training and evaluation procedures bias systems towards ideal daytime conditions. In real-world problems, the deployed systems must be capable of operating regardless of the current seasonal and weather conditions. Despite this, many of the common training datasets [35, 60, 75] do not include such variation, exhibiting heavy biases towards clear daytime conditions. Even the few datasets which exhibit temporal variance [107] are unable to provide the accurate pixel-level correspondence ground truth needed to train techniques such as *SAND*. This chapter addresses the challenging task of learning general purpose dense pixel-wise feature descriptors without requiring pixel-wise correspondence ground truth. This makes it possible to learn feature descriptors robust to these changes, including both short-term (day *vs*. night) and long-term (summer *vs*. winter) variation.

Even if seasonal variation is present in the dataset [107], it is still non-trivial to train existing feature learning approaches on this data. As discussed in Chapter 2, most of these approaches rely on ground truth pixel-wise correspondences between images. When this data is collected over a long period of time, VO drift and Global Positioning System (GPS) inaccuracies can lead to pointcloud misalignment errors of the order of tens of pixels. Self-supervised approaches [41, 34] do not suffer from this problem since they make use of homographic augmentations. However, this comes at the cost of not showing the network real world variation, which was shown to reduce robustness in the previous chapter.



(a) Traditional **(b)** Proposed

Figure 4.1: *Déjà-Vu* Feature Learning. Descriptor learning is commonly framed as a supervised problem based on ground truth geometric correspondences. Obtaining this ground truth in cross-seasonal environments is highly challenging. We propose a method for leaning dense features from image-level labels indicating if images correspond to the same location.

We propose a middle ground making use of real images with weak supervision in the form of image-level relationships of (dis)similarity, *i.e*. if the images correspond to roughly the same location in the world. No correspondence ground truth is required between the contents of the two images. Unlike the pixel-wise correspondences of the previous chapter, it is trivial to automate the process of obtaining these weak image-level annotations by simply comparing GPS measurements. We then train the network to produce globally *similar* dense feature maps for images corresponding to the same location. This is based on the intuition that, in a positive pair, each descriptor should have one and only one matching feature in the second image. Since this compares the feature maps in a dense manner, we do not require the image pairs to be aligned. As such, our approach is more robust to inaccurate ground truth compared to other approaches based on camera pose and epipolar geometry supervision [194]. We refer to our approach as *Déjà-Vu* features [173]. In the context of this thesis, this helps address objectives 1 & 2, since we aim to reduce the amount of labels required to train dense feature approaches.

4.1 Methodology

As with *SAND*, the objective is to produce a dense feature description of the input image. However, this is achieved by using only weak supervision. This process is illustrated in Figure 4.2. We employ the same network as previously discussed in Section 3.1.1. To summarize, this is an FCN encoder-decoder network mapping $\Phi_{\mathbf{F}} : \mathbb{N}^{h \times w \times 3} \mapsto \mathbb{R}^{h/f \times w/f \times n_{dim}}$, with n_{dim}

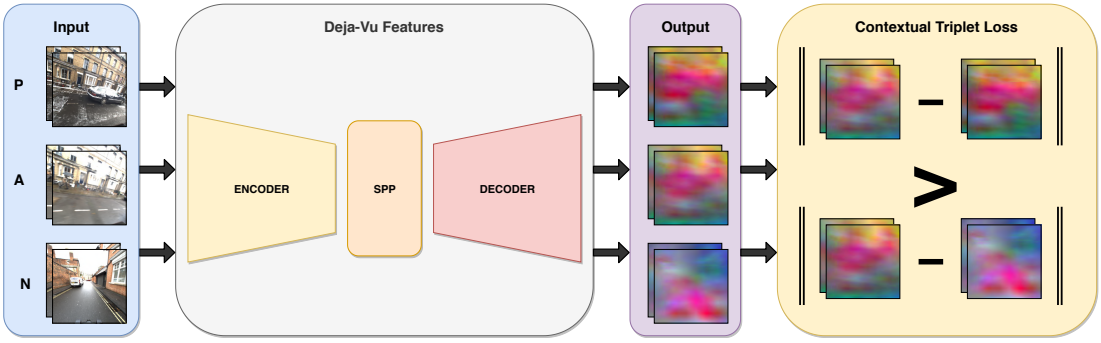


Figure 4.2: Proposed *Déjà-Vu* Overview. We train the network with image-level similarity labels, consisting of consecutive **A**nchor - **P**ositive - **N**egative triplets. The loss computes the global similarity based on the average discriminative power of each descriptor in a dense feature map, without requiring spatial alignment between images or geometric correspondences.

as the descriptor dimensionality and f as the target downsampling factor. The network also incorporates skip-connections between correspondingly sized layers of the encoder and decoder.

4.1.1 Aligned Pixel-wise Contrastive Loss

Déjà-Vu is a weakly supervised approach. However, we fist provide a supervised baseline based on the PixCon loss from (3.3) introduced in the previous chapter. As discussed, the pixel-wise correspondences required for training are usually obtained by reprojecting LiDAR points onto images according to the reprojection equation (3.9). Obtaining accurate ground truth data in cross-seasonal environments is highly challenging. We therefore opt for a synthetic dataset, described in more detail in Section 4.2.3, to train this baseline.

Given a synthetic dataset, it is possible to create perfectly aligned cross-seasonal environments, including both static and dynamic objects. In other words, the camera position and scene contents will be identical between the different seasons, meaning that every pixel has a ground truth correspondence with the same pixel in the other image. This allows us to introduce the Aligned PixCon loss using a cross-seasonal triplet $\mathbb{T}_X = \{\mathbf{I}_A, \mathbf{I}_P, \mathbf{I}_N\}$. Given the anchor image $\mathbf{I}_A$, we define the positive sample $\mathbf{I}_P$ as an image with the same location, but a different season. Meanwhile, the negative sample $\mathbf{I}_N$ comes from a different location and any season.

The alignment of this synthetic dataset allows us to simplify the summation over correspondence pairs from equations (3.3), replacing it with elementwise matrix subtraction. We can thus define

the Aligned PixCon loss as

$$\ell_{a\text{-}con}\left(y, \mathbf{F}_1, \mathbf{F}_2\right) = \begin{cases} \|\mathbf{F}_1\text{-}\mathbf{F}_2\|_2^2 & \text{if } y = 1 \\ {}^{+}(m\text{-}\left\|\mathbf{F}_1\text{-}\mathbf{F}_2\right\|_2)^2 & \text{if } y = 0 \end{cases}, \tag{4.1}$$

where we are computing the L_2 distance between pairs of feature descriptors at corresponding spatial locations and y, in this case, is the image-level label indicating whether the images correspond to the same location. Given the cross-seasonal triplet $\mathbb{T}_X$, the final loss is obtained through

$$\mathcal{L}_{a\text{-}con}\left(\mathbb{T}_X\right) = \ell_{a\text{-}con}\left(1, \mathbf{F}_A, \mathbf{F}_P\right) + \ell_{a\text{-}con}\left(0, \mathbf{F}_A, \mathbf{F}_N\right), \tag{4.2}$$

where again only image-level labels are required.

4.1.2 Contextual Similarity

The proposed Aligned PixCon loss is only applicable if the cross-seasonal pairs are perfectly aligned. This is practically impossible in the case of real world data, since it would require perfect synchronization of sensors across multiple runs, as well as aligned dynamic objects. We therefore require a similarity metric that can be applied to non-aligned images. We take inspiration from [115], quantifying how uniquely each feature within a source map matches to a feature in the target dense feature map.

More formally, we define two feature maps as similar if each feature descriptor in $\mathbf{F}_1$ has only one descriptor from $\mathbf{F}_2$ significantly closer than the rest in the embedding space. To measure this, we first build the distance matrix as

$$\mathbf{C}\left(\mathbf{p}_1, \mathbf{p}_2\right) = \left\|\mathbf{F}_1\left(\mathbf{p}_1\right)\text{-}\mathbf{F}_2\left(\mathbf{p}_2\right)\right\|_2, \tag{4.3}$$

which is normalized according to

$$\hat{\mathbf{C}}\left(\mathbf{p}_1, \mathbf{p}_2\right) = \frac{\mathbf{C}\left(\mathbf{p}_1, \mathbf{p}_2\right)}{\min\limits_{\mathbf{p}_2} \mathbf{C}\left(\mathbf{p}_1, \mathbf{p}_2\right)}. \tag{4.4}$$

In this case, the best match is assigned a cost of 1. The remaining costs are described as their ratio w.r.t. the best match in the range $[1, \infty)$. For instance, a point with a feature distance twice that of the best match will be assigned a cost of 2. On the other hand, if there are multiple

equally good matches, several features will have a cost of 1. Note that this is akin to SNN matching based on the ratio test [102].

These individual ratio test costs are then converted into a uniqueness measure via the softmax

$$\widetilde{\mathbf{C}}\left(\mathbf{p}_1, \mathbf{p}_2\right) = \frac{\exp\left(\frac{1\text{-}\hat{\mathbf{C}}(\mathbf{p}_1, \mathbf{p}_2)}{\tau}\right)}{\sum_{\mathbf{p}_2} \exp\left(\frac{1\text{-}\hat{\mathbf{C}}\left(\mathbf{p}_1, \mathbf{p}_2\right)}{\tau}\right)}, \tag{4.5}$$

where τ represents the softmax temperature. In the case where a feature has a single good match, *i.e*. where the lowest distance in $\mathbf{C}$ is much smaller than the second lowest distance, the resulting similarity in $\widetilde{\mathbf{C}}$ will be large. However, if multiple points have low costs $\hat{\mathbf{C}}$, they will all obtain a similarly reduced uniqueness score.

Finally, the global similarity between dense feature maps is given by averaging these uniqueness scores across all samples

$$CX\left(\mathbf{F}_1, \mathbf{F}_2\right) = \sum\!\!\!\!\!\!- _{\mathbf{p}_2} \max_{\mathbf{p}_1} \widetilde{\mathbf{C}}\left(\mathbf{p}_1, \mathbf{p}_2\right), \tag{4.6}$$

where $\sum\!\!\!\!\!\!-$ once again represents the average summation. Since we are computing the average similarity for each descriptor in the feature map, the resulting metric is constrained to the range $[0, 1]$, with 1 signifying that the feature maps are identical. This allows us to quantify both the distances and uniqueness of each feature in the embedding space without requiring geometric correspondences. Note that this metric can also be used at test time to determine the likelihood of a pair of images corresponding to the same location.

4.1.3 Contextual Triplet Loss

To make use of the proposed contextual similarity measure during training, we incorporate it into a triplet framework. Note that, unlike in the Aligned PixCon loss, positive pairs should be those with a high similarity.

The base triplet loss is therefore given by

$$\ell_{cx}\left(\mathbb{T}\right) = {}^{+}(m + CX\left(\mathbf{F}_A, \mathbf{F}_P\right) - CX\left(\mathbf{F}_A, \mathbf{F}_N\right)), \tag{4.7}$$

where m is once again the target margin, in this case representing the separation in terms of global feature map similarity.

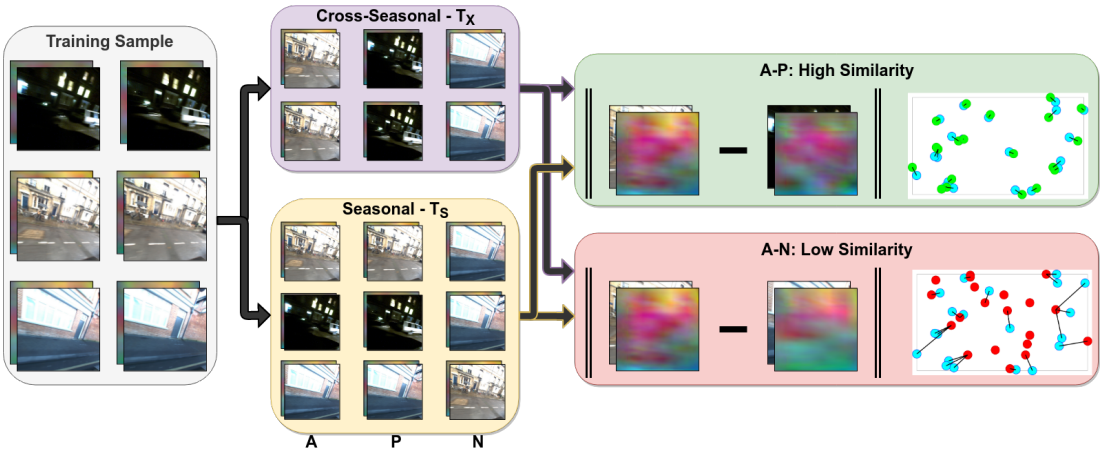


Figure 4.3: Contextual Triplet Loss Framework. Each training sample consists of two consecutive triplets. These triplet can further be combined into either seasonal or cross-seasonal triplets, aiding in short- or long-term matching, respectively. Positive pairs aim to maximise the global similarity between dense feature maps based on the assumption that each feature should only match well with one other feature in the second image.

The previously introduced cross-seasonal triplet $\mathbb{T}_X$ provides effective supervision for matching across different seasons. Unfortunately, it does not include any within-season changes. For this reason, we expand each training sample to contain the following frame for each image in the original triplet $\mathbb{T}_X^2 = \{\mathbf{I}'_A, \mathbf{I}'_P, \mathbf{I}'_N\}$. As illustrated in Figure 4.3, this leads to three new within-season triplets

$$\mathbb{T}_S^1 = \left\{\mathbf{I}_A, \mathbf{I}'_A, \mathbf{I}_N\right\}, \; \mathbb{T}_S^2 = \left\{\mathbf{I}_P, \mathbf{I}'_P, \mathbf{I}'_N\right\}, \; \mathbb{T}_S^3 = \left\{\mathbf{I}_N, \mathbf{I}'_N, \mathbf{I}_A\right\},$$

which provide additional consistency within each season's features.

Once again, we incorporate all of this information into the final loss, given by

$$\mathcal{L}_{cx}\left(\mathbb{T}_X, \mathbb{T}_S\right) = \sum_i \ell_{cx}\left(\mathbb{T}_X^i\right) + \lambda_{cx} \sum_i \ell_{cx}\left(\mathbb{T}_S^i\right), \tag{4.8}$$

where $\lambda_{cx} \in [0, 1]$ is the weight balancing the contribution of the seasonal triplets. It is worth reiterating that the simple image-level labels y are the only ground truth driving both the within season and cross-seasonal learning of these pixel-wise dense features.

4.2 Datasets

This section covers the various datasets used during both training and evaluation. We make use of two real world datasets: RobotCar Seasons [157] and UTBM RoboCar [200]. RobotCar Seasons

serves as the primary training dataset, since it contains a large amount of seasonal variation and well defined sequences from which to draw positive pairs. To show the transferability and generality of the learnt features, we carry out a similar evaluation on UTBM RoboCar. Finally, we make use of the synthetic CARLA Seasons dataset. We use this dataset to explore the Aligned PixCon loss, representing the ideal case where we have perfectly aligned cross-seasonal data. This additionally allows us to further explore the transfer capabilities of our features, bridging the *sim-to-real* gap.

4.2.1 RobotCar Seasons

The original RobotCar dataset [107] was collected by traversing roughly the same trajectory over the course of a whole year. As such, it contains a large amount of seasonal variation and environment changes caused by construction. The dataset uses a camera rig including stereo forward-facing cameras, both sides and rear viewpoints. Additionally, the dataset provides VO, GPS and LiDAR data. Unfortunately, this data is not accurate enough to provide robust cross-seasonal correspondences. Luckily, these labels are not required by *Déjà-Vu*.

RobotCar Seasons [157] is a subset of RobotCar focusing on cross-seasonal revisitations. The original trajectory is split into 49 distinct sequences, each containing roughly aligned data at several seasonal conditions, including sun, rain, dawn, overcast, dusk and night-time. We make use of 40 sequences to train our models, leaving the remaining 9 for evaluation.

The structure of this dataset provides a natural separation regarding the image-level labels required to train *Déjà-Vu*. From a given **A**nchor image, we consider images from the same sequence and a different season as **P**ositives. Meanwhile, **N**egative pairs are sampled from different sequences and any season.

4.2.2 UTBM RoboCar

The UTBM RoboCar dataset [200] provides another example of real world cross-seasonal variation, collected by traversing the same route over a long period of time. In this case, the dataset contains sequences at evening, night, sunny, cloudy and snow. It mainly targets robotics applications, so a toolkit was developed to make it easier to use in a deep learning environment

and evaluate *Déjà-Vu*. As already discussed, this dataset is used for evaluation purposes to show the ability of *Déjà-Vu* models trained on RobotCar or CARLA Seasons to transfer to unseen real world data. Once again, the dataset provides GPS, Inertial Measurement Unit (IMU) and LiDAR data, which our method does not require.

This dataset unfortunately does not provide clear sequences like RobotCar Seasons. Instead, each trajectory contains approximately 10000 sequential frames. To make the similarity computation tractable, we take 5000 frames from the middle of the sequence and downsample by keeping 1 in every 10 frames. Note that, despite this downsampling, the final evaluation set still consists of approximately 3.1 million image pairs. In order to create ground truth image level correspondence labels for evaluation we make use of the GPS data. We define true positives as pairs with a distance of 10 meters or less and an orientation change smaller than 60°.

4.2.3 CARLA Seasons

Finally, we make use of the CARLA Seasons synthetic dataset. In recent years, simulation has become a crucial tool to generate data that is challenging or impossible to obtain in real life. The CARLA Seasons dataset is composed of 10 unique environments, each run representing one minute of driving at 20 Frames Per Second (FPS). The simulated vehicle is rigged with three forward facing stereo camera pairs, placed on the bonnet, roof and bumper. Additional labels for depth, semantic segmentations and GPS are provided. Note that in the case of *Déjà-Vu* we only require the GPS labels.

The primary objective of the dataset is to provide aligned trajectories over a wide variety of weather and seasonal conditions. Overall, it provides a total of 27 combinations, each perfectly aligned with the rest. Regarding the seasons, it contains heavy/light fog, rain, snow and rain/snow deposits. Each of these is further traversed at midday, sunset and dusk.

To make the training and evaluation more comparable with RobotCar Seasons, we choose a representative subset of seasons/weathers, as shown in Figure 4.4. We use the first four environments as training sequences and the final four as evaluation. Finally, we speed up the evaluation by using a single viewpoint—bonnet-left—from the camera rig. Regarding image pair sampling, we employ the same strategy as the one defined for UTBM RoboCar.

(a) Clear Midday **(b)** Clear Sunset **(c)** Clear Night

(d) Heavy Fog Midday **(e)** Heavy Rain Midday **(f)** Light Snow Midday

Figure 4.4: CARLA Seasons Sample Images. We show a subset of the 27 total combinations of seasonal and weather effects available in the CARLA Seasons dataset. By making use of simulated environments, perfectly aligned data is provided—including dynamic objects—which is practically impossible to obtain in the real world.

4.3 Results

Training details. As previously outlined, the main models are trained on the RobotCar Seasons dataset using all three cameras available—left, right & rear. We train for 100 epochs using SGD with 0.9 momentum and a base learning rate 10^{-3}. Similar to *SAND*, we find VGG-19 [169] pretrained on ImageNet [39] and $n_{dim} = 128$ to perform best. We set the contextual triplet loss margin $m = 0.5$ given that the similarity measure is constrained to the unit interval.

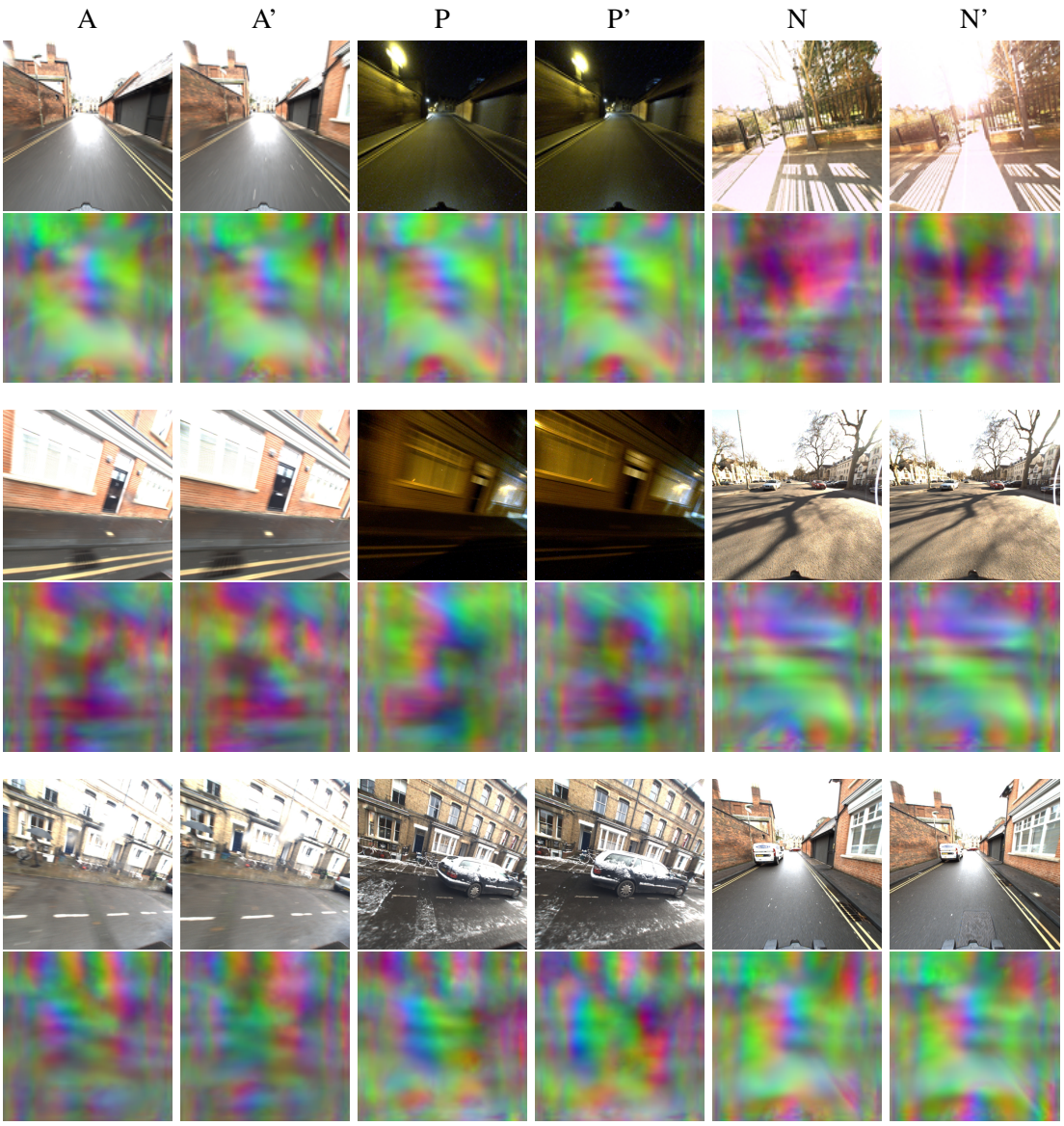


Figure 4.5: *Déjà-Vu* PCA Feature Visualization. Night-time causes drastic changes in visual appearance due to the lack of light present in the scene, as well as motion blur. *Déjà-Vu* is capable of providing a consistent dense feature representation, allowing it to identify the correct matching pair.

Feature visualization. We visualize the features by reducing their dimensionality via PCA and mapping to the RGB cube. Sample triplets are shown in Figure 4.5, illustrating some of the biggest challenges of cross-seasonal feature learning. Features must be robust to the drastic appearance changes caused by different lighting conditions. Night-time is particularly challenging due to the low light levels and motion blur resulting from an increased exposure time. Despite these challenges, *Déjà-Vu* easily distinguishes the correct positive pair.

4.3.1 Cross-Seasonal Retrieval Performance

Déjà-Vu is evaluated using the Area Under the Curve (AUC) of the ROC curve based on the predicted classification of image pairs corresponding to the same location or not. As discussed in Section 4.2, RobotCar Seasons provides a clear split based on the different sequences, each consisting of approximately 10 frames. Meanwhile, since UTBM RoboCar and CARLA Seasons provide only a continuous trajectory, we consider pairs as positive if the distance between them is less than 10 meters and the change in orientation is less than 60°. All other pairs are considered "true negatives". This is the case for both training and evaluation.

To illustrate the complexity of the problem at hand, *Seasonal* and *X-Seasonal* performance are analysed separately. In the *Seasonal* case we only evaluate on positive pairs with corresponding locations and seasons. On the other hand, pairs used in the *X-Seasonal* evaluation originate from the same location, but different seasons.

Déjà-Vu is compared to both hand-crafted and learned baselines, using the code and models provided by the respective authors. *Déjà-Vu* is primarily trained on RobotCar Seasons using the training scheme previously outlined. We further explore the use of the contextual similarity as a metric for image retrieval. To do this, we show variants of the hand-crafted features using either traditional retrieval techniques or the proposed metric from (4.6). Finally, we provide two additional models trained on CARLA Seasons using either the Aligned PixCon or contextual triplet loss. Note that, since Aligned PixCon requires perfect alignment between frames it cannot make use of the seasonal triplets $\mathbb{T}_S$. We modify the contextual loss variant accordingly by setting the seasonal weight $\lambda_{cx} = 0$. Models trained on CARLA Seasons can be identified by the † symbol.

RobotCar Seasons. Table 4.1 contains the results for RobotCar Seasons. We find that most baseline approaches tend to perform well in the *Seasonal* setting. This means that, despite not having been trained with data from multiple seasons, these approaches still provide consistent representations when evaluating in unseen conditions. Whilst this is beneficial, we find that the *X-Seasonal* evaluation nevertheless exhibits a large drop in performance for baseline systems. Meanwhile, *Déjà-Vu* provides consistent results and outperforms all baselines by a large margin under this evaluation protocol. We also note that using our contextual matching scheme greatly enhances the performance of hand-crafted features in comparison to traditional matching

Table 4.1: RobotCar Seasons AUC Evaluation. We split performance when performing image retrieval within the same (*Seasonal*) or different (*X-Seasonal*) seasons. *CX* indicates approaches using the contextual matching from (4.6). *Déjà-Vu* outperforms all baselines in the *X-Seasonal* setting by a large margin. † Trained on CARLA Seasons.

	CX	Seasonal ↑	X-Seasonal ↑
SIFT [102]		80.79	46.81
RootSIFT [7]		97.16	59.78
ORB [148]		96.72	67.58
SIFT	✓	94.49	66.11
RootSIFT	✓	95.70	68.39
ORB	✓	95.17	70.19
VGG [169]	✓	98.94	72.38
NC-Net [146]	✓	97.54	73.95
D2-Net [45]	✓	98.43	74.00
NetVLAD [6]	✓	99.41	77.56
SAND [172]	✓	**99.61**	74.86
Aligned PixCon †	✓	95.89	73.07
Déjà-Vu †	✓	97.83	83.64
Déjà-Vu	✓	99.42	**95.45**

schemes, both in the *Seasonal* and *X-Seasonal* case, improving SIFT results by 20%. The Aligned PixCon baseline performs on par with existing hand-crafted and learned approaches, despite being trained with synthetic data. However, we find that this is still outperformed by our contextual supervision. We believe this is due to the additional constraints imposed by our loss, requiring uniqueness in the feature matching and supported by the ratio test.

CARLA Seasons. Similarly, we carry out an evaluation on the synthetic CARLA Seasons dataset, as seen in Table 4.2. Overall, there is a drop in performance, indicating that CARLA Seasons is still a complex problem to solve. Once again, *Déjà-Vu* provides the best overall performance, outperforming NetVLAD in a *X-Seasonal* setting by over 10%. It is worth noting that this model was trained on RobotCar Seasons and is therefore also performing cross-dataset generalization. Surprisingly, we find that the supervised version trained with the Aligned PixCon

Table 4.2: CARLA Seasons AUC Evaluation. We evaluate on the newly proposed cross-seasonal synthetic dataset. It is worth noting that our weak supervision (Contextual) greatly outperforms even the supervised cross-seasonal baseline (Aligned PixCon). † Trained on CARLA Seasons.

	CX	Seasonal ↑	X-Seasonal ↑
SIFT [102]		73.39	56.49
RootSIFT [7]		85.32	64.29
ORB [148]		81.67	57.87
SIFT	✓	88.25	62.50
RootSIFT	✓	**88.58**	62.93
ORB	✓	78.20	64.48
VGG [169]	✓	73.82	65.71
NC-Net [146]	✓	77.69	65.52
D2-Net [45]	✓	81.80	67.77
NetVLAD [6]	✓	79.40	69.72
SAND [172]	✓	85.49	65.58
Aligned PixCon †	✓	78.82	77.68
Déjà-Vu †	✓	87.42	**87.65**
Déjà-Vu	✓	88.50	80.51

loss is outperformed by the proposed weak supervision, demonstrating the effectiveness of *Déjà-Vu*. However, as expected, the Aligned PixCon version still outperforms all other baselines.

UTBM RoboCar. We provide a final evaluation on real-world data using the UTBM RoboCar dataset. As shown in Table 4.3, *Déjà-Vu* still provides the best *X-Seasonal* performance, demonstrating the generality of the learned features and their ability to transfer to unseen real-life datasets in addition to unseen synthetic datasets. Whilst *Déjà-Vu* does not provide the best *Seasonal* performance, it is comparable. Interestingly, the best performing seasonal approaches are (Root)SIFT, which outperform all other learned baselines on this dataset. Once again we note that the proposed contextual matching approach proves effective in image retrieval scenarios even when using hand-crafted features. Finally, we note that the CARLA Seasons models do not transfer as well to the UTBM RoboCar dataset, as those trained on RobotCar

Table 4.3: UTBM RoboCar AUC Evaluation. We evaluate on a real-world cross-seasonal dataset unseen during training. *Déjà-Vu* provides the best performance, demonstrating that the learnt features transfer well to new datasets. † Trained on CARLA Seasons.

Features	CX	Seasonal ↑	X-Seasonal ↑
SIFT [102]		90.57	65.31
RootSIFT [7]		95.66	76.70
ORB [148]		93.30	62.40
SIFT	✓	95.90	73.56
RootSIFT	✓	**96.35**	75.41
ORB	✓	86.01	63.30
VGG [169]	✓	92.77	60.65
NC-Net [146]	✓	93.61	69.88
D2-Net [45]	✓	95.26	67.47
NetVLAD [6]	✓	93.73	67.20
SAND [172]	✓	93.97	68.09
Aligned PixCon †	✓	91.37	69.39
Déjà-Vu †	✓	89.90	75.19
Déjà-Vu	✓	92.38	**84.94**

Seasons. This is likely due to the challenges bridging the *sim-to-real* gap. Future work could explore the use of domain adaptation techniques to minimize the performance gap.

4.3.2 Ablation Study

Similar to *SAND* in Section 3.2.2, we perform an ablation study of *Déjà-Vu*'s components on RobotCar Seasons. The baseline results are obtained using ResNet-18 with feature resolution $f = 8$, feature dimensionality $n_{dim} = 128$, margin $m = 0.5$, seasonal consistency weight $\lambda_{cx} = 0.5$ and softmax temperature $\tau = 0.5$. These results can be found in Table 4.4.

Table 4.4: *Déjà-Vu* Ablation Study. We study components of *Déjà-Vu*, including architecture, feature dimensionality & resolution, triplet margin, seasonal weight and softmax temperature. Based on this ablation we select the parameters for the final models presented in Section 4.3.1.

	Seasonal ↑	X-Seasonal ↑
MobileNet	99.51	93.88
ResNet-18	99.12	94.26
ResNet-50	99.40	94.24
ResNet-101	99.51	94.90
VGG-11	99.42	95.08
VGG-19	**99.54**	**95.49**
$n_{dim} = 10$	98.61	92.00
$n_{dim} = 32$	99.07	93.77
$n_{dim} = 64$	99.03	93.58
$n_{dim} = 128$	**99.43**	**94.81**
$f = 4$	99.28	92.16
$f = 8$	**99.30**	**95.13**

	Seasonal ↑	X-Seasonal ↑
$m = 0.1$	99.12	93.85
$m = 0.3$	99.26	94.64
$m = 0.5$	**99.43**	**95.10**
$m = 0.7$	98.43	91.95
$m = 1.0$	97.12	88.78
$\lambda_{cx} = 0.0$	98.98	92.84
$\lambda_{cx} = 0.5$	**99.15**	**93.47**
$\lambda_{cx} = 1.0$	99.14	92.63
$\tau = 0.1$	98.76	92.24
$\tau = 0.2$	99.48	94.63
$\tau = 0.5$	**99.56**	**95.78**
$\tau = 1.0$	99.12	94.42

Architecture. We first explore the different available backbones for the encoder portion of the feature network. As with *SAND*, we evaluate MobileNet-v2, ResNet and VGG, pretrained on ImageNet. As expected, increasing the network size typically leads to a steady increase in performance, with VGG-19 being the most accurate.

Descriptor dimensionality. Regarding the dimensionality of the learnt descriptors n_{dim}, we also find similar results to the *SAND* ablation. The lowest dimensionality provides the worst results, but as the dimensionality increases this performance gap tends to shrink. This shows potential for applications where compute power and/or storage capabilities are limited, since we can provide much smaller descriptors that still perform well.

Dense descriptor resolution. This corresponds to the downsapling factor applied to the output dense feature map, with $f = 1$ corresponding to the original image resolution. In the case of *Déjà-Vu*, the contextual similarity has quadratic memory requirements, since we are computing

the similarity between each possible pair of pixels. Due to this, training and evaluating at higher resolutions becomes impractical. However, as with *SAND*, training at a lower resolution leads to better performance.

Triplet loss margin. Here we explore the margin m used in the triplet loss, controlling the target separation between positive and negative pairs. From the results, we find that it is beneficial to have an intermediate margin that is neither too strict nor too lax. A margin that is too lax ($m = 0.1$) can lead to poor separation between positives and negatives. On the other hand, too strict a margin ($m = 1$) results in an unreachable target given that we are optimizing the contextual loss. Once again, this has a pseudo-physical meaning, where a value of 1 indicates identical images and 0 indicates completely different images. As such, a target separation of 50% provides a better optimization objective.

Seasonal consistency weight. This weight λ_{cx} controls the contribution of the seasonal triplets. As seen, disabling the seasonal triplets ($\lambda_{cx} = 0$) results in a slightly decreased Seasonal performance. However, since our primary focus is cross-seasonal retrieval, we find that a balanced weight leads to better results.

Softmax temperature. Finally, we test the temperature τ regulating the harshness of the softmax. As a reminder, a lower temperature value aims to emphasize the difference between probabilities in a distribution, increasing the confidence in the matching predictions. This can be seen as increasing the SNN ratio, resulting in higher $\widetilde{\mathbf{C}}$ values on average. In the context of *SAND*, a low temperature forced the model to make predictions that would result in a better separation between positives and negatives, with the objective of performing SNN matching. However, since *Déjà-Vu* does not use ground truth correspondences, too high a temperature value may result in overconfidence in incorrect predictions, leading to an artificially high similarity value. Similarly, a higher temperature can lead to artificially low similarities. We therefore find the balanced option $\tau = 0.5$ to provide the best performance.

4.3.3 Sparse Feature Matching Performance

Déjà-Vu is trained to produce dense features based only on image-level supervision. We only need to know whether two images correspond to roughly the same location, without requiring the exact alignment between them. Despite this, we will next show how *Déjà-Vu* can still perform sparse pixel-wise descriptor matching.

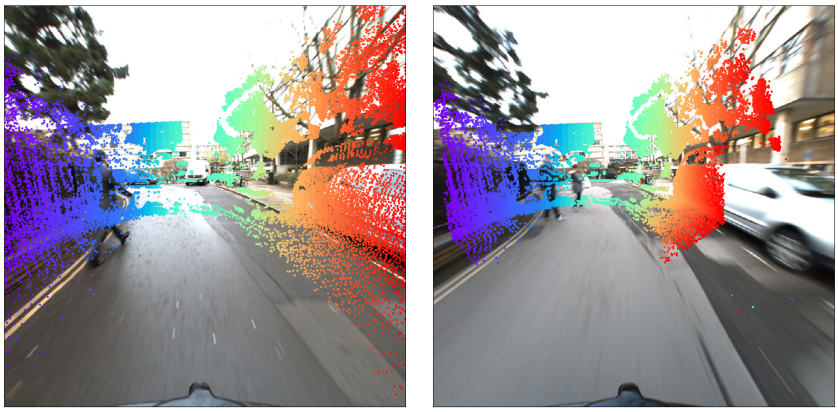


Figure 4.6: Cross-seasonal Correspondences from [90]. Despite using SOTA SfM pipelines [161], the obtained cross-seasonal correspondences are not always accurate. This includes errors due to dynamic objects (*e.g.* cars and pedestrians) as well as global misalignments (*e.g.* the corner of the building or the car). Using such correspondences for either training or evaluating is infeasible.

Recently, new datasets have been proposed containing small numbers of ground truth cross-seasonal pixel correspondences [90]. Upon closer inspection, however, it becomes obvious that this labelling is inaccurate. As shown in Figure 4.6, we find that many of the correspondences are incorrect, typically due to camera pose misalignments and dynamic objects. In the absence of a sparse cross-seasonal correspondence benchmark, we instead opt for qualitative sparse matching visualizations on RobotCar Seasons pairs, using D2-Net and *SAND* as baselines.

In the case of the D2-Net we use the keypoints detected by their *describe-then-detect* approach. Meanwhile, since *SAND* and *Déjà-Vu* do not predict keypoints, we use the well established Shi-Tomasi detector [165]. Descriptors are matched using SNN, with the final matches refined by USAC [141]. Figure 4.7 shows sparse matching performed between consecutive images within the same season. As seen, all approaches perform well in this case, showing that the learnt representations are consistent within a given season. D2-Net produces the largest number of matches, but it is worth recalling that the experiments in Section 3.2.1 showed that these do not tend to be as accurate as other methods. In contrast, Figure 4.8 performs matching between images from different seasons, mostly exhibiting the large appearance changes between day and night. In this case, the performance of *SAND* and D2-Net drops, producing a lower number of matches that are also less accurate. *Déjà-Vu* however, is capable of providing features that are more consistent across seasons. The final column in this figure shows a partial failure case,

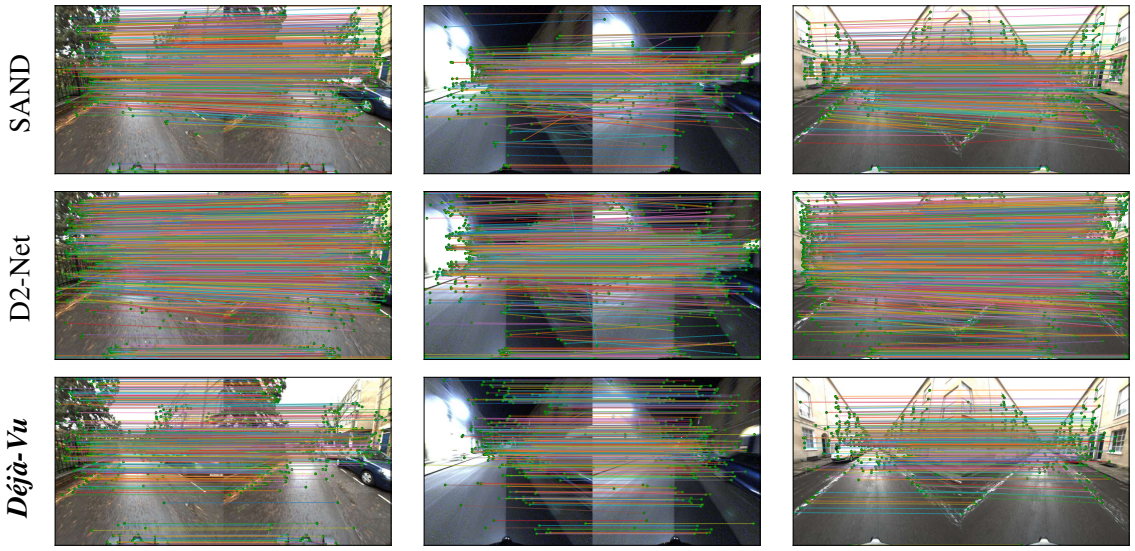


Figure 4.7: Sparse Seasonal Matching. Within season correspondences estimated by *SAND*, D2-Net and *Déjà-Vu*. All approaches provide consistent results within each season, as reflected in the retrieval experiments in Table 4.3.

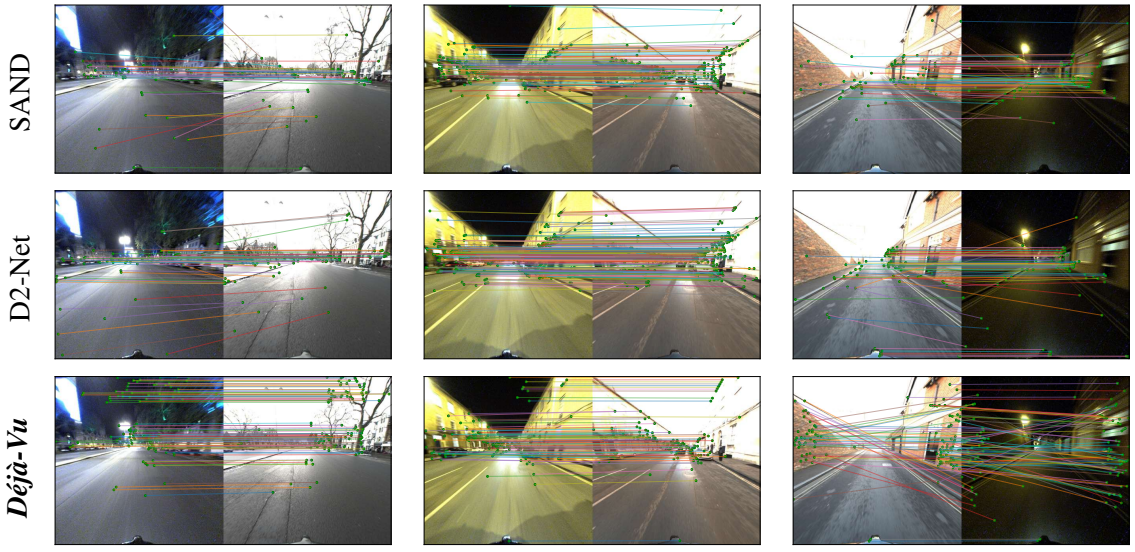


Figure 4.8: Sparse Cross-seasonal Matching. Cross-seasonal correspondences estimated by *SAND*, D2-Net and *Déjà-Vu*. Even though previous approaches can match within a given season, performance drops as we match across seasons. *Déjà-Vu* produces less matches in this scenario, but those obtained tend to be of good quality.

where a subset of the matches predicted by *Déjà-Vu* are incorrect. This is still a challenging problem that requires the incorporation of additional geometric constraints and better quality training data.

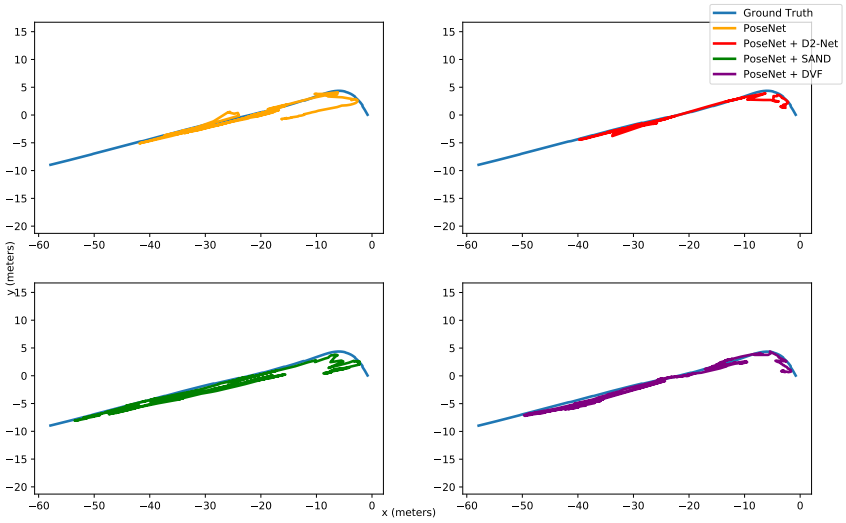


Figure 4.9: Cross-Seasonal Visual Localization. We show the predicted trajectories without any temporal smoothing. This is performed in a cross-seasonal setting, where PoseNet models using each feature baseline are trained on one season, but evaluated on a different one. *Déjà-Vu* provides a smoother trajectory with less outliers than other feature baselines.

4.3.4 Cross-Seasonal Visual Localization

Finally, we show how *Déjà-Vu* can be incorporated into a downstream computer vision task. Similar to Section 3.3.5 in the previous chapter, we use a 6-DoF relocalization pipeline based on PoseNet [80]. To illustrate the benefits of *Déjà-Vu*, we do this in a cross-seasonal manner. The models are trained on a subset of the original RobotCar trajectory at one season, but evaluated using the corresponding subset of frames at a different season.

As done previously, we train the dense feature variants of PoseNet by replacing the input image to the network with the dense feature representation, modifying the first layer accordingly. In this case, we evaluate variants trained on D2-Net, *SAND* and *Déjà-Vu*.

We show qualitative results for the estimated trajectory in Figure 4.9. Note that each prediction is performed independently, without temporal smoothing or additional cues. As expected, the variant using *Déjà-Vu* features follows the ground truth more closely and with less outliers. Once again, it is worth remembering that models were trained and evaluated using data from one season and evaluated on data from a different season. Quantitative results can be found in Table 4.5. As reflected by the previous visualization, *Déjà-Vu* clearly outperforms the baselines,

Table 4.5: Cross-Seasonal Visual Localization. We quantitatively evaluate the results from Figure 4.9 based on the **P**osition (meters) and **R**otation (radians) error. The feature based models were obtained by replacing the input image to PoseNet with the learnt dense feature representation. *Déjà-Vu* almost halves the error compared to the original PoseNet baseline.

	P (m) ↓	**R** (rad) ↓
PoseNet [80]	10.34	0.0170
D2-Net [45]	11.18	**0.0029**
SAND [172]	7.33	0.0045
Déjà-Vu - $\lambda_{cx} = 0$	**5.57**	0.0050
Déjà-Vu - $\lambda_{cx} = 1$	7.20	0.0036

almost halving the translation error. In this case, since we specifically focus on the cross-seasonal performance of the system, it is natural that the fully cross-seasonal model $\lambda_{cx} = 0$ performs best. However, the model using seasonal triplets still generally outperforms the remaining baselines.

4.4 Conclusions

This chapter has introduced *Déjà-Vu* features, a solution that overcomes the reliance of *SAND* on accurate pixel-wise labelling and the resulting issues with data bias. This has allowed us to develop cross-seasonally robust descriptors without needing cross-seasonal labels. *Déjà-Vu* uses only weak image-level supervision, which can be easily obtained from cheap, inaccurate sensors such as GPS. This removes the need for accurate pixel-wise correspondences, while still showing the networks real-world changes caused by different seasons and weathers. As such, this work has addressed objectives 1 & 2 of this thesis.

Our evaluations suggest that existing baseline techniques have learnt to be equivariant to seasonal transformations, not invariant. Two environments that can be identified as being similar under one season will likely appear similar to each other in a different season. However, the same environment under two different seasons is not recognised as begin similar. *Déjà-Vu* was shown to clearly overcome these limitations, outperforming these baselines and providing over 90%

accuracy. We further showed how incorporating the proposed contextual matching can lead to improved location retrieval robustness, for both hand-crafted and learning based features. We hope that the introduced technique is a step towards generalizing performance in challenging environments.

Unfortunately, *Déjà-Vu*'s main strength also provides its biggest limitation. By removing spatial constraints from our loss function we are able to successfully incorporate non-aligned data from multiple different seasons. However, this means that the correspondences estimated when computing the similarity are not required to be correct. Whilst in practice we have shown that *Déjà-Vu* is still capable of performing sparse feature matching, the next chapter will show how re-incorporating these spatial and geometric constraints leads to a further improvement.

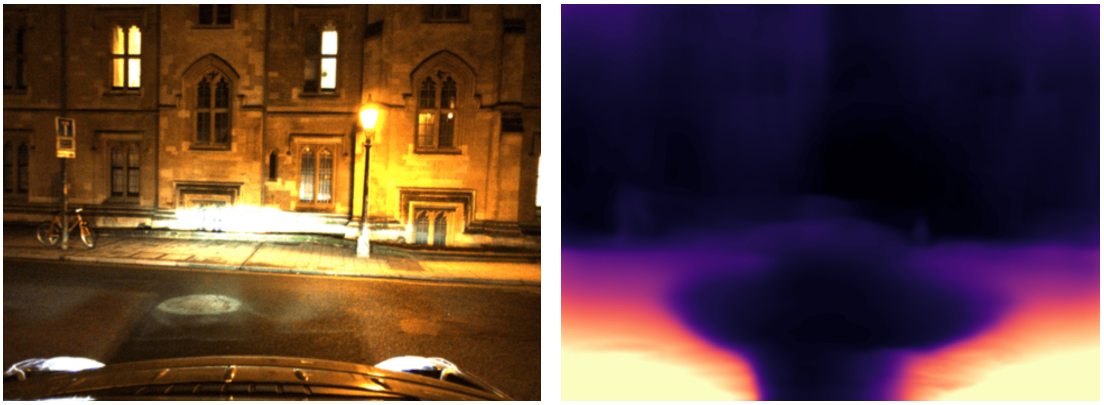
Chapter 5

Simultaneous Depth & Feature Learning in Challenging Conditions

In Chapter 3, *SAND* provided a framework for learning dense descriptors in a supervised manner. Given pixel-wise correspondences, we could establish relationships of (dis)similarity between feature descriptors. This relied on the availability of ground truth data, which is increasingly difficult to obtain as viewpoint and lighting conditions change. *Déjà-Vu* instead targeted cross-seasonal descriptors using non-aligned images from multiple seasons. Whilst this approach is quite flexible—requiring only GPS location—it comes at the expense of not enforcing spatial constraints.

This chapter provides the natural extension to both methods. We adapt the loose cross-seasonal constraints from *Déjà-Vu* and introduce strict spatially-valid pixel-wise constraints within seasons, as in *SAND*. This makes is possible to remove the need for LiDAR ground truth and instead introduce a self-supervised scheme based on monocular depth estimation. This allows us to include within-season matching constraints even for challenging scenes.

Recently there has been an increased interest in monocular depth estimation. We make the observation that depth estimation and feature description are inherently complimentary tasks. The process of estimating dense depth and VO between frames can be used to reconstruct the underlying 3-D geometry of the scene, allowing us to enforce photometric consistency when synthesising the new views required for training monocular depth. This also establishes



(a) **(b)**

Figure 5.1: Night-time Depth Estimation Challenges. **(a)** Challenging lighting conditions during night-time driving. **(b)** A catastrophic failure during depth map estimation for a current SOTA monocular depth estimation framework, after being trained specifically for this scenario.

pseudo-ground truth correspondences to drive feature descriptor learning. What's more, this is done in an entirely self-supervised way, requiring only a sequence of images and the camera's intrinsic parameters.

Unfortunately, as shown in Figure 5.1, current approaches to monocular depth estimation tend to fail in challenging environments. This is largely due to the assumptions of photometric consistency, which break down in low-light environments. Complex lighting from multiple sources, as well as reflections, lead to changes in appearance that are not taken into account in existing frameworks. However, feature descriptors are naturally trained to overcome changes in appearance from viewpoint and lighting changes. As such, they should be able to provide a more robust supervision signal for monocular depth estimation in adverse conditions.

We refer to our approach as *Depth & Feature Network (DeFeat-Net)* [174]. As discussed, the objective is to simultaneously learn monocular depth, dense feature description and VO. This addresses thesis objectives 1 & 4, which relate to the learning of dense features and simultaneous optimization of multiple tasks. Similar to the previous chapter, the only supervision signals required are the camera intrinsics and a rough measurement of which images are adjacent to each other. Neither LiDAR measurements nor accurate ground truth camera poses are needed. As such, this targets objective 2, reducing the amount of labels required to train the whole system.

5.1 Literature Review

Chapter 2 provided an overview of existing approaches to feature detection and description. This section provides additional background related to recent methods for depth estimation, including (self-)supervised approaches. Within these categories it is also worth making a distinction between those that rely on multiple images (*stereo*) *vs*. a single image (*monocular*).

5.1.1 Supervised

Stereo. Traditionally, depth estimation techniques simplified the process of finding geometric correspondences by stereo rectifying pairs of images. This reduces the problem to a search along a single row in the target image, known as disparity estimation. Initial approaches used a sliding SSD over a small window, incorporating smoothness and energy minimization constraints. Ladickỳ [87] and Žbontar [208] showed how learning the matching function drastically improved performance. Following approaches incorporated deep learning techniques to regress this disparity. For instance, DispNet [113] applied an FCN architecture [100] to directly predict the disparity between two images. Pang *et al*. [135] improved the resolution of DispNet by introducing additional refinement stages. Meanwhile, Kendall *et al*. introduced GC-Net [81], making use of a matching cost-volume in a 3-D CNN.

Monocular. Later approaches attempted to estimate the depth of a scene from a single image. This is a fundamentally ill-posed problem, since without additional cues it is impossible to differentiate between the size of an object and its depth. Saxena *et al*. [158] demonstrated that it was possible to approximate the scene's geometry by dividing it into superpixels and estimating their position and orientation independently. Liu *et al*. [98] expanded this by learning the models using a CNN, while Ladickỳ *et al*. [86] incorporated additional semantic cues.

Eigen *et al*. [48, 47] introduced the first framework capable of estimating monocular depth using end-to-end deep learning via a scale invariant loss. Laina *et al*. [89] improved the performance by using deeper networks and the more robust berHu loss [134, 217]. On the other hand, Cao *et al*. [19] treated depth estimation as a classification problem, where the depth was first quantized into multiple bins. Meanwhile, DeMon [187] targeted SfM and incorporated VO estimation. The *DeFeat-Net* approach proposed in this chapter takes this one step further, additionally learning a dense feature representation for correspondence estimation.

5.1.2 Self-supervised

Stereo. The ground truth required to train the previously presented methods typically relies on LiDAR data, making it costly and challenging to obtain. To circumvent this, several approaches have been proposed that instead use photometric constraints to learn depth estimation. For example, DeepStereo [54] synthesised novel views using pixel RGB values from arbitrary nearby views. Deep3D [198] restricted this to stereo rectified pairs and introduced a novel image reconstruction loss. The performance of these methods was greatly improved by Garg *et al*. [58] and Godard *et al*. [61], who introduced an additional autoencoder and left-right consistency, respectively. UnDeepVO [94] additionally learnt VO by adapting [216] and enforcing consistency between the stereo streams. A series of approaches have since made use of GANs [4, 137], forcing the reconstructed view to appear more realistic. Most notably, Sharma *et al*. [164] used GANs to perform day-to-night conversions to improve performance at night-time. However, this still does not generalize well and is highly sensitive to both adversarial training and the quality of the generated images.

Monocular. Finally, in order to learn monocular depth from a single stream of images without stereo information, it is necessary to incorporate motion information. If the whole framework is to be self-supervised, the motion must be learnt alongside the depth estimation. Zhou *et al*. [216] introduced this concept, warping the previous and next frames in a sequence to reconstruct the target view. This approach was extended by Zhan *et al*. [209], who incorporated a feature based warp loss. Meanwhile, Babu *et al*. [108] introduced an unsupervised version of DeMon [187].

More recently, research has focused on overcoming the "static world" assumption of the photometric loss, where dynamic objects result in unaccounted for occlusions. This has been achieved though additional temporal [109] or semantic [27] constraints, as well as edges & normals [203] or cycle consistency [136, 196]. The popular Monodepth2 [62] provided a simple and effective way of handling occlusions via the minimum reprojection loss. An additional automasking procedure is introduced to remove stationary pixels in the target frames, improving robustness to dynamic objects. Unfortunately, due to the previously discussed limitations of the photometric loss, performance is still decreased in challenging seasonal conditions.

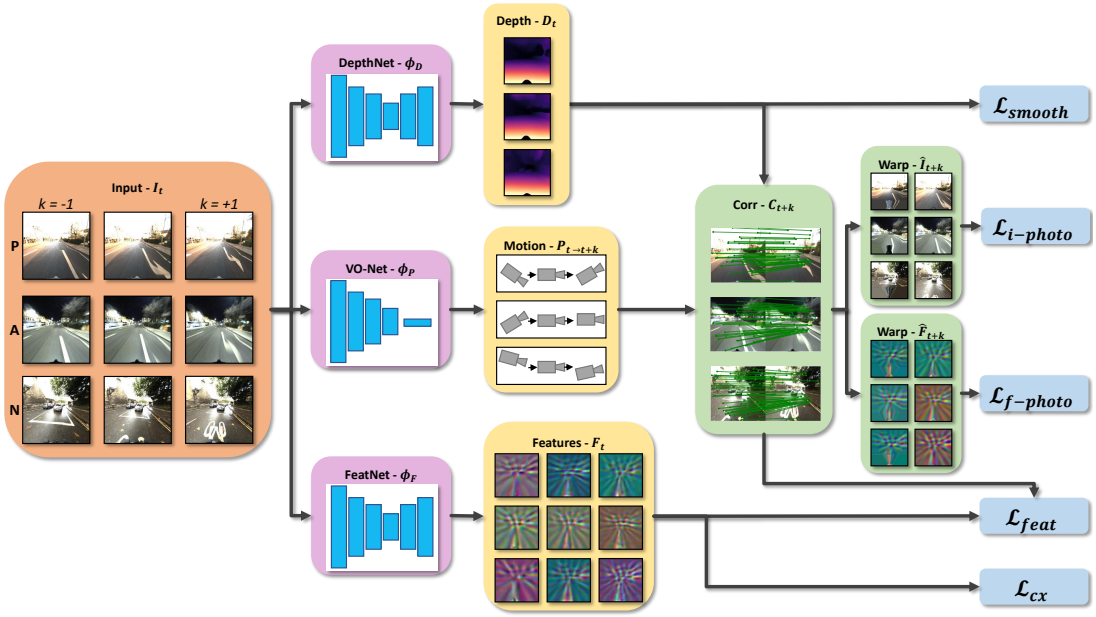


Figure 5.2: *DeFeat-Net* Overview. We simultaneously learn monocular depth estimation, dense feature description and VO. The correspondences obtained through the predicted depth and motion can be used in photometric warp losses and as feature supervision. Introducing dense feature learning improves the robustness of the system to complex seasonal conditions.

5.2 Methodology

The objective of *DeFeat-Net* is to jointly learn dense features, monocular depth and VO. By introducing the task of learning dense feature descriptors, which are trained to be invariant to lighting conditions, we can increase the robustness of the framework to adverse weather conditions. What's more, the geometric constraints linking both tasks allow us to achieve this in a fully self-supervised manner. The system can also be extended to improve the cross-seasonal consistency by incorporating GPS labels, which are inexpensive and easy to obtain.

Figure 5.2 shows the overview for the proposed system. The base monocular depth algorithm contains a target frame $\mathbf{I}_t$ and multiple supporting frames $\mathbf{I}_{t+k}$. These are typically the previous and next frames, *i.e.* $k \in \{-1, 1\}$. The estimated depth and motion allows us to obtain pixel-wise geometric correspondences, which support both the photometric and feature losses. To incorporate the contextual loss, we follow the procedure introduced in the previous chapter. The positive sample consists of consecutive frames at the same location and a different season, while negative samples are from a different location and any season.

5.2.1 Networks

DepthNet. Given an input image $\mathbf{I}_t$, the disparity is obtained through

$$\hat{\mathbf{D}}_t = \Phi_{\hat{\mathbf{D}}} \left(\mathbf{I}_t \right), \tag{5.1}$$

where $\Phi_{\hat{\mathbf{D}}}$ is a ResNet-based encoder-decoder network. The decoder contains four blocks, each upsampling the features by a factor of two and connected to the correspondingly size encoder layers via a skip connection. Each stage additionally produces an initial disparity estimate, which is used as additional supervision during training. The disparity produced by the network is constrained to the range $[0, 1]$. This is converted into a scaled depth via

$$\mathbf{D}_t = \frac{1}{a\hat{\mathbf{D}}_t + b}, \tag{5.2}$$

where a & b are constants chosen to scale the final depth map to the range $[0.1, 100]$.

VO-Net. The second network present in most monocular depth estimation frameworks is the VO regression network $\Phi_{\mathbf{P}}$. Similar to the the depth network, the encoder is based on residual convolutional blocks. However, since we are predicting relative motion, it is necessary to combine information from two frames. This is done by concatenating the images channel-wise and modifying the first convolution layer accordingly. Furthermore, the disparity network produces a dense output of the same shape as the input. Meanwhile, VO is a global task where a single 6-DoF pose is regressed for the whole image. Formally, we define this process as

$$\mathbf{P}_{t \to t+k} = \Phi_{\mathbf{P}} \left(\mathbf{I}_t \oplus \mathbf{I}_{t+k} \right), \tag{5.3}$$

where $\oplus$ is channel-wise concatenation and $\mathbf{P}_{t \to t+k}$ is the transform between the cameras at time t and $t + k$. As in previous work [216, 62], this transform is represented as a translation and axis-angle rotation, scaled by a factor 0.01.

FeatNet. The final network in the system, and the one related to our main contribution, is the dense feature descriptor network $\Phi_{\mathbf{F}}$. We use the same network as previously introduced in Chapter 3. As a reminder, this consists of an encoder-decoder structure with skip connections between corresponding layers and an optional SPP as the final encoder stage. The output of the network is given by

$$\mathbf{F}_t = \Phi_{\mathbf{F}} \left(\mathbf{I}_t \right), \tag{5.4}$$

where $\mathbf{F}_t$ is of shape ${}^{h}/_{f} \times {}^{w}/_{f} \times n_{dim}$. Once again, the objective is to produce a dense feature map, while allowing for a downsampling factor f to improve the global consistency.

5.2.2 Correspondence Module

Following the equations presented in Section 3.1.3, we can use the reprojection function (3.9) along with the estimated depth and motion to establish correspondences between frames. We therefore define the dense set of correspondences as

$$C_{t+k}\left(\mathbf{p}_t\right) = \Pi\left(\mathbf{p}_t | \mathbf{K}_t, \mathbf{P}_{t\rightarrow t+k}, \mathbb{I}_4, \mathbf{K}_t, \mathbf{D}_t\right) \forall\ \mathbf{p}_t \in \mathbf{I}_t, \tag{5.5}$$

where Π once again is the reprojection function, conditioned on the camera intrinsics $\mathbf{K}_t$, the depth at the point $\mathbf{D}_t\left\langle\mathbf{p}_t\right\rangle$ and the relative position change between cameras $\mathbf{P}_{t\rightarrow t+k}$. Note that since we use the relative position, the first image is therefore located at the origin and its pose is represented by the 4×4 Identity matrix $\mathbb{I}_4$.

These correspondences are commonly used as the sampling locations when synthesizing new views in the photometric warp loss. We additionally use them as positive matches in the feature learning loss. This allows us to obtain pixel-wise geometric constraints in an entirely self-supervised manner.

5.2.3 Losses

Relational feature loss. As in Chapter 3, any relational loss with a notion of positive and negative samples can be used to train the feature descriptor network. We make use of the previously introduced PixCon and NT-Xent losses. The relational feature loss $\mathcal{L}_{feat}$ therefore corresponds to either $\mathcal{L}_{con}$ from (3.4) or $\mathcal{L}_{xent}$ from (3.6). As a reminder, these losses require a correspondence map $\mathbf{Y}$ indicating the ground truth relationship between each pair of pixels in the dense feature maps: matching, non-matching or ignored. In *SAND*, this correspondence map was obtained by reprojecting LiDAR or SfM data onto pairs of images. In the case of *DeFeat-Net*, we instead used the estimated correspondences C_{t+k} as pseudo-ground truth, resulting in

$$\mathbf{Y}\left(\mathbf{p}_t, \mathbf{p}_{t+k}\right) = [\![C_{t+k}\left(\mathbf{p}_{t+k}\right) = \mathbf{p}_t]\!]. \tag{5.6}$$

This loss serves to drive the feature learning, enabling robustness to changes caused by different weathers and seasons.

Contextual feature loss. The correspondences C_{t+k} are only obtained between sequential images of the same season. This ensures that the resulting descriptors can be used to match within each of the training seasons. However, as we saw in the previous chapter, this does not guarantee that the descriptors are consistent across multiple seasons. We therefore incorporate the contextual triplet loss ℓ_{cx} from (4.7). Each training batch is constructed following the procedure from *Déjà-Vu*, where we sample a positive sequence from the same location and a different season, as well as a negative from any season and any location.

Déjà-Vu repeated this process to create a cross-seasonal triplet for the current frame $\mathbb{T}_X^1$ and for the following frames $\mathbb{T}_X^2$. In *DeFeat-Net* we extend this idea to create a triplet $\mathbb{T}_X^k$ for each offset k, including the target image $k = 0$. Note that, due to the inclusion of the within-season pixel-wise supervision $\mathcal{L}_{feat}$, we do not require seasonal triplets. As such, we redefine the contextual loss as

$$\mathcal{L}_{cx}\left(\mathbb{T}_X\right) = \sum_k \ell_{cx}\left(\mathbb{T}_X^k\right). \tag{5.7}$$

Photometric & feature warp. To train the depth and motion networks we additionally use the estimated correspondences in a differentiable bilinear sampler [76]. This allows us to synthesize the target frame and target feature map based on either the support frames or support feature maps. This is defined as

$$\begin{cases} \hat{\mathbf{I}}_{t+k} = \mathbf{I}_t\left\langle C_{t+k}\right\rangle, \\ \hat{\mathbf{F}}_{t+k} = \mathbf{F}_t\left\langle C_{t+k}\right\rangle, \end{cases} \tag{5.8}$$

where $\langle\rangle$ once again represents the bilinear sampling operation. The final reconstruction loss is a weighted combination of the SSIM and L_1 losses as

$$\mathcal{L}_{photo}\left(\mathbf{I}_1, \mathbf{I}_2\right) = \lambda_{ssim}\frac{1\text{-}\mathit{SSIM}\left(\mathbf{I}_1, \mathbf{I}_2\right)}{2} + \left(1\text{-}\lambda_{ssim}\right)\left|\mathbf{I}_1\text{-}\mathbf{I}_2\right|, \tag{5.9}$$

where λ_{ssim} controls the balance between both losses. Note that $\mathcal{L}_{photo}$ produces a dense loss with an error for each pixel in the image. Importantly, this loss can be applied to both the

synthesised image and the synthesised feature map

$$\begin{cases} \mathcal{L}_{i-photo} = \sum\limits_{k} \sum\limits_{\mathbf{p}} \mathcal{L}_{photo} \left(\mathbf{I}_t, \hat{\mathbf{I}}_{t+k} \right) \\ \mathcal{L}_{f-photo} = \sum\limits_{k} \sum\limits_{\mathbf{p}} \mathcal{L}_{photo} \left(\mathbf{F}_t, \hat{\mathbf{F}}_{t+k} \right) \end{cases}, \tag{5.10}$$

where the loss is averaged over all pixels in the image $\mathbf{p}$ and each of the support frame offsets k.

The photometric loss $\mathcal{L}_{i-photo}$ primarily supports the depth and motion learning during the initial stages of training, while the feature space is being learnt. Meanwhile, $\mathcal{L}_{f-photo}$ provides a more reliable supervision in challenging seasonal and weather conditions, where the assumptions of photometric consistency break down.

Smoothness. Finally, we apply a common regularizing constraint [61] to enforce smoothness in the predicted depth proportional to the strength of the edges in the image. This is defined as

$$\mathcal{L}_{smooth} = \lambda_{smooth} \sum_{\mathbf{p}} \left| \partial \mathbf{D}_t \left(\mathbf{p} \right) \right| \ \exp \left(\text{-} \left\| \partial \mathbf{I}_t \left(\mathbf{p} \right) \right\| \right), \tag{5.11}$$

where $\partial \mathbf{I}_t$ are the spatial gradients in the image and λ_{smooth} controls the overall contribution of the loss. Intuitively, this only allows for sudden changes in depth in highly textured regions of the image, where there is more likely to be object boundaries.

5.2.4 Masking & Filtering

Photometric consistency is one of the main assumptions made by previous monocular depth frameworks. Our feature matching loss seeks to loosen this assumption. The second common assumption is that of a static world, where no objects move across different frames. However, this is frequently violated due to dynamic moving objects such as vehicles and pedestrians. This causes incorrect correspondences which can lead to oversmoothed boundaries and inaccurate predictions. It is therefore natural that recent improvements in monocular depth prediction have arisen from explicitly handling occlusion filtering and stationary pixel masking [62].

Minimum reprojection. Throughout the motion in a sequence, different objects will become (dis)occluded. To provide increased robustness, multiple supporting frames can be used, each with different occlusions. Instead of taking the mean error over these offsets, as in (5.10), it

can be beneficial in this situation to assume that the point with the lowest loss is the correct correspondence and to ignore the others. We therefore redefine the photometric losses as

$$\begin{cases} \mathcal{L}_{i-photo} = \sum\limits_{\mathbf{p}} \min\limits_{k} \ \mathcal{L}_{photo}\left(\mathbf{I}_t, \hat{\mathbf{I}}_{t+k}\right) \\ \mathcal{L}_{f-photo} = \sum\limits_{\mathbf{p}} \min\limits_{k} \ \mathcal{L}_{photo}\left(\mathbf{F}_t, \hat{\mathbf{F}}_{t+k}\right) \end{cases} . \tag{5.12}$$

Note that these minimum reprojections are also used to refine the correspondence map $\mathbf{Y}$ generated in (5.6), used to compute the relational feature learning loss $\mathcal{L}_{feat}$.

Automasking. Objects in the image that are further away will move a smaller distance than those that are closer. Therefore, an object that does not move at all must be at an infinite distance. However, this effect can also be caused by static scenes where the camera has not moved, or by objects travelling at similar speeds to the reference camera *i.e.* other cars. This results in incorrect predictions of infinite depth. Previous works [216] attempted to simultaneously predict which points should be masked out from the loss. A simpler and more effective alternative is

$$\mathbb{M} = [\![\min_{k} \ \mathcal{L}_{photo}\left(\mathbf{I}_t, \hat{\mathbf{I}}_{t+k}\right) < \mathcal{L}_{photo}\left(\mathbf{I}_t, \mathbf{I}_{t+k}\right)]\!], \tag{5.13}$$

where $\mathbb{M}$ is the resulting mask indicating if a correspondence is valid. Note that the second loss term in the Iverson bracket uses the original non-warped support image $\mathbf{I}_{t+k}$. Intuitively, this masks pixels where the original unwarped support frame leads to a smaller loss than the warped frame. This would be expected if a pixel does not move across frames. Since low-light data suffers from a reduced overall brightness and contrast, the automasking process may result in the filtering of correct correspondences. To account for this, the automasking procedure can also be applied using the feature-based photometric loss $\mathcal{L}_{f-photo}$. Once again, the improved robustness to lighting conditions provided by the learned feature descriptors can provide a more informative supervision loss.

5.3 Results

Implementation details. Each network in *DeFeat-Net* uses a ResNet-18 backbone, pretrained on ImageNet [39]. Due to the high memory requirements, the feature network $\Phi_{\mathbf{F}}$ uses a feature dimensionality $n_{dim} = 10$ and a downsampling factor $f = 8$. As mentioned previously, we use

the previous and next frames as support frames $k \in \{-1, 1\}$. Regarding the losses, the final relational feature loss $\mathcal{L}_{feat}$ uses NT-Xent and randomly samples 5000 pseudo-ground truth correspondences from the depth & motion reprojection. We use an SSIM weight $\lambda_{ssim} = 0.85$ and a smoothness factor $\lambda_{smooth} = 0.001$. The feature-based photometric loss $\mathcal{L}_{f-photo}$ computes a separate automask based on the learnt dense features. *DeFeat-Net* is trained using an Adam optimizer with a base LR of 10^{-4}. Models using Kitti train for 30 epochs, while those on RobotCar Seasons use 100 epochs.

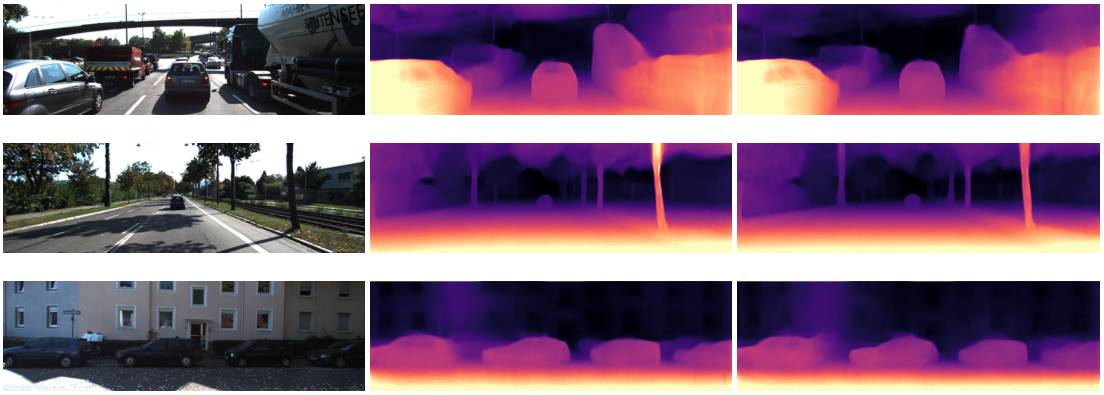
Datasets. We use the Kitti dataset [60] to train our baseline daytime models. As is common, we follow the Eigen-Zhou evaluation protocol, which provides almost 44k images for training and 4.5k for validation. As with *Déjà-Vu*, the cross-seasonal models are trained using RobotCar Seasons [157], which contains a wide variety of seasonal conditions.

Unfortunately, RobotCar Seasons does not contain ground truth depth data. Since *DeFeat-Net* only requires the GPS-level supervision from *Déjà-Vu*, this does not affect out training procedure. However, it means that a quantitative evaluation on this dataset is impossible. We therefore make use of the original RobotCar dataset [107] with sequences "2015-08-27-10-06-57" and "2014-12-16-18-44-24" as daytime and night-time, respectively. We ensure that this data does not overlap the sequences used for training in RobotCar Seasons. This results in approximately 3000 images with corresponding ground truth LiDAR data.

5.3.1 Depth Evaluation - Canonical Season (Kitti)

We first evaluate *DeFeat-Net* in a traditional daytime driving scenario, where the photometric consistency assumption typically holds. As discussed, we use the Eigen-Zhou evaluation protocol on Kitti, which includes standard error metrics such as absolute relative depth error, relative square error and Root Mean Squared Error (RMSE). We additionally show the inlier ratio measures, reporting the relative depth errors within 25%, 56% and 95% of the ground truth. In this case, since Kitti provides only daytime data, *DeFeat-Net* cannot be trained with the contextual loss $\mathcal{L}_{cx}$.

These results can be found in Table 5.1 and visualized in Figure 5.3. In this case, our approach provides comparable performance to the current SOTA [62], while outperforming all previous approaches. Note that these results were obtained by retraining the code provided by the authors.



(a) Monodepth2 **(b)** ***DeFeat-Net***

Figure 5.3: Kitti Depth Estimation. We visualize the network predictions on the Kitti daytime data. In this case, both Monodepth2 and *DeFeat-Net* provide similar results and robustness.

Table 5.1: Monocular Depth Evaluation on Kitti. We evaluate on ideal daytime driving conditions following the Eigen-Zhou protocol. In these conditions the photometric assumptions typically hold. As such, *DeFeat-Net* simply increases the complexity by introducing a new task.

	Abs-Rel ↓	Sq-Rel ↓	RMSE ↓	RMSE-log ↓	A1 ↑	A2 ↑	A3 ↑
SfMLearner [216]	0.183	1.595	6.709	0.270	0.734	0.902	0.959
LEGO [203]	0.162	1.352	6.276	0.252	-	-	-
Ranjan [142]	0.148	1.149	5.464	0.226	0.815	0.935	0.973
EPC++ [104]	0.141	1.029	5.350	0.216	0.816	0.941	0.976
Struct2depth [22]	0.141	1.026	5.291	0.215	0.816	0.945	0.979
Monodepth2 [62]	**0.121**	**0.851**	**4.700**	**0.191**	**0.867**	**0.961**	**0.983**
DeFeat-Net	0.122	0.869	4.807	**0.191**	0.861	0.960	**0.983**

It is not surprising that *DeFeat-Net* does not provide significantly different results under these experimental conditions. In clear daytime driving, the assumptions of photometric consistency are typically valid. As such, *DeFeat-Net* is simply adding an extra task to be learnt (dense feature descriptors), whilst not providing additional training support. However, this changes when training and evaluating in challenging environmental conditions.

Table 5.2: Monocular Depth Evaluation on RobotCar. We evaluate on daytime and night-time RobotCar sequences, using models trained on all seasons in RobotCar Seasons. *DeFeat-Net* provides a clear advantage over its purely photometric counterpart [62]. This is particularly noticeable at night-time, where the relative error metrics are reduced by over 30%. Meanwhile, A1 robustness is improved by 6%.

	Test domain	Abs-Rel ↓	Sq-Rel ↓	RMSE ↓	RMSE-log ↓	A1 ↑	A2 ↑	A3 ↑
Monodepth2 [62]	Day	0.272	3.806	9.629	0.329	0.592	0.824	0.925
DeFeat-Net	Day	**0.259**	**3.571**	**9.443**	**0.319**	**0.608**	**0.837**	**0.933**
Monodepth2 [62]	Night	0.336	5.657	9.546	0.370	0.603	0.822	0.911
DeFeat-Net	Night	**0.230**	**3.021**	**8.817**	**0.292**	**0.665**	**0.864**	**0.943**

5.3.2 Depth Evaluation - All Seasons (RobotCar)

The performance on the challenging RobotCar dataset shows the true benefit of the joint optimization performed by *DeFeat-Net*. RobotCar (Seasons) provide a wide variety of weather conditions including night, night & rain, snow, dusk and dawn. In most of these cases, these environments contain multiple light sources, complex reflections, increased blur and brightness changes that invalidate the photometric assumption.

As seen in Table 5.2, *DeFeat-Net* provides a clear advantage over Monodepth2 [62] in these conditions. Whilst improvements in the daytime sequence are modest, night-time results show an overall improvement $> 30\%$ in the relative error measures. *DeFeat-Net* is particularly robust with regards to the number of outliers, showing consistent A1, A2 & A3 metrics across day and night data. This suggests that when facing regions of uncertain depth, such as blurring or under-exposure, the proposed approach fails gracefully rather than producing catastrophically incorrect predictions. This is also reflected in the qualitative visualizations in Figure 5.4, where Monodepth2 shows many holes of infinite depth instead of predicting the road surface.

In the case of *DeFeat-Net*, incorporating the dense feature learning task increases the overall complexity of the system. However, the base photometric loss is not robust enough to the complex lighting conditions, leading to a weak supervision signal. Meanwhile, the feature supervision losses result in feature representations invariant to these conditions. This allows the feature warp loss to provide a more stable supervision that leads to improved depth predictions.

(a) Monodepth2 **(b)** ***DeFeat-Net***

Figure 5.4: RobotCar Depth Estimation. We visualize the network predictions on the RobotCar data. *DeFeat-Net* provides clear improvements under the challenging night-time conditions. The basic photometric loss is not capable of providing effective supervision in these conditions, leading to more holes of infinite depth.

Table 5.3: *DeFeat-Net* Ablation Study. We test the various contributions of *DeFeat-Net*. Each of these results in improvements over the featureless baseline [62]. Most notably, we find that better features directly correlate better depth estimation.

$\Phi_{\mathbf{F}}$	$\mathcal{L}_{feat}$	F-Mask	CX	Abs-Rel ↓	Sq-Rel ↓	RMSE ↓	RMSE-log ↓	A1 ↑	A2 ↑	A3 ↑
				0.336	5.657	9.546	0.370	0.603	0.822	0.911
$f = 1$	$\mathcal{L}_{con}$			**0.295**	**5.459**	**9.266**	**0.340**	**0.632**	**0.848**	**0.930**
$f = 1$	$\mathcal{L}_{con}$			0.295	5.459	9.266	0.340	0.632	0.848	0.930
$f = 8$	$\mathcal{L}_{con}$			0.277	4.086	**9.020**	0.333	0.657	0.851	0.929
$f = 8$	$\mathcal{L}_{xent}$			**0.233**	**3.677**	9.051	**0.299**	**0.677**	**0.864**	**0.940**
$f = 1$	$\mathcal{L}_{con}$			0.295	5.459	9.266	0.340	0.632	0.848	**0.930**
$f = 1$	$\mathcal{L}_{con}$	✓		**0.280**	**4.556**	**9.176**	**0.334**	**0.655**	**0.850**	**0.930**
$f = 8$	$\mathcal{L}_{xent}$	✓		0.235	3.045	8.842	0.299	0.651	0.860	0.941
$f = 8$	$\mathcal{L}_{xent}$	✓	✓	**0.230**	**3.021**	**8.817**	**0.292**	**0.665**	**0.864**	**0.943**

5.3.3 Ablation Study

As in previous chapters, we perform an ablation study to test the effects of the various components that form *DeFeat-Net*. This ablation is performed on the RobotCar night-time data. The base models use ResNet-18 backbones pretrained on ImageNet, a feature network with scale $f = 1$ and the PixCon loss $\mathcal{L}_{con}$. These results can be found in Table 5.3.

Monocular depth *vs*. *DeFeat-Net*. We first compare performance with a baseline model based exclusively on the photometric image loss [62]. We then introduce the feature learning task proposed by *DeFeat-Net*, consisting of the feature network $\Phi_{\mathbf{F}}$, the relational feature loss $\mathcal{L}_{con}$ and the feature-based photometric loss $\mathcal{L}_{f-photo}$. As seen in the first two rows in Table 5.3, even in its most basic form, incorporating the feature learning task results in significant improvements. As previously discussed, this is due to the additional robustness provided by the feature-based photometric loss in challenging conditions.

Feature resolution & loss. As in previous chapters, we find that a lower resolution feature map results in improved consistency. Note that in this case we upsample the features to the original image resolution when computing the feature-based photometric loss $\mathcal{L}_{f-photo}$. Similar

to *SAND*, the NT-Xent loss provides better results than PixCon. This ablation highlights the importance of the learnt feature representation. A more discriminative feature representation directly correlates with improved depth estimation in challenging conditions.

Feature automasking. Following [62], we introduce automasking to filter points from static scenes and objects moving at similar speeds to the vehicle. This procedure suffers from similar limitations to the image-based photometric loss when applied to night-time data. Following our intuition for the proposed photometric feature loss, we compute a new automask using the features as opposed to the raw images. As seen in the results in rows 6 & 7 of Table 5.3, incorporating this automasking results in further improvements.

Contextual loss. Finally, we incorporate the contextual supervision used in *Déjà-Vu*, which only requires the use of rough alignment labels obtained via GPS. This is shown in the final rows of Table 5.3. Despite not directly interacting with the depth estimation, incorporating this loss still results in improved performance. We believe this is due to the fact that this loss improves invariance to adverse weather conditions (as previously demonstrated with *Déjà-Vu*), further increasing the robustness and stability of the photometric feature loss $\mathcal{L}_{f-photo}$.

5.4 Conclusions

This chapter presented the *DeFeat-Net* framework, with the objective of combining the advantages of both *SAND* and *Déjà-Vu*. We achieved this by simultaneously learning monocular depth estimation, dense features and VO. Most importantly, we were still capable of providing robustness to drastic appearance changes caused by different seasons and weathers, while requiring minimal labelling. This approach provides a bridge between supervision using sparse geometric constraints (within seasons) and global similarity labels (across seasons). This has therefore tackled objectives 1, 2 & 4 in this thesis.

This chapter demonstrated that dense features can be learnt simultaneously alongside other computer vision tasks, complementing them and boosting their performance. As shown in the results, *DeFeat-Net* provides a significant improvement over previous SOTA when applied to complex real-world scenes. However, one of the limitations of *DeFeat-Net* is its large memory requirements. For instance, each of the different tasks trains a completely separate network.

Paired with the memory requirements for the multiple losses, *i.e*. synthesising new views and sparse & dense distance cost matrices, the resulting system must use a limited batch size with downsampled images.

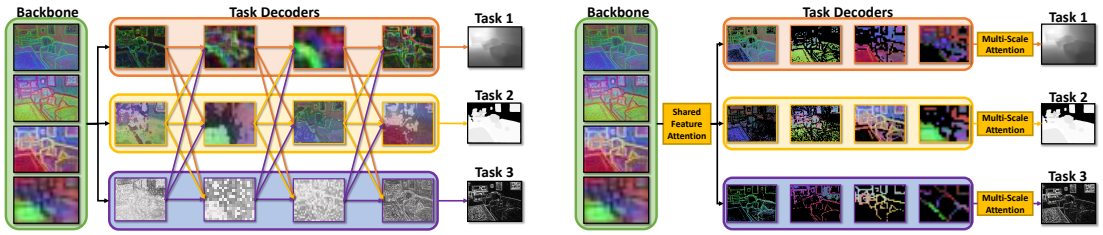
The main reason for separating the depth, pose and feature networks is that naïve multi-tasking (where the encoder is shared and each task has a separate decoder) can actually lead to worse performance. This is caused by the sharing of irrelevant distractor features. Deciding what features should be shared is a complex research problem, since this relationship changes depending on the combination of tasks, scales and each individual image. The following chapter will discuss these challenges in greater detail, providing a simpler solution that can still maintain high performance.

Chapter 6

Universal Feature Learning via Attentional Multitasking

Chapter 3 presented a two-stage pipeline, where generic features were first trained using ground truth correspondences and then applied to various downstream tasks. In most cases, these features required finetuning to reach their maximum performance. Unfortunately, this process destroyed the generality of those features and resulted in them *forgetting* [114, 143] previous tasks. *SAND* also required hand-picking an appropriate spatial negative mining scale based on our intuition of the properties of the target task. On the other hand, *DeFeat-Net* simultaneously trained monocular depth, VO and dense features in a naïve multitasking system. This provided a boost in performance due to the inherent relationship between the tasks, but did not explicitly share information at the feature level. This made the system less efficient and harder to train, since none of the computation was shared.

This chapter focuses on learning generic feature representations which are simultaneously applicable to a wide variety of computer vision tasks. We therefore approach this from the perspective of MTL. Initial naïve approaches to MTL focused on sharing computation at the backbone level, from which multiple independent task heads emerged [83, 99, 124, 57]. This makes the network highly efficient. Unfortunately, these approaches do not perform well due to the fact that different regions of the image have differing levels of importance for each task. To improve performance, recent approaches [199, 213, 191] attempt to model the relationship between tasks by incorporating spatial attention connections between each possible pair of tasks.



(a) Traditional **(b)** Proposed

Figure 6.1: Universal Feature Learning. Current approaches to MTL incorporate connections between every pair of tasks, resulting in a quadratic parameter complexity w.r.t. the number of tasks. We reduce this to a linear scaling by keeping isolated task heads. Furthermore, incorporating two spatial attention mechanisms (SFA & MSA) matches the performance of these complex methods, while learning features that transfer beyond the original training set.

This comes at the cost of a quadratic parameter complexity, limiting the applicability of these approaches to a larger number of tasks.

We make the observation that all of these techniques result in a system that is highly optimized only for the set of training tasks. In other words, the set of training and evaluating tasks are the same. We argue that these systems are therefore not generic and reusable, especially given that adding a new task would require incorporating new connections to all previous tasks and retraining the whole network. In contrast, we want the features in the network to generalize beyond the original training tasks, *i.e*. the training and evaluating tasks are different. We refer to this process as Universal Feature Learning (UFL).

We propose to revisit architectures with completely independent task heads, where the backbone is the only shared component. This creates an information bottleneck that forces the backbone to learn generic features suitable for all target tasks, as illustrated in Figure 6.1. To mitigate the effects of negative transfer between unrelated tasks we introduce a spatial attention mechanism between the backbone and each task head at each scale. The task head features are refined and used to make initial predictions at each scale for additional supervision during training. Finally, the task features at all scales are combined using the novel Multi-Scale Attention (MSA) task head, accounting for the fact that different scales have different roles in the final prediction. The whole procedure results in a highly efficient feature extraction process that is only linear w.r.t. the number of tasks. It also becomes trivial to add new task heads after the initial training stage. These can be trained independently while making use of the learnt shared features.

6.1 Literature Review

The objective of MTL [21, 149, 190] is to train a network that can simultaneously perform multiple tasks. This leads to an improved efficiency and/or performance over multiple independent networks due to the fact that information between tasks can be shared. UberNet [83] proposed a multi-scale and multi-head architecture consisting of a shared backbone and additional feature sharing layers. Meanwhile, Cross-stitch networks [124] learned linear combinations of task features, introducing the concept of soft feature sharing. Further approaches [150, 57] extended this concept, replacing the linear combinations with subspace & skip connection sharing and dimensionality reduction, respectively. The main drawback of the soft feature sharing approaches is that they first require each task to be trained separately, from which the features are then finetuned. This also means that these approaches focus on improving performance, not on making the MTL system more efficient.

Unfortunately, the performance of these systems was not always improved, despite the incorporation of feature sharing layers. This is due to the fact that not all tasks are related, and not all information should be shared. The sharing of unrelated information, harming performance for both tasks, is known as *negative transfer* [83, 214]. Vandenhende *et al*. [189] used precomputed task affinity scores [46] to decide the structure of the network and what layers to share. Other approaches instead aim to learn these relationships, optimizing the network architecture directly. For instance, FAFS [103] begins with a fully shared model, which is then optimized to separate dissimilar tasks and regularized to maintain a low complexity. Meanwhile, BMTAS [16] and LTB [63] represented branching points in the shared backbone using the Gumbel softmax [77].

More recent approaches have instead opted for maintaining a static network architecture, where the flow of information between tasks is dynamic based on attention mechanisms. For instance, MTAN [99] created parallel task encoders by introducing spatial attention at each stage in the shared backbone. PAD-Net [199] introduced the concept of multi-task distillation, where each task head first made initial predictions. These predictions were then refined and combined via spatial attention between each possible pair of tasks. MTI-Net [191] extended this to a multi-scale approach, including addition feature propagation modules between scales. PAP-Net [213] replaced the spatial attention in the multi-task distillation with a learnt affinity and diffusion between each pair of tasks. TRL [212] instead proposed a recursive procedure for refining

sequential task predictions, which are combined using spatial attention. These approaches generally result in an improved performance. However, introducing connections between all possible pairs of tasks leads to a quadratic growth in complexity w.r.t. to the number of tasks. This is not a problem when only a small number of tasks are being trained, but causes severe scaling issues as the number of tasks increases.

6.2 Methodology

We refer to the approach proposed in this chapter as *Medusa*. It aims to learn generic features that can be applied to a wide range of computer vision tasks. This includes extending beyond the original set of training tasks. An overview of the proposed network can be found in Figure 6.2, illustrating the two main components of *Medusa*: a shared backbone and independent task heads. The shared backbone represents an information bottleneck that forces the network to learn a generic representation suitable for all training tasks. Each task head is connected to the backbone via the Shared Feature Attention (SFA), allowing it to retain only the relevant information for that task and mitigating negative transfer. The predictions for each task are further processed by the novel MSA heads, providing an effective way of combining information from multiple scales. Altogether, this results in a highly efficient architecture that maintains performance over multiple tasks.

6.2.1 Shared Feature Attention

The only component shared between the multiple tasks is the common backbone. Given an input image $\mathbf{I}$, this backbone produces a multi-scale feature representation $\mathbf{B}_s = \Phi_{\mathbf{B}}(\mathbf{I})$, where the scale s represents an increasing downsampling factor. Since these features are common to all downstream tasks heads, the learnt representation must be generic and reusable. However, as previously discussed, the sharing of information across unrelated tasks can be harmful to their performance. We therefore require a way of filtering the information in such a way that each task head keeps only relevant features. This is achieved by introducing a local spatial attention mechanism between the backbone features and each of the individual task heads. Following previous work [38, 99, 199, 191], we define the process of applying spatial attention SA to an

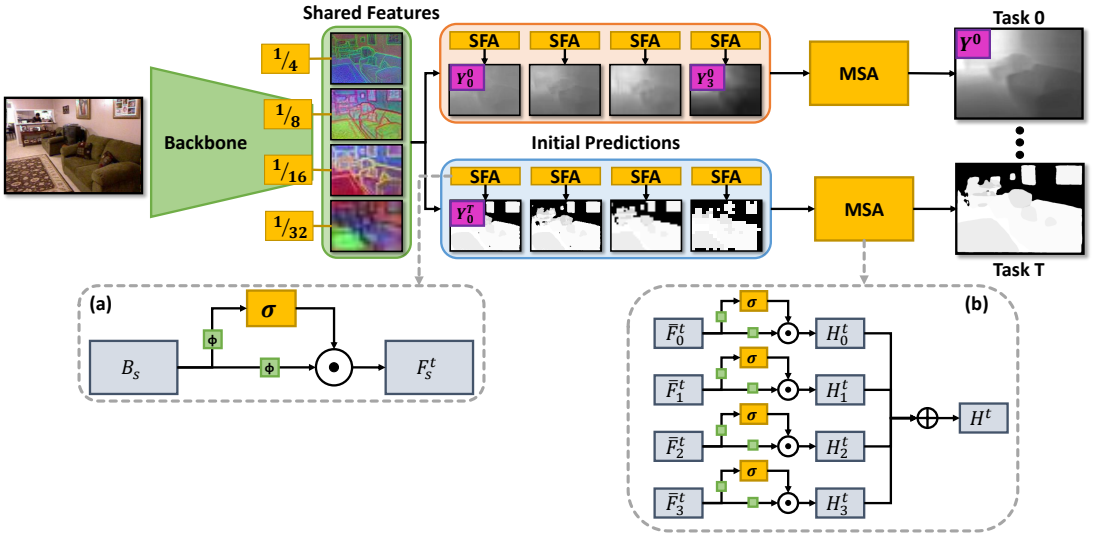


Figure 6.2: Proposed *Medusa* Architecture. Our architecture focuses on maintaining independent task heads. This results in a more efficient scaling to a larger number of tasks, while learning more generic and reusable features. **(a)** Shared Feature Attention filtering shared backbone features into task specific features at each backbone scale through per-channel spatial attention. **(b)** Novel Multi-Scale Attention head combining task features at different scales to generate the final predictions.

arbitrary feature map $\mathbf{F}$ as

$$SA\left(\mathbf{F}\right) = \sigma\left(\phi_1\left(\mathbf{F}\right)\right) \odot \phi_2\left(\mathbf{F}\right), \tag{6.1}$$

where σ is the sigmoid operation, $\odot$ the Hadamard product and ϕ a convolutional block including BatchNorm and ReLU.

In the case of *Medusa*, we learn a separate attention for each target task at each backbone scale, resulting in $N_{\mathbb{T}} \cdot N_s$ independent attention blocks. Therefore, $\mathbf{F}_s^{\mathbb{T}} = SA_s^{\mathbb{T}}\left(\mathbf{B}_s\right)$ represents the initial task features for a given scale s and task $\mathbb{T}$. To summarize, the per-channel spatial attention $\sigma\left(\phi_1\left(\mathbf{F}\right)\right)$ allows each task head to learn which of the generic features within the backbone are more suitable for its task, while discarding the rest. This reduces the likelihood of *negative transfer* in a highly cost-effective manner. Furthermore, the multi-scale nature of the proposed architecture results in a wide variety of information being available to the following stages in the task head when making the final predictions.

6.2.2 Multi-Scale Task Predictions

Previous approaches [124, 99, 111] used task heads that made a direct prediction for each task. However, recent methods [199, 213, 191] have instead shown the benefit of incorporating intermediate predictions as additional supervision during training. Following these approaches, *Medusa* makes intermediate predictions for each task at each of the scales produced by the backbone. Once again, these predictions are only used as intermediate supervision during training and can be discarded when evaluating the final model. To generate these predictions, the filtered shared features at each scale $\mathbf{F}_s^{\mathbb{T}}$ are refined into task specific features through

$$\bar{\mathbf{F}}_s^{\mathbb{T}} = \psi_1 \left(\psi_2 \left(\mathbf{F}_s^{\mathbb{T}} \right) \right), \tag{6.2}$$

where $\bar{\mathbf{F}}_s^{\mathbb{T}}$ are the resulting refined features and $\psi = \phi\left(\mathbf{F}\right) + \mathbf{F}$ is a convolutional residual block. The initial predictions for each scale and task are then given by $\mathbf{O}_s^{\mathbb{T}} = \phi_s^{\mathbb{T}} \left(\bar{\mathbf{F}}_s^{\mathbb{T}} \right)$, where $\phi_s^{\mathbb{T}}$ maps from the number of backbone channels at that scale C_s to the channels required by the task. Once again, these intermediate predictions are used only as additional supervision during the training phase. This helps to guide the intermediate task features towards the target task, allowing them to more effectively support the final prediction.

6.2.3 Multi-Scale Attention Task Heads

The final step is producing the output predictions for each task. Previous approaches [99, 111] operated sequentially, where the final predictions were generated exclusively based on the features of the last layer. More recent multi-scale approaches [191] instead upsample all intermediate predictions to the highest resolution, concatenate them together and process them in a final convolutional stage. We refer to this task head as HRHead in the results discussed in Section 6.3.

The main drawback of this task head is that it does not take into account the different roles that each scale provides. In practice, different scales have complementary benefits and failure modes [117]. For instance, through the wider context available, lower resolution predictions are capable of providing a cohesive overview of the whole scene. However, this results in a lack of high-frequency detail and oversmoothed predictions. Conversely, high resolution predictions capture the nuances in the scene, but can be inconsistent on a global scale.

We account for this by introducing the novel Multi-Scale Attention head. Given a set of refined task features at multiple resolutions, we let the task head decide what information from each scale should be used and how to combine it most effectively. Once again, this is achieved through the use of the spatial attention defined in (6.1). Formally, we define this process as

$$\mathbf{H}_s^{\mathbb{T}} = SA_s^{\mathbb{T}}\left(\bar{\mathbf{F}}_s^{\mathbb{T}}\right), \tag{6.3}$$

$$\mathbf{H}^{\mathbb{T}} = \mathbf{H}_0^{\mathbb{T}} \oplus \mathbf{H}_1^{\mathbb{T}} \oplus \ldots \oplus \mathbf{H}_s^{\mathbb{T}} \tag{6.4}$$

where $\mathbf{H}^{\mathbb{T}}$ represents the channel-wise concatenation of the attended per-task per-scale features $\mathbf{H}_s^{\mathbb{T}}$. Note that the spatial attention $SA_s^{\mathbb{T}}$ across scales used in the MSA task head is learned independently of that used when filtering the initial task features from the backbone (see Section 6.2.1). The final prediction for each task is then obtained through $\mathbf{O}^{\mathbb{T}} = \phi^{\mathbb{T}}\left(\mathbf{H}^{\mathbb{T}}\right)$, where $\phi^{\mathbb{T}}$ again maps the final number of channels $\sum_s C_s$ to the channels required by the task.

It is worth noting that the resulting task heads are completely independent from each other. Once the features leave the backbone, they do not interact with any of the features from other tasks. This simple design choice makes it trivial to attach or detach task heads after the initial training stage without affecting the existing tasks. As such, new task heads can be trained based on the fixed shared features, leading to an efficient transfer process. Furthermore, the resulting architecture scales only linearly with the number of task heads, while still maintaining performance comparable to that of highly complex methods such as MTI-Net [191].

6.3 Results

Dataset. To train *Medusa* we make use of the NYUD-v2 dataset [166]. This contains images for complex indoor scenes, in addition to labels for semantic segmentation maps, depth, surface normals and edges. Following previous approaches [199, 191], we primarily focus on the semantic segmentation and depth estimation tasks. Meanwhile, surface normal estimation and edge detection are left as auxiliary tasks that can help to learn more generic features and boost performance. We evaluate the performance of the depth predictions using the RMSE, while semantic segmentation uses the m-IoU.

Implementation details. Since we focus on dense prediction tasks, we opt for HRNet-18 [176] as the shared backbone architecture. This backbone produces feature maps at four different

scales, with downsampling factors of $\{4, 8, 16, 32\}$. We train *Medusa* using the Adam optimizer with a learning rate of 10^{-4} and a polynomial decay [30] for 100 epochs. Since we use an ImageNet [39] pretrained network, the backbone uses a lower learning rate (a factor of 10) than the task heads. Regression tasks (depth, surface normals) use the L_1 loss, while classification (semantic segmentation, human part segmentation, saliency) use cross-entropy.

6.3.1 Multi-task Evaluation

It is worth remembering that the primary objective for *Medusa* is performing UFL. We show results for this objective further on in Section 6.3.2. However, we additionally carry out experiments in traditional MTL to provide a fair comparison with existing approaches and illustrate the complexities of UFL.

In the context of MTL, we follow the procedure introduced by ASTMT [111], evaluating semantic segmentation and depth estimation on NYUD-v2. Similar to MTI-Net [191], surface normals and edge detection (N + E) are left as auxiliary tasks that can support the feature learning for the main tasks. Given multiple target tasks, the overall multi-task performance Δ_m is defined as

$$\Delta_m = \sum_{\mathbb{T}=0}^{N_{\mathbb{T}}-1} (\text{-}1)^{l^{\mathbb{T}}} \frac{M_m^{\mathbb{T}} \text{-} M_b^{\mathbb{T}}}{M_b^{\mathbb{T}}}, \tag{6.5}$$

where $l^{\mathbb{T}}$ is the label indicating whether a lower value means a better performance in that task, $M_b^{\mathbb{T}}$ is the performance of the single task baseline and $M_m^{\mathbb{T}}$ is the performance of the target MTL system we are evaluating. Intuitively, this represents the average improvement in performance relative to the single task baseline $M_b^{\mathbb{T}}$ over all tasks. The results for the single task baseline (ST) are obtained by training a completely independent network for each task.

We compare against a multi-task baseline (MT) as well as six previous SOTA approaches. The MT baseline represents a naïve MTL implementation, where the tasks share a common backbone, connected directly to each task head. Three of the SOTA baselines [124, 57, 99] use a single scale backbone based on ResNet [70], combined with the popular DeepLab-v3 [25] head. Multi-scale models [199, 191] instead employ the HRNet backbone along with the simple HRHead described previously. Since MTAN [99] is the best performing single-scale model and the most similar in concept to *Medusa* we additionally adapt it to use the HRNet backbone

Table 6.1: Multi-task Evaluation on NYUD-v2. The (N+E) column indicates the presence of auxiliary surface normals and edges tasks. The multi-task performance Δ_m represents the average increase in performance across all tasks. *Medusa* performs on par with current SOTA [191]. This is achieved while using a much lower number of computational resources (see Figure 6.4). This is due to the focus on the shared features as well as the novel lightweight MSA head.

	Backbone	Head	N+E	Seg ↑	Depth ↓	$\Delta_m\%$ ↑
ST Baseline	ResNet-18	DeepLab-v3+		35.77	0.600	+0.00
MT Baseline	ResNet-18	DeepLab-v3+		35.74	0.597	+0.12
Cross-stitch [124]	ResNet-18	DeepLab-v3+		36.01	0.600	+0.30
NDDR-CNN [57]	ResNet-18	DeepLab-v3+		34.72	0.611	-2.47
MTAN [99]	ResNet-18	DeepLab-v3+		36.00	0.594	+0.79
ST Baseline	HRNet-18	HRHead		34.57	0.606	+0.00
MT Baseline	HRNet-18	HRHead		33.21	0.614	-2.63
MTAN [99]	HRNet-18	DeepLab-v3+		35.25	0.581	+3.02
MTAN	HRNet-18	DeepLab-v3+	✓	36.19	0.567	+5.57
PAD-Net [199]	HRNet-18	HRHead		34.39	0.617	-1.23
PAD-Net	HRNet-18	HRHead	✓	35.46	0.604	+1.43
MTI-Net [191]	HRNet-18	HRHead		36.94	0.559	+7.26
MTI-Net	HRNet-18	HRHead	✓	37.40	**0.540**	**+9.48**
***Medusa* (ours)**	HRNet-18	**MSA (ours)**		36.99	0.573	+6.19
Medusa	HRNet-18	**MSA**	✓	**37.48**	0.545	+9.24

for a fairer comparison. However, the nature of MTAN's architecture still requires the use of the DeepLab-v3 head. We use the code provided by the authors of [111, 191] to train these baselines.

As shown by the results in Table 6.1, the naïve MTL approach (MT) actually results in a decrease in performance. This is due to the fact that the network is forced to share all information across all tasks even if they are unrelated. Once again, this process is known as *negative transfer* [83, 214]. It can also be seen how single-scale approaches making direct predictions [124, 57, 99], do not provide a large increase in performance. *Medusa* can be seen to outperform all existing

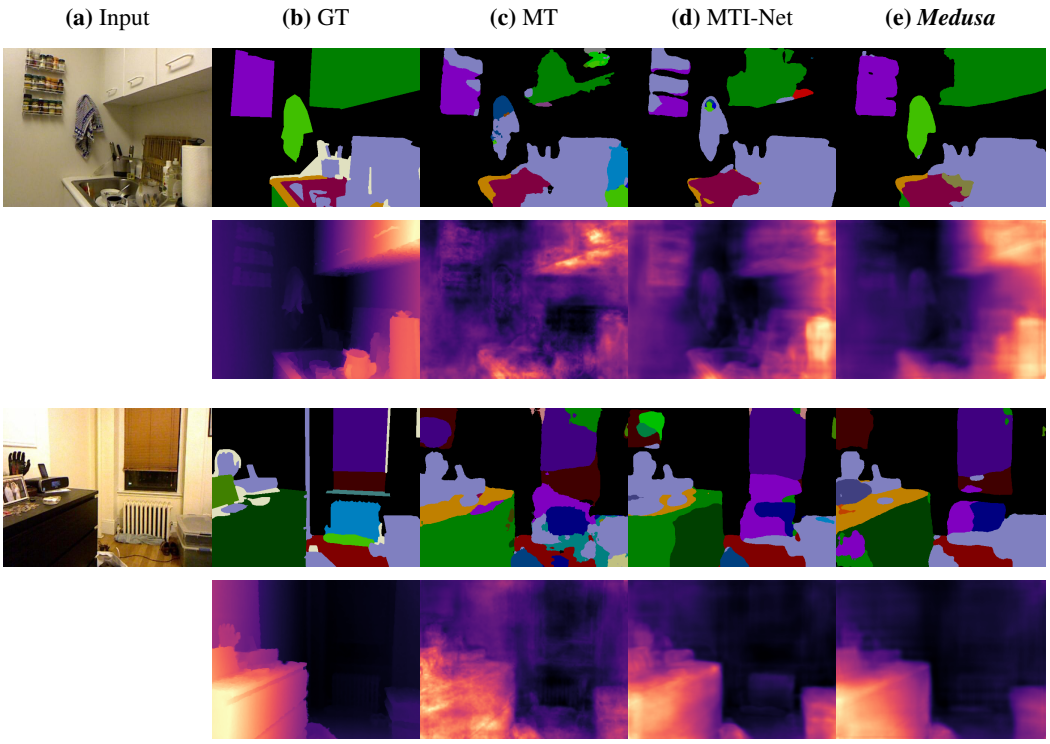


Figure 6.3: Qualitative Evaluation. Through the proposed MSA heads, *Medusa*'s predictions are both globally consistent and have well defined borders. Results are on par with the current SOTA while using less resources.

approaches with independent task heads [99]. Interestingly, we also see improvements when incorporating the auxiliary (N + E) tasks. This means that forcing the common features to encode properties useful for predicting surface normals results in features that are also better at performing depth estimation, despite the fact that the two task heads are completely unaware of each other. Finally, *Medusa* performs on par with current SOTA methods [191] that include connections between all task heads at each different scale. This is achieved while having much lower resource requirements, as we will discuss later. We show qualitative visualizations for the network predictions in Figure 6.3. As seen, the baseline MT predictions are highly noisy and produce many inconsistencies. *Medusa*'s predictions instead are generally globally consistent while still providing sharp edges in semantic segmentation. However, some objects still get misclassified and cluttered scenes are challenging to represent in high detail.

Ablation. To understand the effect of the various components of *Medusa*, we perform an ablation study. This focuses on the dual attention mechanisms incorporated into the network. To mitigate the effect of reduced model complexity, we do not simply remove the SFA modules.

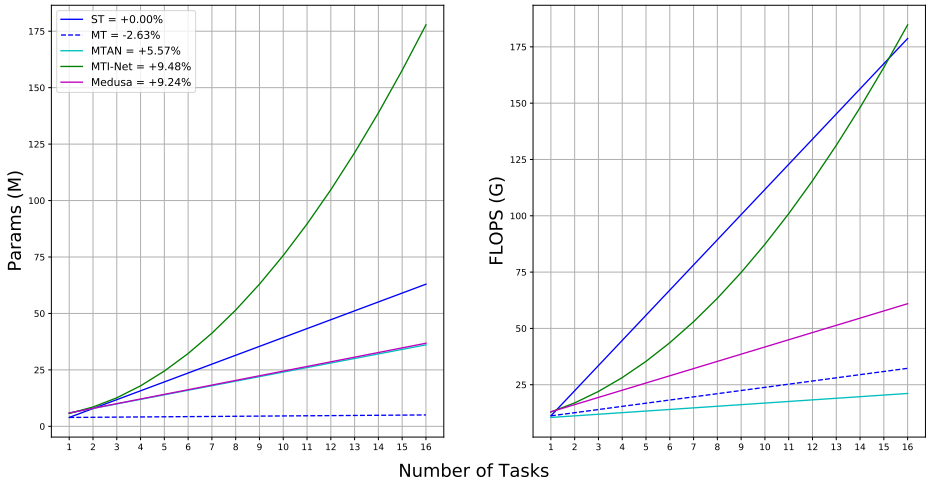
Table 6.2: Spatial Attention Ablation Study. We study the effect of the spatial attention components of *Medusa*, when all approaches use the HRNet-18 backbone. In the SFA column we replace the spatial attention block between the shared backbone and the task heads with a traditional convolutional block. The MSA task head incorporates spatial attention when combining the multi-scale task features prior to making the final prediction. Incorporating both types of attention results in a relative improvement of 37.7%.

	SFA	Head	N+E	Seg ↑	Depth ↓	$\Delta_m\%$ ↑
ST Baseline		HRHead		34.57	0.606	+0.00
MT Baseline		HRHead		33.21	0.614	-2.63
MT Baseline		**MSA**		35.58	0.598	+2.12
Medusa		HRHead	✓	36.50	0.558	+6.71
Medusa	✓	HRHead	✓	36.64	0.553	+7.31
Medusa		**MSA**	✓	37.14	0.555	+7.91
Medusa	✓	**MSA**	✓	**37.48**	**0.545**	**+9.24**

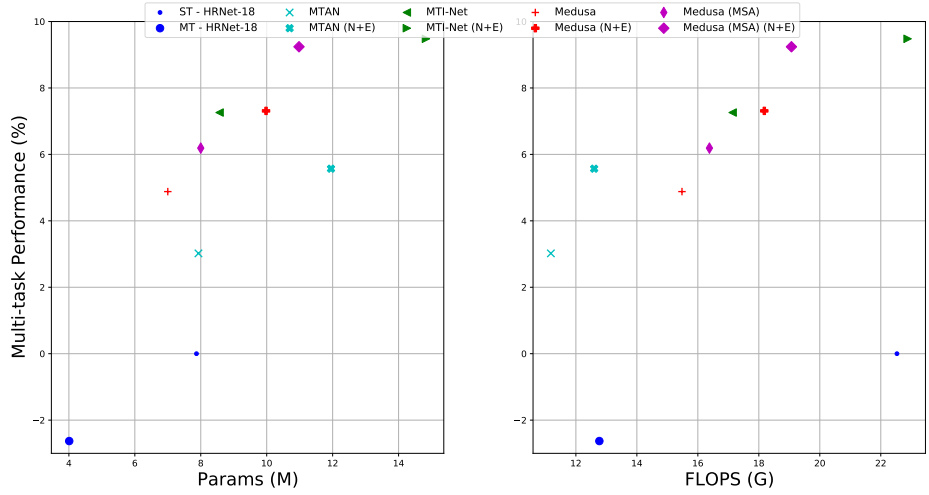
Instead we replace them with a simple convolutional block using BatchNorm and a ReLU activation. We follow a similar approach with the proposed MSA head, which incorporates attention into the basic HRHead used by previous approaches.

The results of these experiments can be found in Table 6.2, demonstrating the benefits of each attention block. Incorporating the SFA results in a consistent relative improvement across the different techniques of 8.94% and 16.81%. Meanwhile, the MSA head leads to even larger performance gains—from 17.88% to 180.60%. Most notably, the MSA head boosts the performance of the MT baseline such that it improves over the ST baseline by mitigating negative transfer. This illustrates the importance of flexibility when combining features at multiple scales, allowing the network to decide which features are more suitable within a local neighbourhood. Overall, the proposed changes lead to a relative performance increase of 37.7% w.r.t. the base model without any spatial attention. It is worth noting that these changes are very simple to implement and do not result in a large increase in resource requirements.

Resources. As discussed, the architecture used by *Medusa* is simple in concept, but is highly effective. This is due to the fact that we focus on the core challenges of multi-task and multi-scale learning, namely effective feature sharing across both tasks and scales. As shown in Figure 6.4a,



(a)



(b)

Figure 6.4: *Medusa* Resource Usage. Modelling the relationships between each pair of tasks and scales [191] results in a quadratic increase in paramters/GFLOPS w.r.t. the number of tasks. This does not scale well to an increasing number of tasks. *Medusa*'s independent task heads lead to a much more efficient scaling, while focusing on features that are more generic and reusable. Exact values can be found in Table 6.3.

this leads to an architecture that scales only linearly w.r.t. the number of tasks. We further contrast the trade-off between resource usage and performance in Figure 6.4b & Table 6.3. By sharing the computation at the backbone, *Medusa* is more efficient than the ST baseline, which creates $N_{\mathbb{T}}$ duplicate backbones. While the MT baseline is the most efficient, its performance is lacklustre due to the lack of attention and intermediate predictions. Finally, the additional connections between all pairs of tasks at each scale from MTI-Net [191] result in a quadratic

Table 6.3: *Medusa* Resource Usage. All single-scale baselines require large amounts of both parameters and FLOPS. Multi-scale baselines reduce these requirements, while improving performance significantly. However, methods with densely connected task heads [199, 191] do not scale effectively to an increasing number of tasks. *Medusa* provides the best balance between resource usage and performance.

	Backbone	Head	N+E	Params (M) ↓	Flops (G) ↓	$\Delta_m\%$ ↑
ST Baseline	ResNet-18	DeepLab-v3+		31.80	156.89	+0.00
MT Baseline	ResNet-18	DeepLab-v3+		20.63	100.49	+0.12
Cross-stitch	ResNet-18	DeepLab-v3+		31.81	156.89	+0.30
NDDR-CNN	ResNet-18	DeepLab-v3+		33.22	164.12	-2.47
MTAN	ResNet-18	DeepLab-v3+		21.65	108.48	+0.79
ST Baseline	HRNet-18	HRHead		7.87	22.53	+0.00
MT Baseline	HRNet-18	HRHead		**4.01**	12.77	-2.63
MTAN	HRNet-18	DeepLab-v3+		7.93	**11.18**	+3.02
MTAN	HRNet-18	DeepLab-v3+	✓	11.95	12.60	+5.57
PAD-Net	HRNet-18	HRHead		7.01	70.24	-1.23
PAD-Net	HRNet-18	HRHead	✓	12.16	168.90	+1.43
MTI-Net	HRNet-18	HRHead		8.57	17.14	+7.26
MTI-Net	HRNet-18	HRHead	✓	14.83	22.88	**+9.48**
***Medusa* (ours)**	HRNet-18	**MSA**		8.00	16.38	+6.19
Medusa	HRNet-18	**MSA**	✓	10.98	19.07	+9.24

increase in parameters. Even at a small number of tasks—three or more—this approach requires more parameters than the ST baseline. In other words, MTI-Net is less efficient than training entirely independent models for each task with no parameter sharing. As the number of tasks increases this gap only widens. *Medusa* reduces the number of parameters w.r.t. MTI-Net by 25.9%, with only a slight performance drop of 2.5%. On the other hand, *Medusa* has roughly the same number of parameters as MTAN, but provides a relative improvement of 65%.

Table 6.4: Universal Feature Learning. We evaluate the transfer capabilities to new tasks and new datasets (PASCAL-Context) of the highest performing MTL approaches in Table 6.1 (NYUD-v2). Since *Medusa* places a larger focus on the shared feature representation, the resulting features provide large improvements over commonly used ImageNet pretrained features (ST Baseline). This is archived while using orders of magnitude less data.

	NYUD-v2			PASCAL-Context		
	Seg ↑	Depth ↓	$\Delta_m\%$ ↑	Parts ↑	Sal ↑	$\Delta_m\%$ ↑
ST Baseline	34.57	0.606	+0.00	48.73	56.44	+0.00
MT Baseline	33.21	0.614	-2.63	36.13	51.96	-12.93
MTAN [99]	36.19	0.567	+5.57	47.37	57.84	+4.26
MTI-Net [191]	37.40	**0.540**	**+9.48**	51.50	60.19	+10.76
Medusa	**37.48**	0.545	+9.24	**52.24**	**61.91**	**+13.18**

6.3.2 Universal Feature Learning

To conclude, we evaluate *Medusa* in the proposed UFL task for which it was developed. The objective is to train a network with a wide range of tasks, such that the resulting features generalize beyond the original training set. In other words, the set of evaluation tasks is different from the training tasks. This differs from standard MTL, where we only require the features to perform well on the original training tasks.

We make use of the PASCAL-Context dataset [30] to show the transfer capabilities of *Medusa*. This means we are not only transferring to new tasks, but also to a previously unseen dataset. This dataset contains ground truth labels for semantic segmentation, human part segmentation and edge detection, as well as pseudo-ground truth labels for surface normals and saliency based on SOTA models [11, 26]. Three task are common to NYUD-v2, while two of them (human part segmentation and saliency) are unique. To carry out this evaluation we use the previous models trained on NYUD-v2 in Section 6.3.1 with the auxiliary (N+E) tasks and check their transfer capability to the new target tasks in the PASCAL-Context dataset. This is done by freezing the shared feature backbone network and adding a new task head corresponding to either saliency estimation or human part segmentation. In the case of MTI-Net, adding their proposed task head would require incorporating new connections to each of the existing tasks

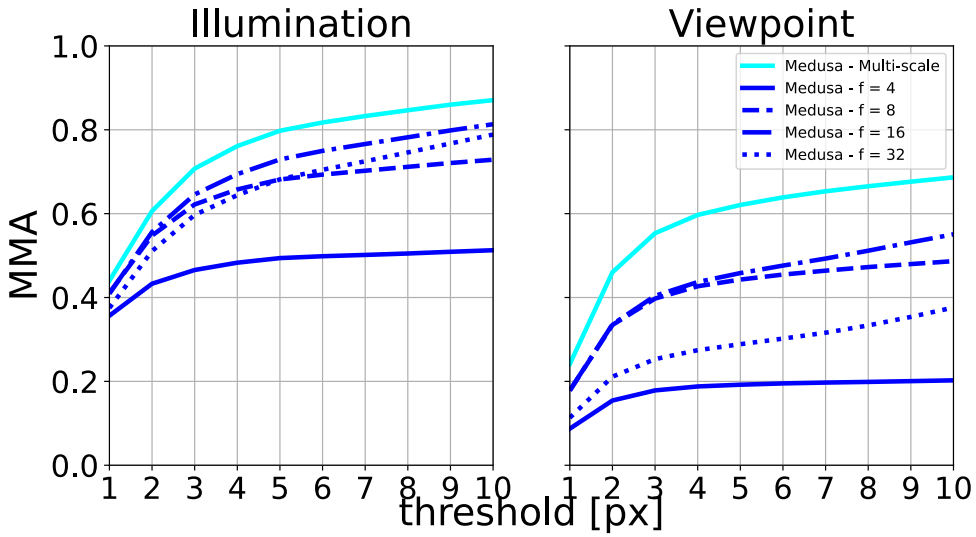
and retraining. Since this defeats the point of UFL, we instead replace the new task heads with the common task head used by other approaches. It is also worth noting that here, the single task baseline (ST) represents features obtained by pretraining on ImageNet [39], freezing the backbone and training only the target task head.

These results are shown in Table 6.4, where we include the previous MTL results on NYUD-v2 for comparison. The MT baseline is completely unable to transfer the learnt features to these new tasks. This illustrates the core difference between UFL and MTL, where the goal of MTL is to perform well only in the original training tasks. This is only exacerbated by the naïve multi-task implementation, resulting in a large amount of negative transfer between tasks. Meanwhile, *Medusa* provides the best transfer capabilities. Even through MTL performance is almost equal to MTI-Net (9.24% *vs*. 9.48%), the features learnt by *Medusa* generalize to a broader range of tasks (13.18% *vs*. 10.76%). This is due to the fact that *Medusa* places more importance on learning an effective shared representation. Each task head is capable of extracting the relevant features that provide the best performance in the target task. In contrast, MTI-Net focused on modelling the connections between task heads. As such, the features do not learn to be generic and instead focus on solving only the original set of training tasks.

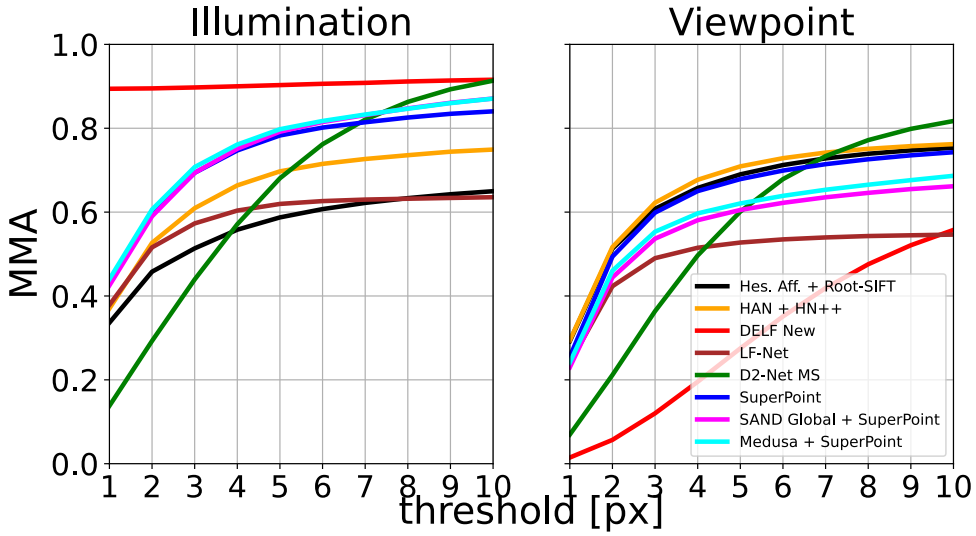
6.3.3 Image Matching

Despite not being optimized for the task of correspondence estimation, we theorize that *Medusa*'s generic features may encode properties useful for this task. To test this, we again make use of the image matching evaluation on HPatches [9] described in Section 3.2.1. This reports the MMA, defined as the percentage of correct matches given a varying threshold for the reprojection error for each keypoint. We use keypoints detected by SuperPoint [41].

Medusa does not have an explicit task head trained to produce feature descriptors. We therefore instead use the shared backbone, which produces a feature representation at multiple scales with an increasing downsampling factor. We first perform an ablation to find the most effective scale(s). As show in Figure 6.5a, each scale separately provides poor performance. Similar to the ablations from *SAND* and *Déjà-Vu*, we find that higher downsampling factors typically provide increasing performance. However, concatenating the descriptors from all scales channel-wise results in a much larger performance boost, especially under viewpoint changes.



(a)



(b)

Figure 6.5: *Medusa* HPatches Evaluation. Despite not being optimized for feature description, *Medusa* is robust to both illumination and viewpoint changes. **(a)** Concatenating *Medusa*'s feature representations over all scales improves performance drastically. **(b)** *Medusa* outperforms most feature descriptors in illumination invariance and matches *SAND*'s viewpoint performance.

Figure 6.5b shows this performance in the context of SOTA descriptors. Surprisingly, despite the lack of direct supervision, *Medusa* provides highly competitive performance. Most notably, *Medusa* is more robust to illumination changes than SuperPoint [41], HardNet [121], RootSIFT [7] and D2-Net [45] at the majority of thresholds. This shows the effectiveness of

the proposed training approach, combining multiple-tasks and multiple-scales to learn highly generic features that can be applied to many downstream tasks. In this case, this is achieved without requiring any additional training.

6.4 Conclusions

This chapter has presented a framework for learning universal features through the lens of MTL. We proposed a simple framework, focusing on the shared feature representation, that is capable of matching the MTL performance of architectures with densely connected task heads and additional feature propagation modules. Furthermore, we show that *Medusa* is capable of extending beyond the original set of training tasks, solving the problem of UFL. This addressed objective 4 of this thesis.

One of the main limitations of *Medusa* is the high annotation requirements. Currently, each image in the dataset is required to have labels for all target training tasks. Generating or annotating these labels is time consuming and expensive, especially as the size of the dataset increases. Similarly, while *Medusa* is effective at learning generic features, it still depends on the set of training tasks. It is unlikely that the shared features will transfer well to a new task if it is completely unrelated to the training tasks. Both of these issues could be mitigated by making the training process more flexible. For instance, it would be interesting to combine multiple datasets where each has a different set of training tasks. In this case, only a subset of the task heads and associated losses would be activated for each input example. Another option could be to instead perform multi-task distillation from SOTA models for multiple different tasks. This would offset the need for large amounts of labels.

Chapter 7

Conclusions and Future Work

Feature description is a long explored topic of computer vision research. Traditional hand-crafted features were hard-coded to encode attributes about the image patch with a physical meaning, such as the strength of the spatial gradients or their orientation [102, 120, 184]. Recent advances in deep learning made it possible to directly learn a more discriminative feature representation. This was complemented by the development of new loss functions [181, 121, 180], which encouraged a more effective use of the embedding space. More recently, the description process has been done in a dense manner [33, 160, 45], where the whole image is processed in a single forward pass to generate a feature descriptor for each pixel in the image.

The purpose of this thesis was to bridge the gap between *metric* features for tasks such as correspondence estimation and the wider world of computer vision. This was motivated by the observation that current *metric* feature learning approaches learn to encode the similarity between different image regions. We hypothesized these properties can be beneficial for a much wider range of tasks. The contributions in this thesis demonstrated this is indeed the case. We further showed how to manipulate the learnt feature representation and embed additional properties that are important for many tasks, leading to highly generic and robust feature descriptors. This resulted in the following objectives for this thesis:

1. To explore deep neural network solutions to dense metric feature learning and correspondence estimation.
2. To reduce the amount of labels required to train these representations whilst still showing real-world variation.
3. To explore the application of dense metric feature descriptors to a wide range of traditionally descriptive computer vision tasks.
4. To explore the simultaneous optimization of multiple tasks based on shared feature spaces.

Chapter 3 introduced *SAND* features, with the objective of training a feature representation for correspondence estimation that could be applied to multiple downstream computer vision tasks. This therefore targeted thesis objectives 1 & 3. To train these features we required LiDAR or SfM data to determine which points in the image corresponded to each other. However, this did not impose restrictions on the negative samples used in the relational loss. To this end, we introduced the concept of spatial negative mining, where the negative samples are chosen based on a pre-defined context region around the original correspondence. This training regime limited the region in which we required each descriptor to be unique and could be modified according to the properties required by the downstream task. We demonstrated application to three downstream deep learning tasks, each with different requirements: semantic segmentation, disparity estimation and visual localisation. In each case, we showed how *SAND* could improve the performance in these tasks by replacing the input image to the network with its dense feature representation. Finally, we also demonstrated how *SAND* could be used in a traditional sparse correspondence estimation task, replacing hand-crafted features in a SLAM pipeline.

Chapter 4 addressed thesis objectives 1 & 2 trying to reduce supervision requirements via *Déjà-Vu*. As previously discussed, robustness to lighting and seasonal conditions is one of the critical requirements for feature descriptors. Unfortunately, obtaining the required pixel-wise ground truth correspondences in these scenarios is still highly challenging, even if ground truth location and depth data is available. Any slight misalignment or drift in the data leads to wildly incorrect correspondences that harm the performance of the features. This is exacerbated by the presence of dynamic objects, such as pedestrians and other vehicles. *Déjà-Vu* solved this issue using weak supervision, generating (cross-)seasonal triplets of roughly aligned images based

on their GPS position. The overall similarity between images was computed using the average matching score for each pixel. This assumed that each pixel in the image should match well with only one pixel in the second image. We demonstrated *Déjà-Vu*'s performance by performing image retrieval in a cross-seasonal setting.

Chapter 5 proposed a combination of both *SAND* and *Déjà-Vu*, resulting in *DeFeat-Net*. *Déjà-Vu* removed all spatial constraints in order to train with cross-seasonal data. *DeFeat-Net* re-introduced these constrains within each season in a self-supervised manner by additionally learning depth and motion, from which pseudo-ground truth correspondences were obtained. As such, this addressed thesis objectives 1, 2 & 4 by learning general features for many tasks without ground truth correspondence supervision. Since the resulting dense features are robust to seasonal changes, they were also used to create a view synthesis warp loss. This was more stable and provided a stronger depth supervision signal than the traditional photometric consistency used in modern monocular depth approaches. We evaluated *DeFeat-Net* on the challenging RobotCar [107] dataset and showed the improved performance over the previous SOTA.

Finally, objective 4 was handled in Chapter 6 by *Medusa*. We introduced a generic and efficient framework for learning feature representations that could support many different computer vision tasks. We approached this from the perspective of MTL, where the network is trained using labels for all target tasks simultaneously. This was expanded by the concept of UFL, where we require the learnt features to generalize beyond the original training tasks. By keeping independent task heads, *Medusa* provided a flexible architecture that could easily be expanded. Furthermore, the use of spatial attention allowed the system to overcome negative transfer between unrelated tasks. As a result, the proposed approach performed on par with previous SOTA while scaling much more efficiently to an increasing number of tasks. Furthermore, the learned features were shown to transfer more effectively to new tasks on new datasets.

7.1 Directions for the Field

The contributions in this thesis will help to guide future research in the field of generic and reusable feature learning. This is most notable when combining the various ideas from multiple contributions together. This section discusses long-term future work and directions for the field guided by the contributions in this thesis.

Chapters 4 & 5 have highlighted some exciting avenues for future research. Current deep learning solutions to many computer vision tasks are not necessarily robust to adverse conditions that are commonplace in the real world. This can be due to either violated assumptions that affect the optimization objective or the large domain shift caused by drastic appearance changes. The contributions from these chapters have shown that it is possible to learn a feature space invariant to these conditions and that these features can in turn support existing frameworks and improve their performance. Future work should explore how cross-seasonal robustness can be integrated into a broader range of non-geometric tasks, if these are to be deployed in the real world. For instance, tasks such as biometric identification via face recognition, iris recognition or even gait recognition may be impacted by the current weather conditions.

These concepts also extend beyond the realm of seasonal appearance changes. For instance, it has become increasingly common to develop vision systems using alternative imaging methods, such as near/far infrared [152, 82] or hyperspectral cameras [31]. These sensors have different characteristics and failure modes that complement each other and improve the overall robustness of the systems. However, rather than treating each of these sensors independently, it is now possible to learn feature spaces that are common and shared across all different modalities. This lends itself to important research tasks related to medical imaging, such as multi-modal image registration [73, 37, 66].

Chapter 6 demonstrated the effectiveness of multi-tasking frameworks. This is a core component of many complex real-world systems, such as autonomous vehicles, smart cities, surveillance, personal robot assistants, augmented reality and more. A framework capable of efficiently learning and performing all target tasks simultaneously would be greatly beneficial. *Medusa* focused on modelling the relationships between tasks at the shared feature level, allowing for an efficient approach where task heads are independent from each other. Existing work has also shown the benefits of incorporating additional constraints between the outputs of multiple tasks [174, 17, 116, 28]. However, this has been limited to a very small number of tasks—two or three—which may use ground truth labels. One could imagine a more generalized multi-tasking system that improves performance by both sharing features within the network and ensuring consistency in the predictions based on the relationships between tasks. This could include generalized inter-data constraints, such as those used in the spatial negative mining introduced in Chapter 3, or human-inspired cues that identify useful priors or common sources of error.

One of the benefits of such an approach is the ability to dynamically add or remove task heads as required. Agents could learn low-level vision tasks—depth estimation, surface normals, edge detection, feature description, classification—alongside high-level tasks specific to the agent's goals. For instance, an autonomous vehicle could additionally learn lane segmentation, traffic light/sign detection or motion/crash prediction. During deployment, the low-level task heads not strictly required to control the vehicle can be discarded, leading to a more efficient network that nonetheless still contains the information for the low-level tasks in its shared feature representation. Even more ambitiously, a task head could be trained to predict the acceleration and steering wheel angle required to control the vehicle. Once again, all other task heads could be removed, resulting in a single end-to-end network capable of directly operating the vehicle without requiring auxiliary outputs. It is also trivial to imagine the opposite case, where we want to add new functionality to an existing system. After the system is deployed, a new task head could be attached and trained based on the frozen shared feature representation, without deteriorating the performance of previously trained task heads. It may even be possible to imagine a situation where new tasks are detected and trained on the fly, according to some existing knowledge base and rule structure.

In summary, the work presented in this thesis has opened up many new potential areas of research across the field of computer vision. We hope these contributions have helped to pave the way towards one of the holy grails of computer vision: a truly generic feature representation that can easily be applied to all seasonal conditions and serve all downstream tasks.

Bibliography

[1] Alexandre Alahi, Raphael Ortiz, and Pierre Vandergheynst. FREAK: Fast retina keypoint. In *Conference on Computer Vision and Pattern Recognition*, pages 510–517. IEEE Computer Society, jun 2012.

[2] Pablo F. Alcantarilla, Jesús Nuevo, and Adrien Bartoli. Fast explicit diffusion for accelerated features in nonlinear scale spaces. In *British Machine Vision Conference*. BMVA Press, 2013.

[3] Pablo Fernández Alcantarilla, Adrien Bartoli, and Andrew J Davison. KAZE Features. In *European Conference on Computer Vision*, pages 214–227. Springer International Publishing, 2012.

[4] Filippo Aleotti, Fabio Tosi, Matteo Poggi, and Stefano Mattoccia. Generative Adversarial Networks for Unsupervised Monocular Depth Prediction. *European Conference on Computer Vision Workshops*, pages 337–354, sep 2018.

[5] Giuseppe Amato, Paolo Bolettieri, Fabrizio Falchi, and Claudio Gennaro. Large Scale Image Retrieval Using Vector of Locally Aggregated Descriptors. *International Conference on Similarity Search and Applications*, 8199 LNCS:245–256, 2013.

[6] Relja Arandjelovic, Petr Gronat, Akihiko Torii, Tomas Pajdla, and Josef Sivic. NetVLAD: CNN Architecture for Weakly Supervised Place Recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 40(6):1437–1451, jun 2018.

[7] Relja Arandjelovic and Andrew Zisserman. Three things everyone should know to improve object retrieval. In *Conference on Computer Vision and Pattern Recognition*, pages 2911–2918. IEEE Computer Society, jun 2012.

[8] Vassileios Balntas, Edward Johns, Lilian Tang, and Krystian Mikolajczyk. PN-Net: Conjoined Triple Deep Network for Learning Local Image Descriptors. *arXiv preprint*, jan 2016.

[9] Vassileios Balntas, Karel Lenc, Andrea Vedaldi, and Krystian Mikolajczyk. HPatches: A benchmark and evaluation of handcrafted and learned local descriptors. In *Conference on Computer Vision and Pattern Recognition*, pages 3852–3861. IEEE Computer Society, nov 2017.

[10] Vassileios Balntas, Edgar Riba, Daniel Ponsa, and Krystian Mikolajczyk. Learning local feature descriptors with triplets and shallow convolutional neural networks. In *British Machine Vision Conference*, pages 119.1–119.11. BMVA Press, 2016.

[11] Aayush Bansal, Xinlei Chen, Bryan Russell, Abhinav Gupta, and Deva Ramanan. PixelNet: Representation of the pixels, by the pixels, and for the pixels, 2017.

[12] Daniel Barath, Jiri Matas, and Jana Noskova. MAGSAC: Marginalizing sample consensus. In *Conference on Computer Vision and Pattern Recognition*, pages 10189–10197. IEEE Computer Society, jun 2019.

[13] Herbert Bay, Tinne Tuytelaars, and Luc Van Gool. SURF: Speeded Up Robust Features. In *European Conference on Computer Vision*, pages 404–417. Springer International Publishing, 2006.

[14] Paul R. Beaudet. Rotationally Invariant Image Operators. In *International Joint Conference on Pattern Recognition*, pages 579–583, 1979.

[15] Matthew Brown, Gang Hua, and Simon Winder. Discriminative learning of local image descriptors. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 33(1):43–57, jan 2011.

[16] David Bruggemann, Menelaos Kanakis, and Stamatios Georgoulis. Automated Search for Resource-Efficient Branched Multi-Task Networks. In *British Machine Vision Conference*. BMVA Press, 2020.

[17] Ignas Budvytis, Marvin Teichmann, Tomas Vojir, and Roberto Cipolla. Large scale joint

semantic re-localisation and scene understanding via globally unique instance coordinate regression. In *British Machine Vision Conference*, page 31. BMVA Press, 2019.

[18] Michael Calonder, Vincent Lepetit, Christoph Strecha, and Pascal Fua. BRIEF: Binary Robust Independent Elementary Features. In *European Conference on Computer Vision*, pages 778–792. Springer International Publishing, 2010.

[19] Yuanzhouhan Cao, Zifeng Wu, and Chunhua Shen. Estimating Depth from Monocular Images as Classification Using Deep Fully Convolutional Residual Networks. *IEEE Transactions on Circuits and Systems for Video Technology*, 28(11):3174–3182, nov 2018.

[20] Nicolas Carion, Francisco Massa, Gabriel Synnaeve, Nicolas Usunier, Alexander Kirillov, and Sergey Zagoruyko. End-to-End Object Detection with Transformers. *European Conference on Computer Vision*, 12346 LNCS:213–229, aug 2020.

[21] Rich Caruana. Multitask Learning. *Machine Learning*, 28(1):41–75, 1997.

[22] Vincent Casser, Soeren Pirk, Reza Mahjourian, and Anelia Angelova. Depth Prediction without the Sensors: Leveraging Structure for Unsupervised Learning from Monocular Videos. *Conference on Artificial Intelligence*, 33(01):8001–8008, jul 2019.

[23] Vijay Chandrasekhar, Gabriel Takacs, David M Chen, Sam S Tsai, Yuriy Reznik, Radek Grzeszczuk, and Bernd Girod. Compressed Histogram of Gradients: A Low-Bitrate Descriptor. *International Journal of Computer Vision*, 96(3):384–399, 2012.

[24] Jia Ren Chang and Yong Sheng Chen. Pyramid Stereo Matching Network. In *Conference on Computer Vision and Pattern Recognition*, pages 5410–5418. IEEE Computer Society, dec 2018.

[25] Liang Chieh Chen, George Papandreou, Iasonas Kokkinos, Kevin Murphy, and Alan L Yuille. DeepLab: Semantic Image Segmentation with Deep Convolutional Nets, Atrous Convolution, and Fully Connected CRFs. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 40(4):834–848, apr 2018.

[26] Liang Chieh Chen, Yukun Zhu, George Papandreou, Florian Schroff, and Hartwig Adam. Encoder-decoder with atrous separable convolution for semantic image segmentation.

In *European Conference on Computer Vision*, volume 11211 LNCS, pages 833–851. Springer International Publishing, sep 2018.

[27] Po-Yi Chen, Alexander H. Liu, Yen-Cheng Liu, and Yu-Chiang Frank Wang. Towards Scene Understanding: Unsupervised Monocular Depth Estimation With Semantic-Aware Representation. *Conference on Computer Vision and Pattern Recognition*, 2019-June:2619–2627, jun 2019.

[28] Po Yi Chen, Alexander H. Liu, Yen Cheng Liu, and Yu Chiang Frank Wang. Towards scene understanding: Unsupervised monocular depth estimation with semantic-aware representation. *Conference on Computer Vision and Pattern Recognition*, pages 2619–2627, jun 2019.

[29] Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. A simple framework for contrastive learning of visual representations. In *International Conference on Machine Learning*, volume PartF16814, pages 1575–1585. International Machine Learning Society, feb 2020.

[30] Xianjie Chen, Roozbeh Mottaghi, Xiaobai Liu, Sanja Fidler, Raquel Urtasun, and Alan Yuille. Detect what you can: Detecting and representing objects using holistic models and body parts. In *Conference on Computer Vision and Pattern Recognition*, pages 1979–1986. IEEE Computer Society, sep 2014.

[31] Yushi Chen, Zhouhan Lin, Xing Zhao, Gang Wang, and Yanfeng Gu. Deep learning-based classification of hyperspectral data. *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, 7(6):2094–2107, 2014.

[32] Sumit Chopra, Raia Hadsell, and Yann LeCun. Learning a similarity metric discriminatively, with application to face verification. In *Conference on Computer Vision and Pattern Recognition*, volume I, pages 539–546. IEEE Computer Society, jun 2005.

[33] Christopher B Choy, JunYoung Gwak, Silvio Savarese, and Manmohan Chandraker. Universal Correspondence Network. In *Advances in Neural Information Processing Systems*, volume 29. Curran Associates, Inc., 2016.

[34] Peter Hviid Christiansen, Mikkel Fly Kragh, Yury Brodskiy, and Henrik Karstoft. UnsuperPoint: End-to-end Unsupervised Interest Point Detector and Descriptor. *arXiv preprint*, jul 2019.

[35] 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. *Conference on Computer Vision and Pattern Recognition*, pages 3213–3223, jun 2016.

[36] James L. Crowley and Alice C. Parker. A Representation for Shape Based on Peaks and Ridges in the Difference of Low-Pass Transform. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, PAMI-6(2):156–170, mar 1984.

[37] Daewon Lee, Matthias Hofmann, Florian Steinke, Yasemin Altun, Nathan D. Cahill, and Bernhard Scholkopf. Learning similarity measure for multi-modal 3D image registration. *Conference on Computer Vision and Pattern Recognition*, pages 186–193, mar 2009.

[38] Yann N Dauphin, Angela Fan, Michael Auli, and David Grangier. Language Modeling with Gated Convolutional Networks. In *International Conference on Machine Learning*, pages 933–941. PMLR, jul 2017.

[39] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. ImageNet: A large-scale hierarchical image database. In *Conference on Computer Vision and Pattern Recognition*, pages 248–255. IEEE Computer Society, mar 2010.

[40] Daniel DeTone, Tomasz Malisiewicz, and Andrew Rabinovich. Self-Improving Visual Odometry. *arXiv preprint*, dec 2018.

[41] Daniel DeTone, Tomasz Malisiewicz, and Andrew Rabinovich. SuperPoint: Self-supervised interest point detection and description. In *Conference on Computer Vision and Pattern Recognition Workshops*, pages 337–349. IEEE Computer Society, dec 2018.

[42] Jingming Dong and Stefano Soatto. Domain-size pooling in local descriptors: DSP-SIFT. In *Conference on Computer Vision and Pattern Recognition*, pages 5097–5106. IEEE Computer Society, oct 2015.

[43] Michael Donoser and Horst Bischof. Efficient Maximally Stable Extremal Region (MSER) tracking. In *Conference on Computer Vision and Pattern Recognition*, volume 1, pages 553–560. IEEE Computer Society, jun 2006.

[44] Alexey Dosovitskiy, Jost Tobias Springenberg, and Thomas Brox. Unsupervised feature learning by augmenting single images. In *International Conference on Learning Representations Workshops*, dec 2014.

[45] Mihai Dusmanu, Ignacio Rocco, Tomas Pajdla, Marc Pollefeys, Josef Sivic, Akihiko Torii, and Torsten Sattler. D2-Net: A trainable CNN for joint description and detection of local features. In *Conference on Computer Vision and Pattern Recognition*, pages 8084–8093. IEEE Computer Society, jun 2019.

[46] Kshitij Dwivedi and Gemma Roig. Representation similarity analysis for efficient task taxonomy & transfer learning. In *Conference on Computer Vision and Pattern Recognition*, volume 2019-June, pages 12379–12388. IEEE Computer Society, jun 2019.

[47] David Eigen and Rob Fergus. Predicting Depth, Surface Normals and Semantic Labels with a Common Multi-scale Convolutional Architecture. In *International Conference on Computer Vision*, pages 2650–2658. IEEE Computer Society, 2015.

[48] David Eigen, Christian Puhrsch, and Rob Fergus. Depth Map Prediction from a Single Image using a Multi-Scale Deep Network. In *Advances in Neural Information Processing Systems*, volume 27. Curran Associates, Inc., 2014.

[49] Jakob Engel, Vladlen Koltun, and Daniel Cremers. Direct Sparse Odometry. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 40(3):611–625, mar 2018.

[50] Bin Fan, Fuchao Wu, and Zhanyi Hu. Aggregating gradient distributions into intensity orders: A novel local image descriptor. In *Conference on Computer Vision and Pattern Recognition*, pages 2377–2384. IEEE Computer Society, jun 2011.

[51] Mohammed E Fathy, Quoc-Huy Tran, M Zeeshan Zia, Paul Vernaza, and Manmohan Chandraker. Hierarchical Metric Learning and Matching for 2D and 3D Geometric Correspondences. In *European Conference on Computer Vision*, pages 832–850, Cham, 2018. Springer International Publishing.

[52] Philipp Fischer, Alexey Dosovitskiy, and Thomas Brox. Descriptor Matching with Convolutional Neural Networks: a Comparison to SIFT. *arXiv preprint*, may 2014.

[53] Martin A. Fischler and Robert C. Bolles. Random sample consensus: A Paradigm for Model Fitting with Applications to Image Analysis and Automated Cartography. *Communications of the ACM*, 24(6):381–395, jun 1981.

[54] John Flynn, Ivan Neulander, James Philbin, and Noah Snavely. Deep stereo: Learning to predict new views from the world's imagery. In *Conference on Computer Vision and Pattern Recognition*, volume 2016-Decem, pages 5515–5524. IEEE Computer Society, dec 2016.

[55] Wolfgang Förstner and Eberhard Gülch. A fast operator for detection and precise location of distinct points, corners and centres of circular features. In *ISPRS Itercommission Conference on Fast Processing of Photogrammetric Data*, pages 281–305. Interlaken, 1987.

[56] Huan Fu, Mingming Gong, Chaohui Wang, Kayhan Batmanghelich, and Dacheng Tao. Deep Ordinal Regression Network for Monocular Depth Estimation. *Conference on Computer Vision and Pattern Recognition*, pages 2002–2011, dec 2018.

[57] Yuan Gao, Jiayi Ma, Mingbo Zhao, Wei Liu, and Alan L Yuille. NDDR-CNN: Layerwise feature fusing in multi-task cnns by neural discriminative dimensionality reduction. In *Conference on Computer Vision and Pattern Recognition*, pages 3200–3209. IEEE Computer Society, jun 2019.

[58] Ravi Garg, Vijay Kumar B.G., Gustavo Carneiro, and Ian Reid. Unsupervised CNN for Single View Depth Estimation: Geometry to the Rescue. *European Conference on Computer Vision*, pages 740–756, 2016.

[59] P. Gaussier and J.-P. Cocquerez. Neural networks for complex scene recognition: simulation of a visual system with several cortical areas. In *International Joint Conference on Neural Networks*, pages 233–259. IEEE Computer Society, jan 2003.

[60] Andreas Geiger, Philip Lenz, and Raquel Urtasun. Are we ready for autonomous driving?

the KITTI vision benchmark suite. In *Conference on Computer Vision and Pattern Recognition*, pages 3354–3361. IEEE Computer Society, jun 2012.

[61] Clement Godard, Oisin Mac Aodha, and Gabriel J. Brostow. Unsupervised Monocular Depth Estimation with Left-Right Consistency. *Conference on Computer Vision and Pattern Recognition*, pages 6602–6611, jul 2017.

[62] Clement Godard, Oisin Mac Aodha, Michael Firman, and Gabriel Brostow. Digging Into Self-Supervised Monocular Depth Estimation. *International Conference on Computer Vision*, 2019-Octob:3827–3837, oct 2019.

[63] Pengsheng Guo, Chen-Yu Lee, and Daniel Ulbricht. Learning to Branch for Multi-Task Learning. *International Conference on Machine Learning*, 2020.

[64] Xufeng Han, Thomas Leung, Yangqing Jia, Rahul Sukthankar, and Alexander C. Berg. MatchNet: Unifying feature and metric learning for patch-based matching. In *Conference on Computer Vision and Pattern Recognition*, pages 3279–3286. IEEE Computer Society, oct 2015.

[65] C Harris and M Stephens. A Combined Corner and Edge Detector. In *Alvey Vision Conference*, pages 23.1–23.6. Alvey Vision Club, 1988.

[66] Grant Haskins, Uwe Kruger, and Pingkun Yan. Deep learning in medical image registration: a survey. *Machine Vision and Applications*, 31(1):1–18, jan 2020.

[67] Tal Hassner, Shay Filosof, Viki Mayzels, and Lihi Zelnik-Manor. SIFTing through scales. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 39(7):1431–1443, jul 2017.

[68] Kaiming He, Georgia Gkioxari, Piotr Dollar, and Ross Girshick. Mask R-CNN. *International Conference on Computer Vision*, pages 2980–2988, dec 2017.

[69] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Spatial Pyramid Pooling in Deep Convolutional Networks for Visual Recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 37(9):1904–1916, sep 2015.

[70] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Conference on Computer Vision and Pattern Recognition*, pages 770–778. IEEE Computer Society, dec 2016.

[71] Kun He, Yan Lu, and Stan Sclaroff. Local Descriptors Optimized for Average Precision. In *Conference on Computer Vision and Pattern Recognition*, pages 596–605. IEEE Computer Society, dec 2018.

[72] Marko Heikkilä, Matti Pietikäinen, and Cordelia Schmid. Description of Interest Regions with Center-Symmetric Local Binary Patterns. In *Computer Vision, Graphics and Image Processing*, pages 58–69. Springer International Publishing, 2006.

[73] Derek L G Hill, Philipp G Batchelor, Mark Holden, and David J Hawkes. Medical image registration. *Physics in Medicine & Biology*, 46(3):R1, mar 2001.

[74] Elad Hoffer and Nir Ailon. Deep Metric Learning Using Triplet Network. In *International Workshop on Similarity-based Pattern Recognition*, pages 84–92. Springer International Publishing, 2015.

[75] Xinyu Huang, Peng Wang, Xinjing Cheng, Dingfu Zhou, Qichuan Geng, and Ruigang Yang. The ApolloScape Open Dataset for Autonomous Driving and Its Application. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 42(10):2702–2719, oct 2020.

[76] Max Jaderberg, Karen Simonyan, Andrew Zisserman, and Koray Kavukcuoglu. Spatial Transformer Networks. In *Advances in Neural Information Processing Systems*, volume 28. Curran Associates, Inc., 2015.

[77] Eric Jang, Shixiang Gu, and Ben Poole. Categorical Reparameterization with Gumbel-Softmax. In *International Conference on Learning Representations*, nov 2017.

[78] Shihao Jiang, Dylan Campbell, Yao Lu, Hongdong Li, and Richard Hartley. Learning To Estimate Hidden Motions With Global Motion Aggregation. In *International Conference on Computer Vision*, pages 9772–9781. IEEE Computer Society, 2021.

[79] Yan Ke and Rahul Sukthankar. PCA-SIFT: A More Distinctive Representation for Local

Image Descriptors. *Conference on Computer Vision and Pattern Recognition*, pages 506–513, jun 2004.

[80] Alex Kendall, Matthew Grimes, and Roberto Cipolla. PoseNet: A Convolutional Network for Real-Time 6-DOF Camera Relocalization. In *International Conference on Computer Vision*, pages 2938–2946, 2015.

[81] Alex Kendall, Hayk Martirosyan, Saumitro Dasgupta, Peter Henry, Ryan Kennedy, Abraham Bachrach, and Adam Bry. End-to-End Learning of Geometry and Context for Deep Stereo Regression. *International Conference on Computer Vision*, pages 66–75, oct 2017.

[82] Seungryong Kim, Dongbo Min, Bumsub Ham, Seungchul Ryu, Minh N. Do, and Kwanghoon Sohn. DASC: Dense adaptive self-correlation descriptor for multi-modal and multi-spectral correspondence. *Conference on Computer Vision and Pattern Recognition*, 07-12-June:2103–2112, oct 2015.

[83] Iasonas Kokkinos. UberNet: Training a universal convolutional neural network for Low-, Mid-, and high-level vision using diverse datasets and limited memory. In *Conference on Computer Vision and Pattern Recognition*, pages 5454–5463. IEEE Computer Society, nov 2017.

[84] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. ImageNet Classification with Deep Convolutional Neural Networks. In *Advances in Neural Information Processing Systems*, volume 25. Curran Associates, Inc., 2012.

[85] B. G.Vijay Kumar, Gustavo Carneiro, and Ian Reid. Learning local image descriptors with deep siamese and triplet convolutional networks by minimizing global loss functions. In *Conference on Computer Vision and Pattern Recognition*, pages 5385–5394. IEEE Computer Society, dec 2016.

[86] Lubor Ladicky, Jianbo Shi, and Marc Pollefeys. Pulling Things out of Perspective. *Conference on Computer Vision and Pattern Recognition*, pages 89–96, jun 2014.

[87] Ľubor Ladický, Christian Häne, and Marc Pollefeys. Learning the Matching Function. *arXiv preprint*, feb 2015.

[88] Axel Barroso Laguna, Edgar Riba, Daniel Ponsa, and Krystian Mikolajczyk. Key.Net: Keypoint detection by handcrafted and learned CNN filters. In *International Conference on Computer Vision*, pages 5835–5843. IEEE Computer Society, oct 2019.

[89] Iro Laina, Christian Rupprecht, Vasileios Belagiannis, Federico Tombari, and Nassir Navab. Deeper Depth Prediction with Fully Convolutional Residual Networks. *International Conference on 3D Vision*, pages 239–248, oct 2016.

[90] Mans Mans Larsson, Erik Stenborg, Lars Hammarstrand, Marc Pollefeys, Torsten Sattler, and Fredrik Kahl. A Cross-Season Correspondence Dataset for Robust Semantic Segmentation. *Conference on Computer Vision and Pattern Recognition*, pages 9524–9534, jun 2019.

[91] Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2323, 1998.

[92] Karel Lenc and Andrea Vedaldi. Learning Covariant Feature Detectors. In Gang Hua and Hervé Jégou, editors, *European Conference on Computer Vision Workshops*, pages 100–117, Cham, 2016. Springer International Publishing.

[93] Stefan Leutenegger, Margarita Chli, and Roland Y. Siegwart. BRISK: Binary Robust invariant scalable keypoints. In *International Conference on Computer Vision*, pages 2548–2555. IEEE Computer Society, nov 2011.

[94] Ruihao Li, Sen Wang, Zhiqiang Long, and Dongbing Gu. UnDeepVO: Monocular Visual Odometry Through Unsupervised Deep Learning. *International Conference on Robotics and Automation*, pages 7286–7291, sep 2018.

[95] Zhengqi Li and Noah Snavely. MegaDepth: Learning Single-View Depth Prediction from Internet Photos. In *Conference on Computer Vision and Pattern Recognition*, pages 2041–2050. IEEE Computer Society, dec 2018.

[96] Tony Lindeberg. Direct estimation of affine image deformations using visual front-end operations with automatic scale selection. In *International Conference on Computer Vision*, pages 134–141. IEEE, jun 1995.

[97] Philipp Lindenberger, Paul-Edouard Sarlin, Viktor Larsson, and Marc Pollefeys. Pixel-Perfect Structure-From-Motion With Featuremetric Refinement. In *Conference on Computer Vision and Pattern Recognition*, pages 5987–5997. IEEE Computer Society, 2021.

[98] Fayao Liu, Chunhua Shen, and Guosheng Lin. Deep convolutional neural fields for depth estimation from a single image. *Conference on Computer Vision and Pattern Recognition*, 07-12-June:5162–5170, jun 2015.

[99] Shikun Liu, Edward Johns, and Andrew J Davison. End-to-end multi-task learning with attention. In *Conference on Computer Vision and Pattern Recognition*, pages 1871–1880. IEEE Computer Society, jun 2019.

[100] Jonathan Long, Evan Shelhamer, and Trevor Darrell. Fully convolutional networks for semantic segmentation. In *Conference on Computer Vision and Pattern Recognition*, volume 07-12-June, pages 431–440. IEEE Computer Society, oct 2015.

[101] David G. Lowe. Object recognition from local scale-invariant features. In *International Conference on Computer Vision*, pages 1150–1157. IEEE Computer Society, sep 1999.

[102] David G Lowe. Distinctive Image Features from Scale-Invariant Keypoints. *International Journal of Computer Vision*, 60(2):91–110, 2004.

[103] Yongxi Lu, Abhishek Kumar, Shuangfei Zhai, Yu Cheng, Tara Javidi, and Rogerio Feris. Fully-adaptive Feature Sharing in Multi-Task Networks with Applications in Person Attribute Classification. In *Conference on Computer Vision and Pattern Recognition*, pages 5334–5343. IEEE Computer Society, 2017.

[104] Zixin Luo, Tianwei Shen, Lei Zhou, Jiahui Zhang, Yao Yao, Shiwei Li, Tian Fang, and Long Quan. Contextdesc: Local descriptor augmentation with cross-modality context. In *Conference on Computer Vision and Pattern Recognition*, pages 2522–2531. IEEE Computer Society, jun 2019.

[105] Zixin Luo, Tianwei Shen, Lei Zhou, Siyu Zhu, Runze Zhang, Yao Yao, Tian Fang, and Long Quan. GeoDesc: Learning Local Descriptors by Integrating Geometry Constraints. In *European Conference on Computer Vision*, pages 170–185, Cham, 2018. Springer International Publishing.

[106] Zixin Luo, Lei Zhou, Xuyang Bai, Hongkai Chen, Jiahui Zhang, Yao Yao, Shiwei Li, Tian Fang, and Long Quan. ASLFeat: Learning local features of accurate shape and localization. In *Conference on Computer Vision and Pattern Recognition*, pages 6588–6597. IEEE Computer Society, jun 2020.

[107] Will Maddern, Geoffrey Pascoe, Chris Linegar, and Paul Newman. 1 year, 1000 km: The Oxford RobotCar dataset:. *International Journal of Robotics Research*, 36(1):3–15, nov 2016.

[108] V. Madhu Babu, Kaushik Das, Anima Majumdar, and Swagat Kumar. UnDEMoN: Unsupervised Deep Network for Depth and Ego-Motion Estimation. *International Conference on Intelligent Robots and Systems*, pages 1082–1088, dec 2018.

[109] Reza Mahjourian, Martin Wicke, and Anelia Angelova. Unsupervised Learning of Depth and Ego-Motion from Monocular Video Using 3D Geometric Constraints. *Conference on Computer Vision and Pattern Recognition*, pages 5667–5675, jun 2018.

[110] Elmar Mair, Gregory D Hager, Darius Burschka, Michael Suppa, and Gerhard Hirzinger. Adaptive and Generic Corner Detection Based on the Accelerated Segment Test. In *European Conference on Computer Vision*, pages 183–196. Springer International Publishing, 2010.

[111] Kevis Kokitsi Maninis, Ilija Radosavovic, and Iasonas Kokkinos. Attentive single-tasking of multiple tasks. In *Conference on Computer Vision and Pattern Recognition*, pages 1851–1860. IEEE Computer Society, jun 2019.

[112] J Matas, O Chum, M Urban, and T Pajdla. Robust Wide Baseline Stereo from Maximally Stable Extremal Regions. In *British Machine Vision Conference*, pages 36.1–36.10. BMVA Press, 2002.

[113] Nikolaus Mayer, Eddy Ilg, Philip Hausser, Philipp Fischer, Daniel Cremers, Alexey Dosovitskiy, and Thomas Brox. A Large Dataset to Train Convolutional Networks for Disparity, Optical Flow, and Scene Flow Estimation. *Conference on Computer Vision and Pattern Recognition*, pages 4040–4048, jun 2016.

[114] Michael McCloskey and Neal J. Cohen. Catastrophic Interference in Connectionist Networks: The Sequential Learning Problem. *Psychology of Learning and Motivation - Advances in Research and Theory*, 24(C):109–165, jan 1989.

[115] Roey Mechrez, Itamar Talmi, and Lihi Zelnik-Manor. The Contextual Loss for Image Transformation with Non-aligned Data. In *European Conference on Computer Vision*, pages 800–815. Springer International Publishing, 2018.

[116] Oscar Mendez, Simon Hadfield, Nicolas Pugeault, and Richard Bowden. SeDAR - Semantic detection and ranging: Humans can localise without lidar, can robots? *International Conference on Robotics and Automation*, pages 6053–6060, sep 2018.

[117] S. Mahdi H. Miangoleh, Sebastian Dille, Long Mai, Sylvain Paris, and Yagiz Aksoy. Boosting Monocular Depth Estimation Models to High-Resolution via Content-Adaptive Multi-Resolution Merging. In *Conference on Computer Vision and Pattern Recognition*, pages 9685–9694, 2021.

[118] Krystian Mikolajczyk and Cordelia Schmid. An Affine Invariant Interest Point Detector. In *European Conference on Computer Vision*, pages 128–142. Springer International Publishing, 2002.

[119] Krystian Mikolajczyk and Cordelia Schmid. Scale & Affine Invariant Interest Point Detectors. *International Journal of Computer Vision*, 60(1):63–86, 2004.

[120] Krystian Mikolajczyk and Cordelia Schmid. A performance evaluation of local descriptors. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 27(10):1615–1630, oct 2005.

[121] Anastasiia Mishchuk, Dmytro Mishkin, Filip Radenovic, and Jiri Matas. Working hard to know your neighbor's margins: Local descriptor learning loss. In *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc., 2017.

[122] Dmytro Mishkin, Jiri Matas, and Michal Perdoch. MODS: Fast and robust method for two-view matching. *Computer Vision and Image Understanding*, 2015.

[123] Dmytro Mishkin, Filip Radenović, and Jiri Matas. Repeatability Is Not Enough: Learning

Affine Regions via Discriminability. In *European Conference on Computer Vision*, pages 287–304, Cham, 2018. Springer International Publishing.

[124] Ishan Misra, Abhinav Shrivastava, Abhinav Gupta, and Martial Hebert. Cross-Stitch Networks for Multi-task Learning. In *Conference on Computer Vision and Pattern Recognition*, volume 2016-Decem, pages 3994–4003. IEEE Computer Society, dec 2016.

[125] Hans Moravec. Obstacle Avoidance and Navigation in the Real World by a Seeing Robot Rover. Technical report, Stanford University, Pittsburgh, PA, sep 1980.

[126] Raul Mur-Artal and Juan D. Tardos. ORB-SLAM2: An Open-Source SLAM System for Monocular, Stereo, and RGB-D Cameras. *IEEE Transactions on Robotics*, 33(5):1255–1262, oct 2017.

[127] Kevin Musgrave, Serge Belongie, and Ser-Nam Lim. A Metric Learning Reality Check. *European Conference on Computer Vision*, 12370 LNCS:681–699, aug 2020.

[128] Richard A. Newcombe, Dieter Fox, and Steven M. Seitz. DynamicFusion: Reconstruction and tracking of non-rigid scenes in real-time. In *Conference on Computer Vision and Pattern Recognition*, pages 343–352. IEEE Computer Society, oct 2015.

[129] Richard A. Newcombe, Shahram Izadi, Otmar Hilliges, David Molyneaux, David Kim, Andrew J. Davison, Pushmeet Kohli, Jamie Shotton, Steve Hodges, and Andrew Fitzgibbon. KinectFusion: Real-time dense surface mapping and tracking. In *International Symposium on Mixed and Augmented Reality*, pages 127–136. IEEE Computer Society, oct 2011.

[130] David Nistér and Henrik Stewénius. Linear Time Maximally Stable Extremal Regions. In *European Conference on Computer Vision*, pages 183–196. Springer International Publishing, 2008.

[131] Hyeonwoo Noh, Andre Araujo, Jack Sim, Tobias Weyand, and Bohyung Han. Large-Scale Image Retrieval with Attentive Deep Local Features. In *International Conference on Computer Vision*, pages 3476–3485. IEEE Computer Society, dec 2017.

[132] Timo Ojala, Topi Mäenpää, and Matti Pietikäinen. Texture classification by multi-predicate local binary pattern operators. *International Conference on Pattern Recognition*, 15(3):939–942, sep 2000.

[133] Yuki Ono, Eduard Trulls, Pascal Fua, and Kwang Moo Yi. LF-Net: Learning Local Features from Images. In *Advances in Neural Information Processing Systems*, volume 31. Curran Associates, Inc., 2018.

[134] Art B Owen. A robust hybrid of lasso and ridge regression. *Contemporary Mathematics*, 443(7):59–72, 2007.

[135] Jiahao Pang, Wenxiu Sun, Jimmy SJ. Ren, Chengxi Yang, and Qiong Yan. Cascade Residual Learning: A Two-Stage Convolutional Neural Network for Stereo Matching. *International Conference on Computer Vision Workshop*, 2018-Janua:878–886, oct 2017.

[136] Andrea Pilzer, Stephane Lathuiliere, Nicu Sebe, and Elisa Ricci. Refine and Distill: Exploiting Cycle-Inconsistency and Knowledge Distillation for Unsupervised Monocular Depth Estimation. *Conference on Computer Vision and Pattern Recognition*, pages 9760–9769, jun 2019.

[137] Andrea Pilzer, Dan Xu, Mihai Puscas, Elisa Ricci, and Nicu Sebe. Unsupervised Adversarial Depth Estimation Using Cycled Generative Networks. *International Conference on 3D Vision*, pages 587–595, sep 2018.

[138] Taihu Pire, Thomas Fischer, Javier Civera, Pablo De Cristoforis, and Julio Jacobo Berlles. Stereo parallel tracking and mapping for robot localization. In *International Conference on Intelligent Robots and Systems*, pages 1373–1378. IEEE Computer Society, dec 2015.

[139] William H Press, Brian P Flannery, Saul A Teukolsky, and William T Vetterling. *Numerical Recipes in C: The Art of Scientific Computing*. Cambridge University Press, USA, 1988.

[140] Filip Radenović, Giorgos Tolias, and Ondřej Chum. Deep Shape Matching. *European Conference on Computer Vision*, 11209 LNCS:774–791, 2018.

[141] Rahul Raguram, Ondrej Chum, Marc Pollefeys, Jiri Matas, and Jan Michael Frahm.

USAC: A universal framework for random sample consensus. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 35(8):2022–2038, 2013.

[142] Anurag Ranjan, Varun Jampani, Lukas Balles, Kihwan Kim, Deqing Sun, Jonas Wulff, and Michael J. Black. Competitive Collaboration: Joint Unsupervised Learning of Depth, Camera Motion, Optical Flow and Motion Segmentation. *Conference on Computer Vision and Pattern Recognition*, pages 12232–12241, jun 2019.

[143] Roger Ratcliff. Connectionist models of recognition memory: constraints imposed by learning and forgetting functions. *Psychological review*, 97(2):285–308, 1990.

[144] Joseph Redmon, Santosh Divvala, Ross Girshick, and Ali Farhadi. You only look once: Unified, real-time object detection. *Conference on Computer Vision and Pattern Recognition*, pages 779–788, dec 2016.

[145] Jerome Revaud, Cesar De Souza, Martin Humenberger, and Philippe Weinzaepfel. R2D2: Reliable and Repeatable Detector and Descriptor. In H Wallach, H Larochelle, A Beygelzimer, F d\textquotesingle Alché-Buc, E Fox, and R Garnett, editors, *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc., 2019.

[146] Ignacio Rocco, Mircea Cimpoi, Relja Arandjelović, Akihiko Torii, Tomas Pajdla, and Josef Sivic. Neighbourhood Consensus Networks. In *Advances in Neural Information Processing Systems*, volume 31. Curran Associates, Inc., 2018.

[147] Edward Rosten and Tom Drummond. Fusing points and lines for high performance tracking. In *International Conference on Computer Vision*, volume II, pages 1508–1515. IEEE Computer Society, oct 2005.

[148] Ethan Rublee, Vincent Rabaud, Kurt Konolige, and Gary Bradski. ORB: An efficient alternative to SIFT or SURF. In *International Conference on Computer Vision*, pages 2564–2571. IEEE Computer Society, nov 2011.

[149] Sebastian Ruder. An Overview of Multi-Task Learning in Deep Neural Networks. *arXiv preprint*, jun 2017.

[150] Sebastian Ruder, Joachim Bingel, Isabelle Augenstein, and Anders Søgaard. Latent multi-task architecture learning. In *Conference on Artificial Intelligence*, pages 4822–4829. AAAI Press, 2019.

[151] David E. Rumelhart, Geoffrey E. Hinton, and Ronald J. Williams. Learning representations by back-propagating errors. *Nature*, 323(6088):533–536, 1986.

[152] Sajid Saleem and Robert Sablatnig. A robust SIFT descriptor for multispectral images. *IEEE Signal Processing Letters*, 21(4):400–403, apr 2014.

[153] Leo Sampaio Ferraz Ribeiro, Tu Bui, John Collomosse, and Moacir Ponti. Sketchformer: Transformer-Based Representation for Sketched Structure. *Conference on Computer Vision and Pattern Recognition*, pages 14141–14150, 2020.

[154] Mark Sandler, Andrew Howard, Menglong Zhu, Andrey Zhmoginov, and Liang Chieh Chen. MobileNetV2: Inverted Residuals and Linear Bottlenecks. In *Conference on Computer Vision and Pattern Recognition*, pages 4510–4520. IEEE Computer Society, dec 2018.

[155] Paul Edouard Sarlin, Cesar Cadena, Roland Siegwart, and Marcin Dymczyk. From coarse to fine: Robust hierarchical localization at large scale. In *Conference on Computer Vision and Pattern Recognition*, pages 12708–12717. IEEE Computer Society, jun 2019.

[156] Paul Edouard Sarlin, Daniel Detone, Tomasz Malisiewicz, and Andrew Rabinovich. SuperGlue: Learning Feature Matching with Graph Neural Networks. In *Conference on Computer Vision and Pattern Recognition*, pages 4937–4946. IEEE Computer Society, jun 2020.

[157] Torsten Sattler, Will Maddern, Carl Toft, Akihiko Torii, Lars Hammarstrand, Erik Stenborg, Daniel Safari, Masatoshi Okutomi, Marc Pollefeys, Josef Sivic, Fredrik Kahl, and Tomas Pajdla. Benchmarking 6DOF Outdoor Visual Localization in Changing Conditions. *Conference on Computer Vision and Pattern Recognition*, pages 8601–8610, jun 2018.

[158] Ashutosh Saxena, Min Sun, and Andrew Y. Ng. Make3D: Learning 3D Scene Structure from a Single Still Image. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 31(05):824–840, may 2009.

[159] Cordelia Schmid and Roger Mohr. Local grayvalue invariants for image retrieval. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 19(5):530–535, may 1997.

[160] Tanner Schmidt, Richard Newcombe, and Dieter Fox. Self-Supervised Visual Descriptor Learning for Dense Correspondence. *IEEE Robotics and Automation Letters*, 2(2):420–427, apr 2017.

[161] Johannes L. Schonberger and Jan Michael Frahm. Structure-from-Motion Revisited. *Conference on Computer Vision and Pattern Recognition*, pages 4104–4113, dec 2016.

[162] Florian Schroff, Dmitry Kalenichenko, and James Philbin. FaceNet: A unified embedding for face recognition and clustering. *Conference on Computer Vision and Pattern Recognition*, pages 815–823, oct 2015.

[163] Rene Schuster, Oliver Wasenmuller, Christian Unger, and DIdier Stricker. SDC-Stacked dilated convolution: A unified descriptor network for dense matching tasks. In *Conference on Computer Vision and Pattern Recognition*, pages 2551–2560. IEEE Computer Society, jun 2019.

[164] Aashish Sharma, Loong-Fah Cheong, Lionel Heng, and Robby T. Tan. Nighttime Stereo Depth Estimation using Joint Translation-Stereo Learning: Light Effects and Uninformative Regions. *International Conference on 3D Vision*, pages 23–31, nov 2020.

[165] Jianbo Shi and Carlo Tomasi. Good features to track. In *Conference on Computer Vision and Pattern Recognition*, pages 593–600. IEEE Computer Society, jan 1994.

[166] Nathan Silberman, Derek Hoiem, Pushmeet Kohli, and Rob Fergus. Indoor segmentation and support inference from RGBD images. In *European Conference on Computer Vision*, volume 7576 LNCS, pages 746–760. Springer International Publishing, 2012.

[167] E Simo-Serra, E Trulls, L Ferraz, I Kokkinos, P Fua, and F Moreno-Noguer. Discriminative Learning of Deep Convolutional Feature Point Descriptors. In *International Conference on Computer Vision*, pages 118–126. IEEE Computer Society, dec 2015.

[168] Karen Simonyan, Andrea Vedaldi, and Andrew Zisserman. Descriptor Learning Using Convex Optimisation. In *European Conference on Computer Vision*, pages 243–256. Springer International Publishing, 2012.

[169] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. In *International Conference on Learning Representations*, sep 2015.

[170] Josef Sivic and Andrew Zisserman. Video Google: A Text Retrieval Approach to Object Matching in Videos. *Conference on Computer Vision and Pattern Recognition*, 2:1470–1470, oct 2003.

[171] Stephen M Smith and J Michael Brady. SUSAN—A New Approach to Low Level Image Processing. *International Journal of Computer Vision*, 23(1):45–78, 1997.

[172] Jaime Spencer, Richard Bowden, and Simon Hadfield. Scale-adaptive neural dense features: Learning via hierarchical context aggregation. In *Conference on Computer Vision and Pattern Recognition*, pages 6193–6202. IEEE Computer Society, jun 2019.

[173] Jaime Spencer, Richard Bowden, and Simon Hadfield. DeFeat-Net: General monocular depth via simultaneous unsupervised representation learning. In *Conference on Computer Vision and Pattern Recognition*, pages 14390–14401. IEEE Computer Society, jun 2020.

[174] Jaime Spencer, Richard Bowden, and Simon Hadfield. Same features, different day: Weakly supervised feature learning for seasonal invariance. In *Conference on Computer Vision and Pattern Recognition*, pages 6458–6467. IEEE Computer Society, jun 2020.

[175] Christoph Strecha, Alexander M. Bronstein, Michael M. Bronstein, and Pascal Fua. LDAHash: Improved matching with smaller descriptors. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 34(1):66–78, jan 2012.

[176] Ke Sun, Bin Xiao, Dong Liu, and Jingdong Wang. Deep high-resolution representation learning for human pose estimation. In *Conference on Computer Vision and Pattern Recognition*, volume 2019-June, pages 5686–5696. IEEE Computer Society, jun 2019.

[177] Mingxing Tan and Quoc Le. EfficientNet: Rethinking Model Scaling for Convolutional Neural Networks. In *International Conference on Machine Learning*, pages 6105–6114. PMLR, may 2019.

[178] Andrew Tao, Karan Sapra, and Bryan Catanzaro. Hierarchical Multi-Scale Attention for Semantic Segmentation. *arXiv preprint*, may 2020.

[179] Yurun Tian, Vassileios Balntas, Tony Ng, Axel Barroso-Laguna, Yiannis Demiris, and Krystian Mikolajczyk. D2D: Keypoint Extraction with Describe to Detect Approach. In *Asian Conference on Computer Vision*, pages 223–240. Springer International Publishing, 2020.

[180] Yurun Tian, Axel Barroso Laguna, Tony Ng, Vassileios Balntas, and Krystian Mikolajczyk. HyNet: Learning Local Descriptor with Hybrid Similarity Measure and Triplet Loss. In *Advances in Neural Information Processing Systems*, volume 33, pages 7401–7412. Curran Associates, Inc., 2020.

[181] Yurun Tian, Bin Fan, and Fuchao Wu. L2-Net: Deep learning of discriminative patch descriptor in Euclidean space. In *Conference on Computer Vision and Pattern Recognition*, pages 6128–6136. Institute of Electrical and Electronics Engineers Inc., nov 2017.

[182] Yurun Tian, Xin Yu, Bin Fan, Fuchao Wu, Huub Heijnen, and Vassileios Balntas. SOSNet: Second order similarity regularization for local descriptor learning. In *Conference on Computer Vision and Pattern Recognition*, volume 2019-June, pages 11008–11017. IEEE Computer Society, jun 2019.

[183] Engin Tola, Vincent Lepetit, and Pascal Fua. A fast local descriptor for dense matching. In *Conference on Computer Vision and Pattern Recognition*, pages 1–8. IEEE Computer Society, jun 2008.

[184] Engin Tola, Vincent Lepetit, and Pascal Fua. DAISY: An efficient dense descriptor applied to wide-baseline stereo. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 32(5):815–830, may 2010.

[185] T Tuytelaars and L van Gool. Wide Baseline Stereo Matching based on Local, Affinely Invariant Regions. In *British Machine Vision Conference*, pages 38.1–38.14. BMVA Press, 2000.

[186] Michał Tyszkiewicz, Pascal Fua, and Eduard Trulls. DISK: Learning local features with policy gradient. In *Advances in Neural Information Processing Systems*, volume 33, pages 14254–14265. Curran Associates, Inc., 2020.

[187] Benjamin Ummenhofer, Huizhong Zhou, Jonas Uhrig, Nikolaus Mayer, Eddy Ilg, Alexey Dosovitskiy, and Thomas Brox. DeMoN: Depth and Motion Network for Learning Monocular Stereo. *Conference on Computer Vision and Pattern Recognition*, pages 5622–5631, jul 2017.

[188] Aaron van den Oord, Yazhe Li, and Oriol Vinyals. Representation Learning with Contrastive Predictive Coding. *arXiv preprint*, jul 2018.

[189] Simon Vandenhende, Stamatios Georgoulis, Bert De Brabandere, and Luc Van Gool. Branched Multi-Task Networks: Deciding What Layers To Share. *British Machine Vision Conference*, apr 2020.

[190] Simon Vandenhende, Stamatios Georgoulis, Wouter Van Gansbeke, Marc Proesmans, Dengxin Dai, and Luc Van Gool. Multi-Task Learning for Dense Prediction Tasks: A Survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, apr 2021.

[191] Simon Vandenhende, Stamatios Georgoulis, and Luc Van Gool. MTI-Net: Multi-scale Task Interaction Networks for Multi-task Learning. In *European Conference on Computer Vision*, pages 527–543. Springer International Publishing, aug 2020.

[192] Yannick Verdie, Kwang Moo Yi, Pascal Fua, and Vincent Lepetit. TILDE: A Temporally Invariant Learned DEtector. In *Conference on Computer Vision and Pattern Recognition*, pages 5279–5288. IEEE Computer Society, oct 2015.

[193] Paul Viola and Michael Jones. Rapid object detection using a boosted cascade of simple features. In *Conference on Computer Vision and Pattern Recognition*, volume 1, pages 511–511. IEEE Computer Society, dec 2001.

[194] Qianqian Wang, Xiaowei Zhou, Bharath Hariharan, and Noah Snavely. Learning Feature Descriptors Using Camera Pose Supervision. In *European Conference on Computer Vision*, pages 757–774, Cham, 2020. Springer International Publishing.

[195] Tobias Weyand, André Araujo, Bingyi Cao, and Jack Sim. Google landmarks dataset v2 A large-scale benchmark for instance-level recognition and retrieval. *Conference on Computer Vision and Pattern Recognition*, pages 2572–2581, 2020.

[196] Alex Wong and Stefano Soatto. Bilateral Cyclic Constraint and Adaptive Regularization for Unsupervised Monocular Depth Prediction. *Conference on Computer Vision and Pattern Recognition*, pages 5637–5646, jun 2019.

[197] Zhirong Wu, Yuanjun Xiong, Stella X. Yu, and Dahua Lin. Unsupervised Feature Learning via Non-parametric Instance Discrimination. In *Conference on Computer Vision and Pattern Recognition*, pages 3733–3742. IEEE Computer Society, dec 2018.

[198] Junyuan Xie, Ross Girshick, and Ali Farhadi. Deep3D: Fully Automatic 2D-to-3D Video Conversion with Deep Convolutional Neural Networks. *European Conference on Computer Vision*, pages 842–857, 2016.

[199] Dan Xu, Wanli Ouyang, Xiaogang Wang, and Nicu Sebe. PAD-Net: Multi-tasks Guided Prediction-and-Distillation Network for Simultaneous Depth Estimation and Scene Parsing. In *Conference on Computer Vision and Pattern Recognition*, pages 675–684. IEEE Computer Society, dec 2018.

[200] Zhi Yan, Li Sun, Tomas Krajnik, and Yassine Ruichek. EU long-term dataset with multiple sensors for autonomous driving. *International Conference on Intelligent Robots and Systems*, pages 10697–10704, oct 2020.

[201] Tsun-Yi Yang, Duy-Kien Nguyen, Huub Heijnen, and Vassileios Balntas. UR2KiD: Unifying Retrieval, Keypoint Detection, and Keypoint Description without Local Correspondence Supervision. *arXiv preprint*, jan 2020.

[202] Xin Yang and Kwang Ting Cheng. LDB: An ultra-fast feature for scalable Augmented Reality on mobile devices. In *International Symposium on Mixed and Augmented Reality*, pages 49–57. IEEE Computer Society, nov 2012.

[203] Zhenheng Yang, Peng Wang, Yang Wang, Wei Xu, and Ram Nevatia. LEGO: Learning Edge with Geometry all at Once by Watching Videos. *Conference on Computer Vision and Pattern Recognition*, pages 225–234, jun 2018.

[204] Kwang Moo Yi, Eduard Trulls, Vincent Lepetit, and Pascal Fua. LIFT: Learned Invariant Feature Transform. In Bastian Leibe, Jiri Matas, Nicu Sebe, and Max Welling, edi-

tors, *European Conference on Computer Vision*, pages 467–483, Cham, 2016. Springer International Publishing.

[205] Kwang Moo Yi, Yannick Verdie, Pascal Fua, and Vincent Lepetit. Learning to assign orientations to feature points. In *Conference on Computer Vision and Pattern Recognition*, pages 107–116. IEEE Computer Society, dec 2016.

[206] Guoshen Yu and Jean-Michel Morel. ASIFT: An Algorithm for Fully Affine Invariant Comparison. *Image Processing On Line*, 1:11–38, feb 2011.

[207] Sergey Zagoruyko and Nikos Komodakis. Learning to compare image patches via convolutional neural networks. In *Conference on Computer Vision and Pattern Recognition*, pages 4353–4361. IEEE Computer Society, oct 2015.

[208] Jurě Zbontar and Yann Lecun. Stereo Matching by Training a Convolutional Neural Network to Compare Image Patches. *Journal of Machine Learning Research*, 17:1–32, 2016.

[209] Huangying Zhan, Ravi Garg, Chamara Saroj Weerasekera, Kejie Li, Harsh Agarwal, and Ian M. Reid. Unsupervised Learning of Monocular Depth Estimation and Visual Odometry with Deep Feature Reconstruction. *Conference on Computer Vision and Pattern Recognition*, pages 340–349, jun 2018.

[210] Dingyi Zhang, Yingming Li, and Zhongfei Zhang. Deep Metric Learning with Spherical Embedding. In *Advances in Neural Information Processing Systems*, volume 33, pages 18772–18783. Curran Associates, Inc., 2020.

[211] Xu Zhang, Felix X. Yu, Sanjiv Kumar, and Shih Fu Chang. Learning Spread-Out Local Feature Descriptors. In *International Conference on Computer Vision*, pages 4605–4613. IEEE Computer Society, dec 2017.

[212] Zhenyu Zhang, Zhen Cui, Chunyan Xu, Zequn Jie, Xiang Li, and Jian Yang. Joint Task-Recursive Learning for Semantic Segmentation and Depth Estimation. In *European Conference on Computer Vision*, pages 235–251. Springer International Publishing, 2018.

[213] Zhenyu Zhang, Zhen Cui, Chunyan Xu, Yan Yan, Nicu Sebe, and Jian Yang. Pattern-affinitive propagation across depth, surface normal and semantic segmentation. In *Con-*

ference on Computer Vision and Pattern Recognition, pages 4101–4110. IEEE Computer Society, jun 2019.

[214] Xiangyun Zhao, Haoxiang Li, Xiaohui Shen, Xiaodan Liang, and Ying Wu. A Modulation Module for Multi-task Learning with Applications in Image Retrieval. In *European Conference on Computer Vision*, volume 11205 LNCS, pages 415–432. Springer International Publishing, sep 2018.

[215] Yong Zhao, Shibiao Xu, Shuhui Bu, Hongkai Jiang, and Pengcheng Han. GSLAM: A general SLAM framework and benchmark. *International Conference on Computer Vision*, pages 1110–1120, oct 2019.

[216] Tinghui Zhou, Matthew Brown, Noah Snavely, and David G. Lowe. Unsupervised Learning of Depth and Ego-Motion from Video. *Conference on Computer Vision and Pattern Recognition*, pages 6612–6619, jul 2017.

[217] Laurent Zwald and Sophie Lambert-Lacroix. The BerHu penalty and the grouped effect. *arXiv preprint*, jul 2012.