Universidade Federal do Espírito Santo - UFES
Laboratório de Computação de Alto Desempenho - LCAD

Discipline: DEEP LEARNING – 2017.2

Professor: ALBERTO FERREIRA DE SOUZA

Mapping Road Lanes Using Laser Remission and Deep Neural Networks

R. V. Carneiro, R. C. Nascimento, R. Guidolini, V. Cardoso,

T. Oliveira-Santos, C. Badue and A. F. De Souza

Submitted to IEEE IJCNN 2018

LABORATÓRIO DE COMPUTAÇÃO DE ALTO DESEMPENHO

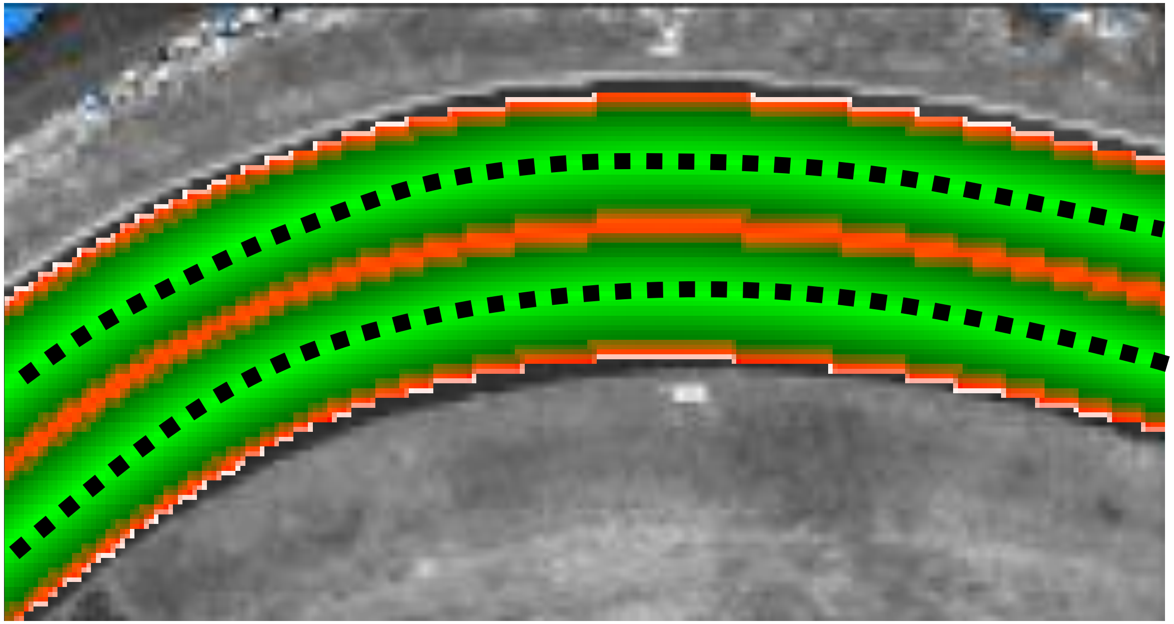
Presented by: RAPHAEL VIVACQUA CARNEIRO

Summary

1. Introduction
2. A DNN for Semantic Segmentation of Road Maps from Remission Grid Maps
3. The Use of Road Grid Maps in Autonomous Cars
    A. The IARA Software System
    B. Computing RDDFs from Road Grid Maps
4. Methodology
    A. Datasets
    B. ENet DNN
5. Experiments
6. Conclusion

Introduction

- Autonomous cars must stay within a road lane and therefore they must have an internal map representing the position and properties of the lanes.



Problem:

- Automatic mechanism for inferring road lanes
- Automatic extraction of waypoints from road lanes

Introduction

Previous works:

- Previous works detect relevant horizontal signalization for building Advanced Driver-Assistance Systems (ADAS) or relevant parts of autonomous cars systems.
- However, sometimes the horizontal signalization is not in good conditions or even absent.
- Previous works either do not generate a road map; and/or do not find the lanes on the road; and/or were not tested on a real autonomous car.

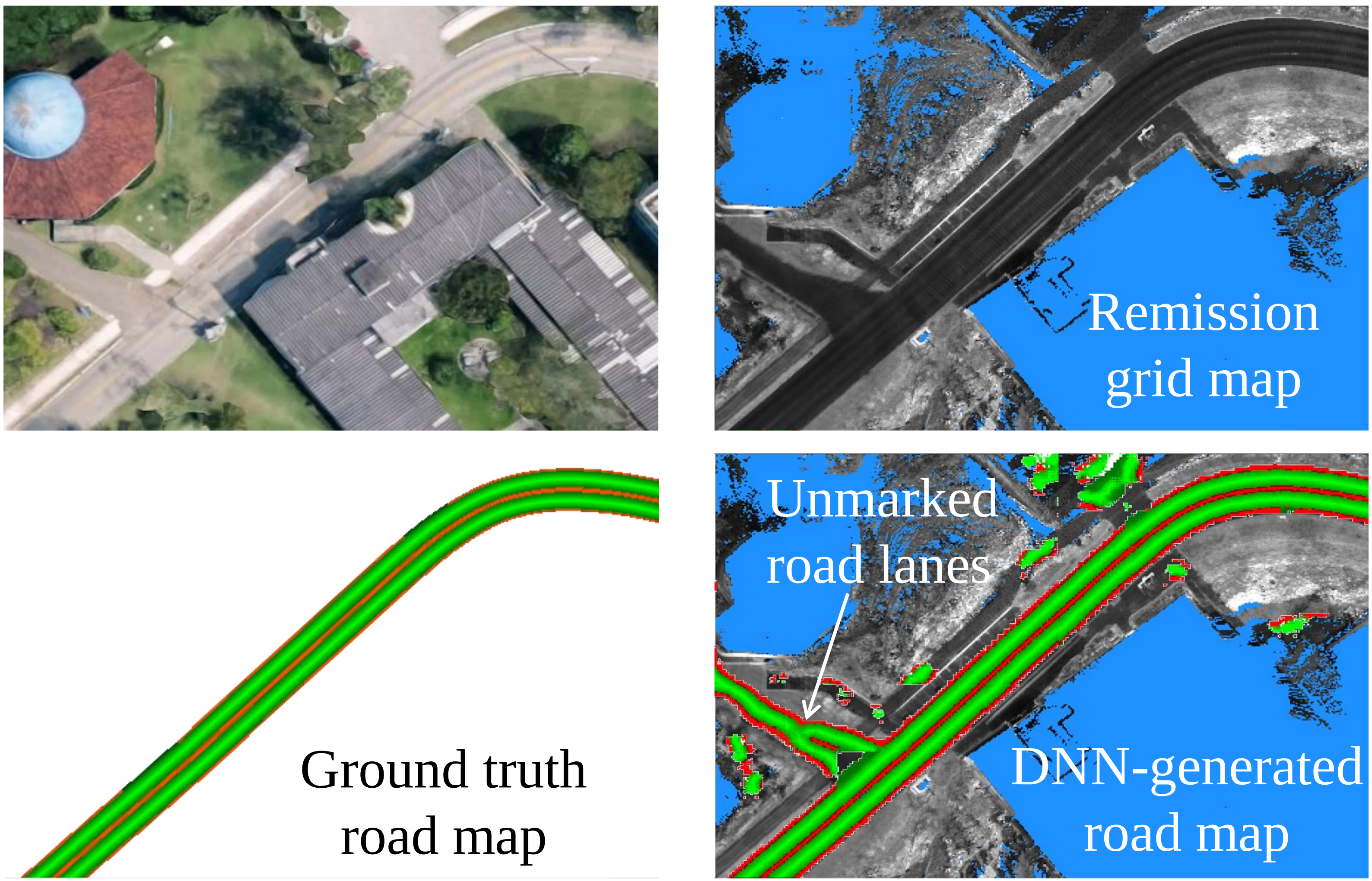
Introduction

Our approach:

- We propose the use of a deep neural network (DNN) for solving the problem.
- We use the Efficient Neural Network (ENet) for making semantic segmentation of LiDAR remission grid maps into road maps.
- To train the DNN, we built a dataset of tens of kilometers of manually marked road lanes.
- After being trained, the DNN achieved an average segmentation accuracy of 83.7% and it was capable of finding existing lanes not marked in the ground truth.

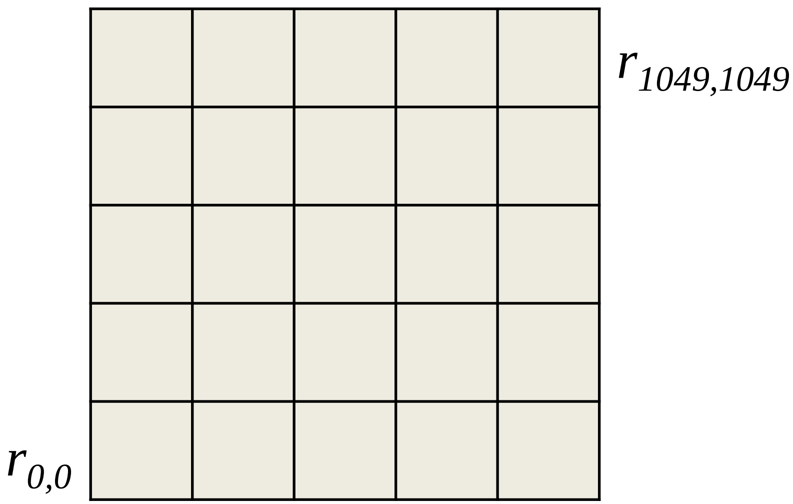
Introduction

Our approach:



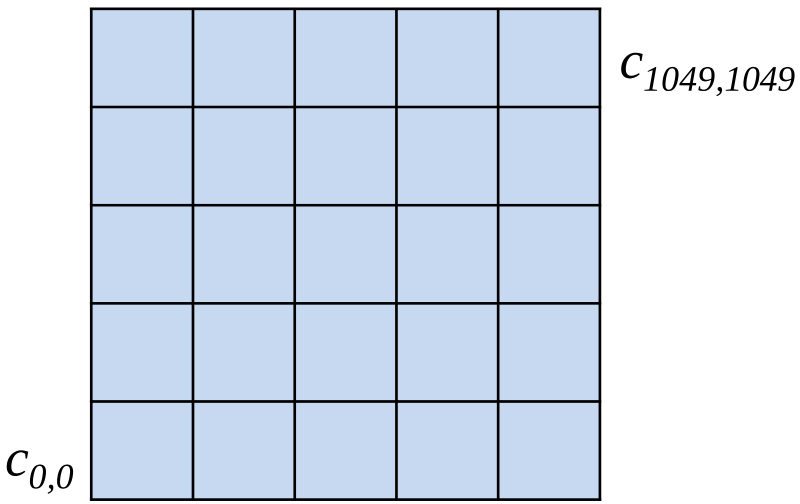
A DNN for Semantic Segmentation of Road Maps from Remission Grid Maps

Remission Grid Map



$R = \{r_{0,0}, \ldots, r_{i,j}, \ldots, r_{s-1,s-1}\}$

Road Grid Map



$C = \{c_{0,0}, \ldots, c_{i,j}, \ldots, c_{s-1,s-1}\}$

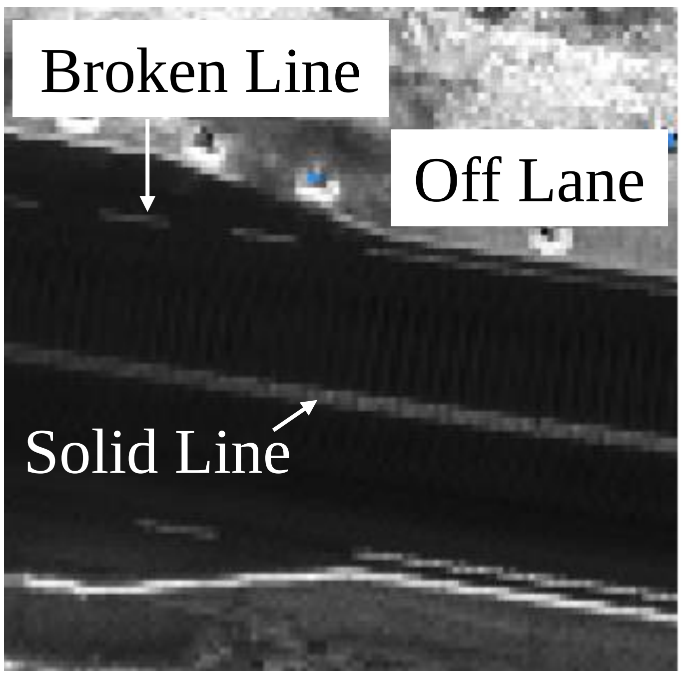
$s = 1050$ cells

Each cell represents an area of 20 cm × 20 cm

A grid map covers a total area of 210 m × 210 m

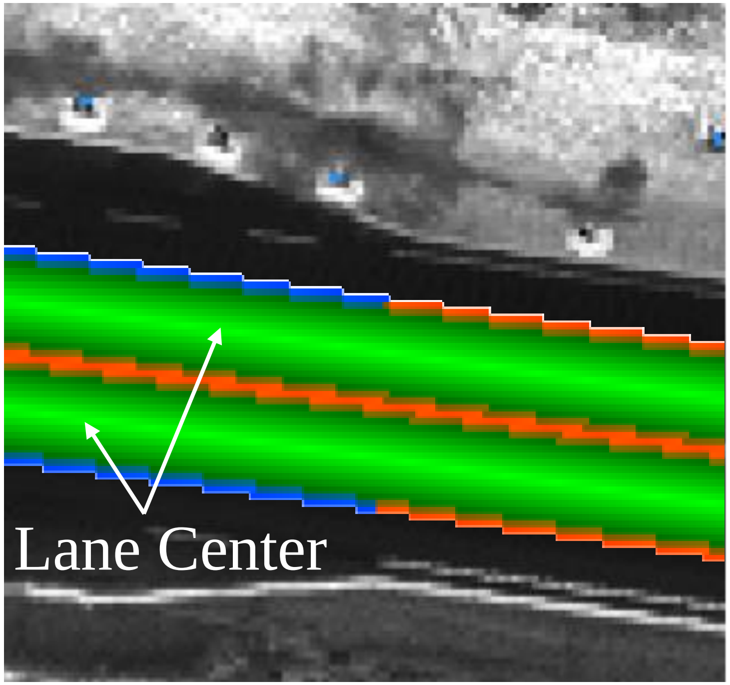
A DNN for Semantic Segmentation of Road Maps from Remission Grid Maps

Remission Grid Map



The value of $r_{i,j}$ is the average LiDAR remission, i.e. intensity of the returned light pulse

Road Grid Map



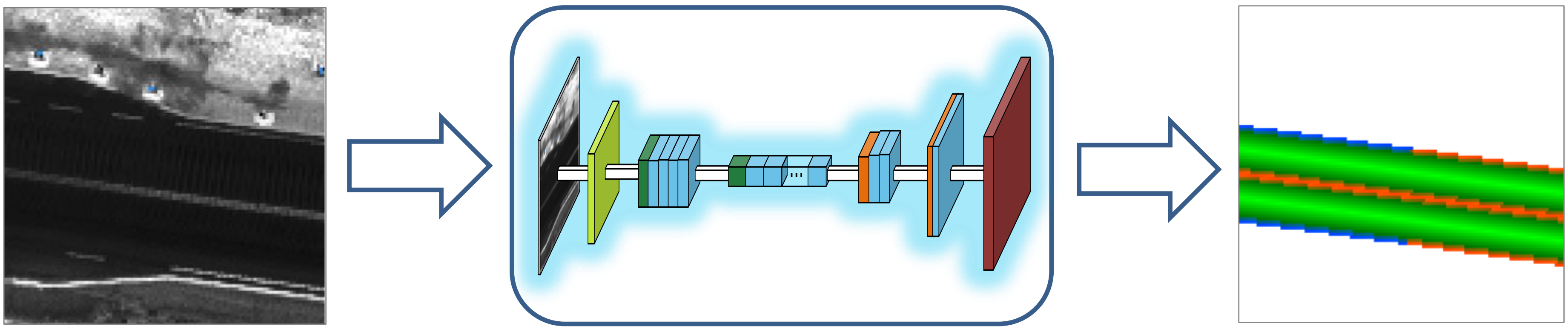
Value of $c_{i,j}$:
0: Off Lane
1: Solid Line
2: Broken Line
3: Solid (50%)
4: Broken (50%)
5: Lane Center
6: Center↔w/22
7: Center↔2w/22
…
16: Center↔11w/22

w = width of 3.2 m

The value of $c_{i,j}$ represents different classes of features

A DNN for Semantic Segmentation of Road Maps from Remission Grid Maps

Remission Grid Map ENet DNN Road Grid Map



During the test phase, the ENet DNN receives as input a crop of the remission grid map and outputs the inferred road grid map in the form of a matrix of softmax classifiers: the class most active of each softmax classifier will be the value inferred for the corresponding $c_{i,j}$ of the road grid map.

A DNN for Semantic Segmentation of Road Maps from Remission Grid Maps

ENet – Efficient Neural Network:

- Semantic segmentation of road grid maps from remission grid maps.
- Similar to SegNet, with several modifications and faster operation, without significant performance degradation
- Training in two stages:
  1. The encoder and a deconvolution layer
  2. The full network (encoder and decoder)
- The final output is a matrix of SoftMax classifiers with the same size of the input (120 × 120 cells).

A. Paszke, A. Chaurasia, S. Kim, and E. Culurciello, "ENet: A deep neural network architecture for real-time semantic segmentation," arXiv preprint arXiv:1606.02147, 2016. (WARSAW / PURDUE)

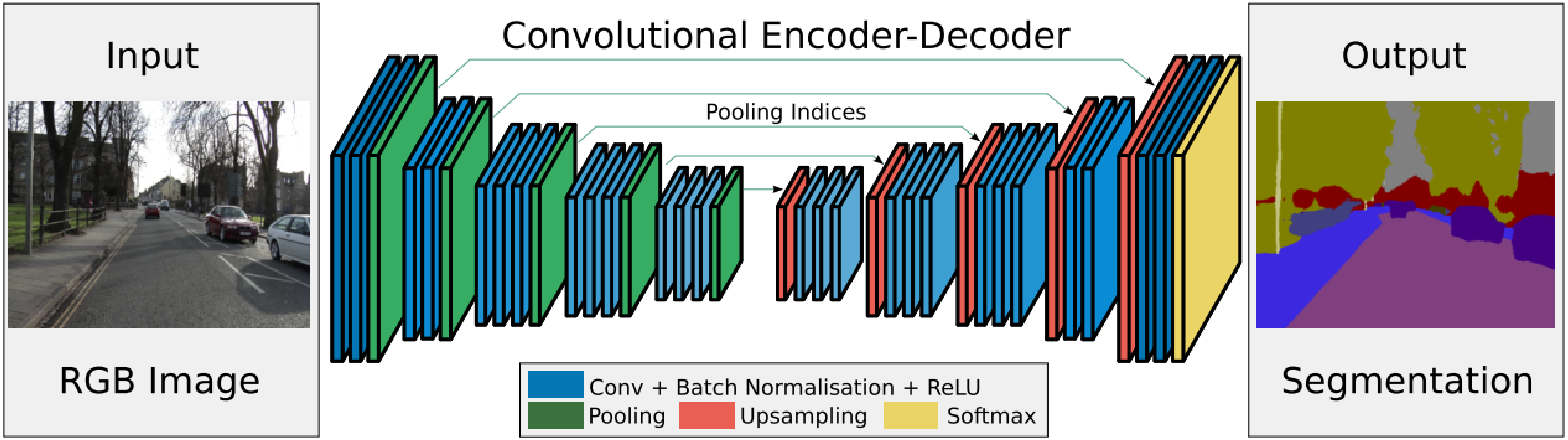
A DNN for Semantic Segmentation of Road Maps from Remission Grid Maps

SegNet – Architecture:

Symmetric Encoder-Decoder architecture:

Encoder: 13 convolutional layers, corresponding to the first 13 layers in VGG16

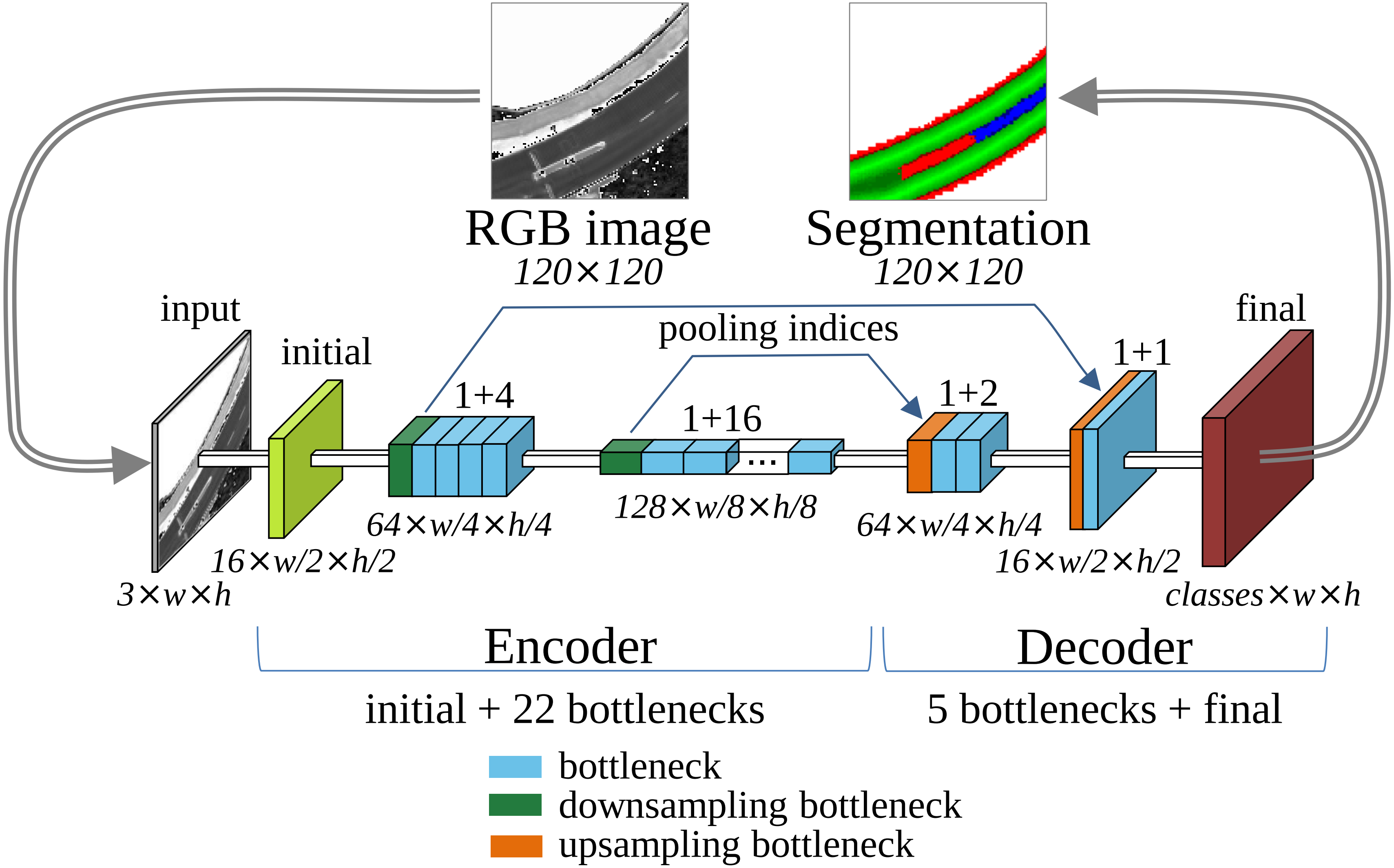
Decoder: 13 convolutional layers



V. Badrinarayanan, A. Handa, and R. Cipolla, “SegNet: A deep convolutional encoder-decoder architecture for robust semantic pixel-wise labelling,” arXiv preprint arXiv:1505.07293, 2015. (CAMBRIDGE)

A DNN for Semantic Segmentation of Road Maps from Remission Grid Maps

ENet – Architecture:



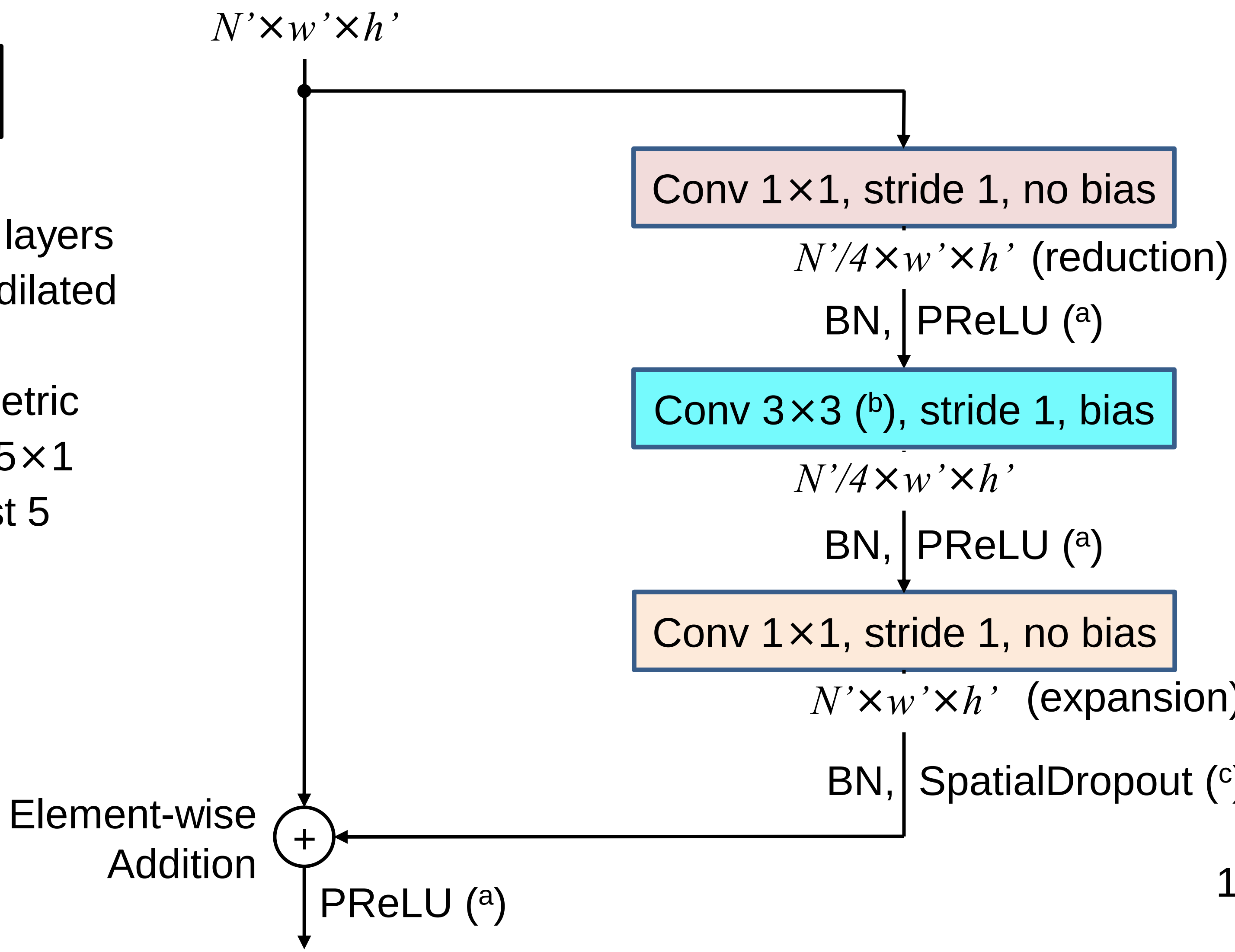
ENet – Architecture:

Bottleneck

([a]) ReLU in Decoder layers

([b]) Some layers use dilated convolutions; some use asymmetric kernels 1×5 and 5×1

([c]) p = 0.01 in the first 5 bottlenecks; otherwise p = 0.1

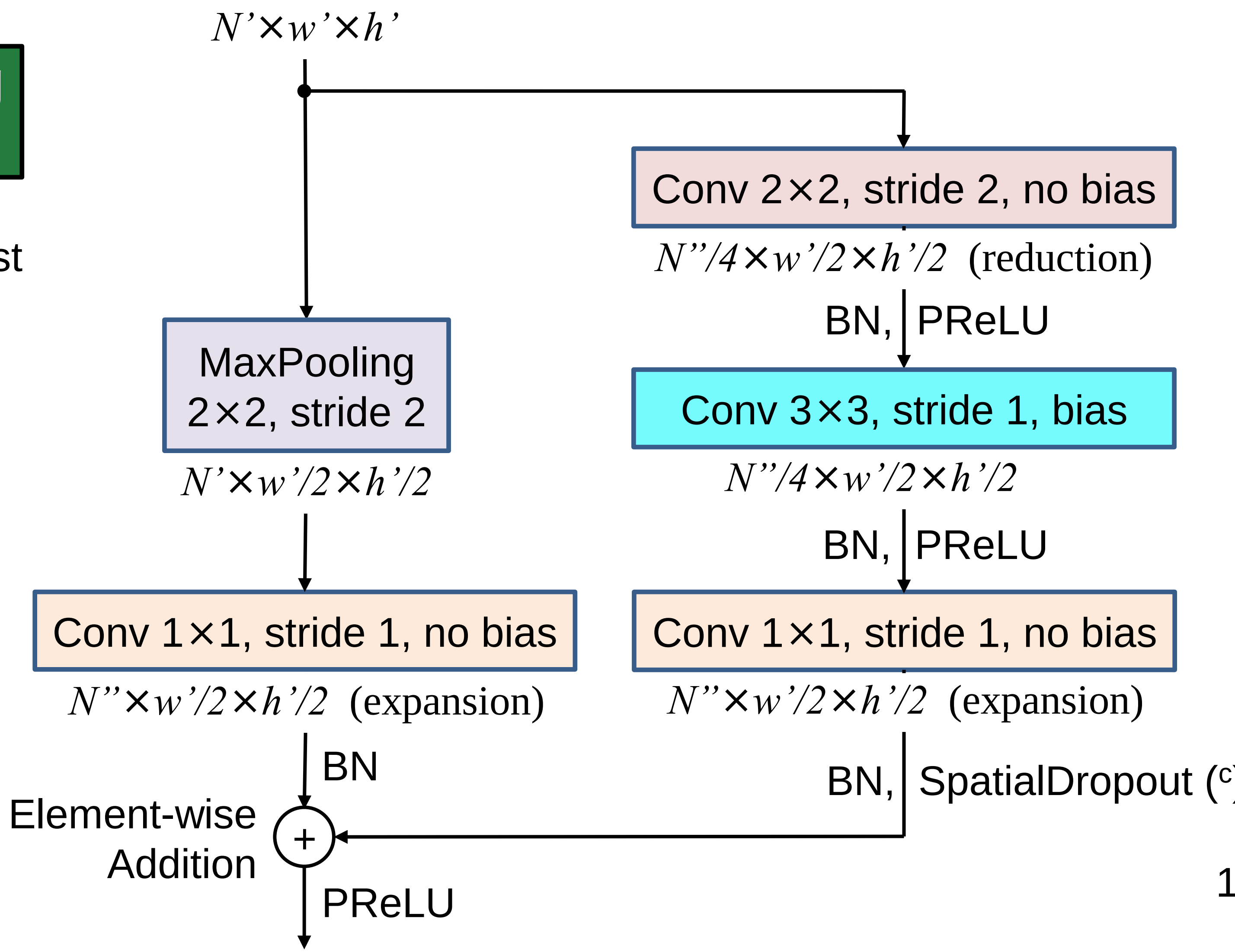


A DNN for Semantic Segmentation of Road Maps from Remission Grid Maps

ENet – Architecture:

Downsampling Bottleneck

(c) p = 0.01 in the first 5 bottlenecks; otherwise p = 0.1

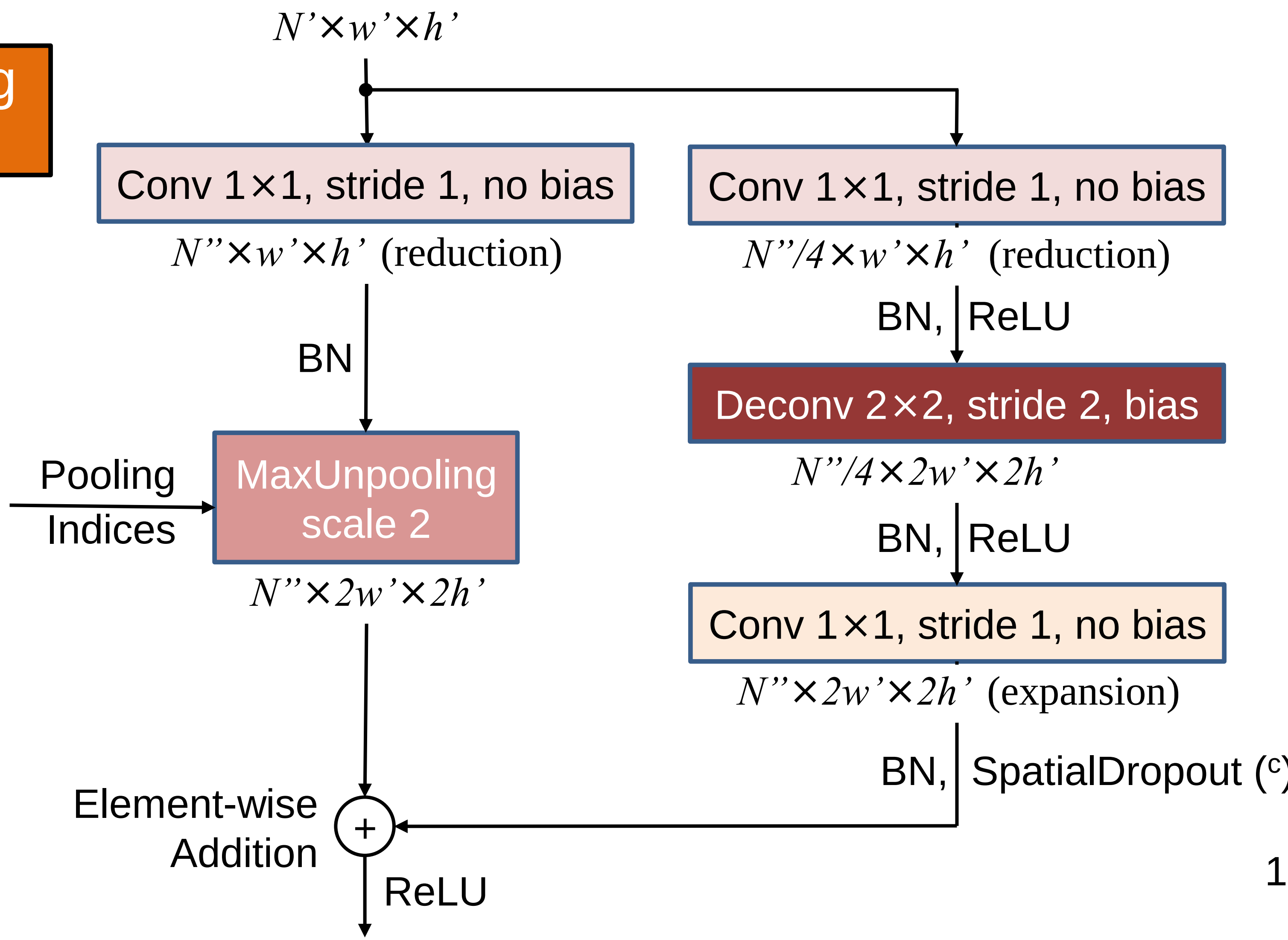


A DNN for Semantic Segmentation of Road Maps from Remission Grid Maps

ENet – Architecture:

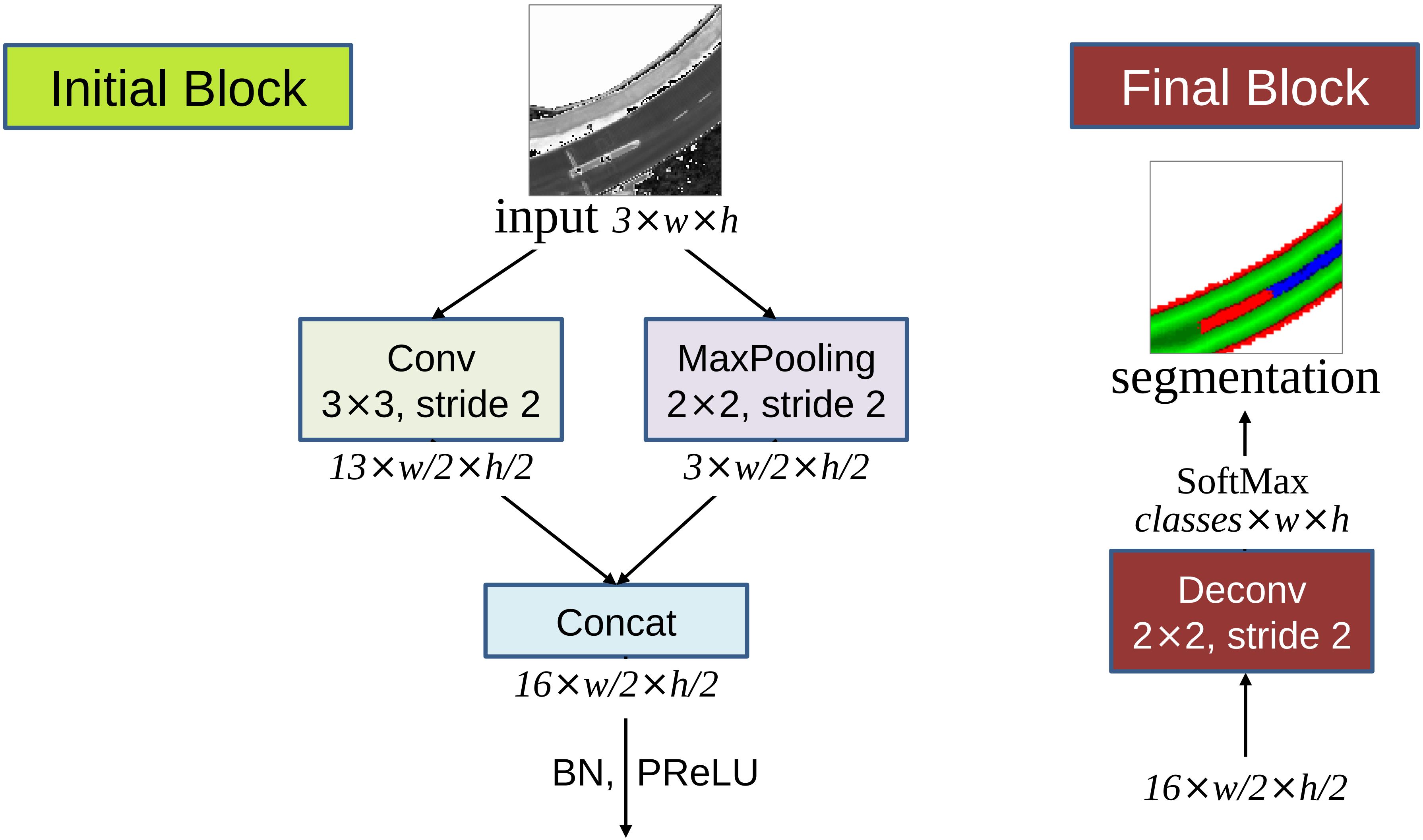
Upsampling Bottleneck

(c) p = 0.1



A DNN for Semantic Segmentation of Road Maps from Remission Grid Maps

ENet – Architecture:



ENet – Architecture:

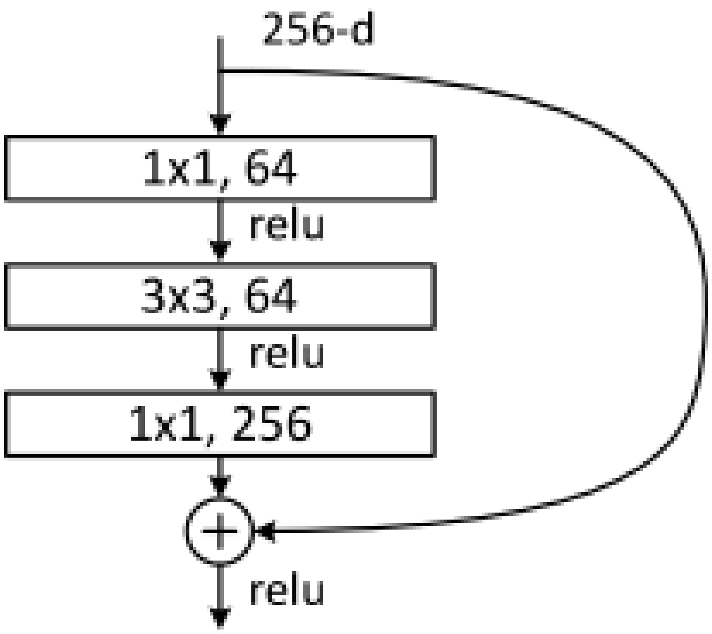
Example:
$w = 512$
$h = 512$

Name	Type	Output size
initial		$16 \times 256 \times 256$
bottleneck1.0	downsampling	$64 \times 128 \times 128$
$4\times$ bottleneck1.x		$64 \times 128 \times 128$
bottleneck2.0	downsampling	$128 \times 64 \times 64$
bottleneck2.1		$128 \times 64 \times 64$
bottleneck2.2	dilated 2	$128 \times 64 \times 64$
bottleneck2.3	asymmetric 5	$128 \times 64 \times 64$
bottleneck2.4	dilated 4	$128 \times 64 \times 64$
bottleneck2.5		$128 \times 64 \times 64$
bottleneck2.6	dilated 8	$128 \times 64 \times 64$
bottleneck2.7	asymmetric 5	$128 \times 64 \times 64$
bottleneck2.8	dilated 16	$128 \times 64 \times 64$
Repeat section 2, without bottleneck2.0		
bottleneck4.0	upsampling	$64 \times 128 \times 128$
bottleneck4.1		$64 \times 128 \times 128$
bottleneck4.2		$64 \times 128 \times 128$
bottleneck5.0	upsampling	$16 \times 256 \times 256$
bottleneck5.1		$16 \times 256 \times 256$
fullconv		$C \times 512 \times 512$

ENet – Design Choices:

Deep Bottleneck Architecture

- ResNet (Residual Network)
- Avoid degradation problem when the depth increases



K. He, X. Zhang, S. Ren, and J. Sun, “Deep residual learning for image recognition,” arXiv preprint arXiv:1512.03385, 2015. (MICROSOFT)

ENet – Design Choices:

Batch Normalization

- Make normalization a part of the model architecture.
- Perform the normalization for each training mini-batch.
- Avoid *internal covariate shift* (the distribution of each layer's inputs changes during training, as the parameters of the previous layers change).
- Allow to use much higher learning rates and be less careful about initialization.
- SegNet uses Batch Normalization

S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," arXiv preprint arXiv:1502.03167, 2015. (GOOGLE)

ENet – Design Choices:

Batch Normalization

- No need to perform normalization of the entire dataset before performing batch normalizations.

$$N_T(x) = \frac{x - \mu_T}{\sigma_T} \qquad N_B(x) = \frac{x - \mu_B}{\sigma_B} = \frac{x - \frac{1}{B}\sum x_i}{\sqrt{\frac{1}{B}\sum\left(x_i - \frac{1}{B}\sum x_i\right)^2}}$$

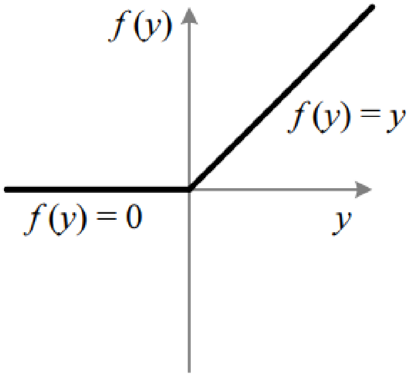
$$N_B(N_T(x)) = \frac{\frac{x - \mu_T}{\sigma_T} - \frac{1}{B}\sum\frac{x_i - \mu_T}{\sigma_T}}{\sqrt{\frac{1}{B}\sum\left(\frac{x_i - \mu_T}{\sigma_T} - \frac{1}{B}\sum\frac{x_i - \mu_T}{\sigma_T}\right)^2}} = \frac{x - \frac{1}{B}\sum x_i}{\sqrt{\frac{1}{B}\sum\left(x_i - \frac{1}{B}\sum x_i\right)^2}} = N_B\,(x)$$

A DNN for Semantic Segmentation of Road Maps from Remission Grid Maps

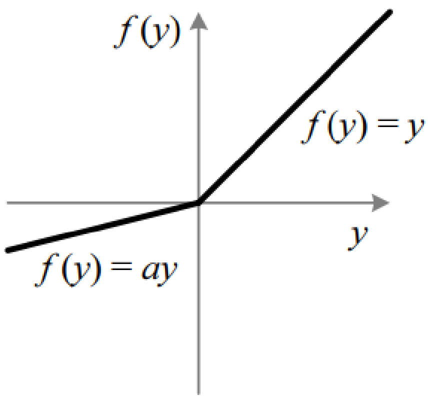
ENet – Design Choices:

Nonlinear Operations

- Parametric Rectified Linear Unit
- PReLU generalizes ReLU
- Improves model fitting with nearly zero extra computational cost and little overfitting risk.



ReLU



PReLU

K. He, X. Zhang, S. Ren, and J. Sun, “Delving deep into rectifiers: Surpassing human-level performance on imagenet classification,” pp. 1026–1034, 2015. (MICROSOFT)

ENet – Design Choices:

Bias Terms

- ENet does not use bias terms in any of the projections (reduction and expansion conv layers), in order to reduce the number of kernel calls and overall memory operations, as cuDNN uses separate kernels for convolution and bias addition.
- This choice did not cause any impact on the accuracy.

S. Chetlur, C. Woolley, P. Vandermersch, J. Cohen, J. Tran, B. Catanzaro, and E. Shelhamer, “cuDNN: Efficient primitives for deep learning,” arXiv preprint arXiv:1410.0759, 2014. (NVIDIA)

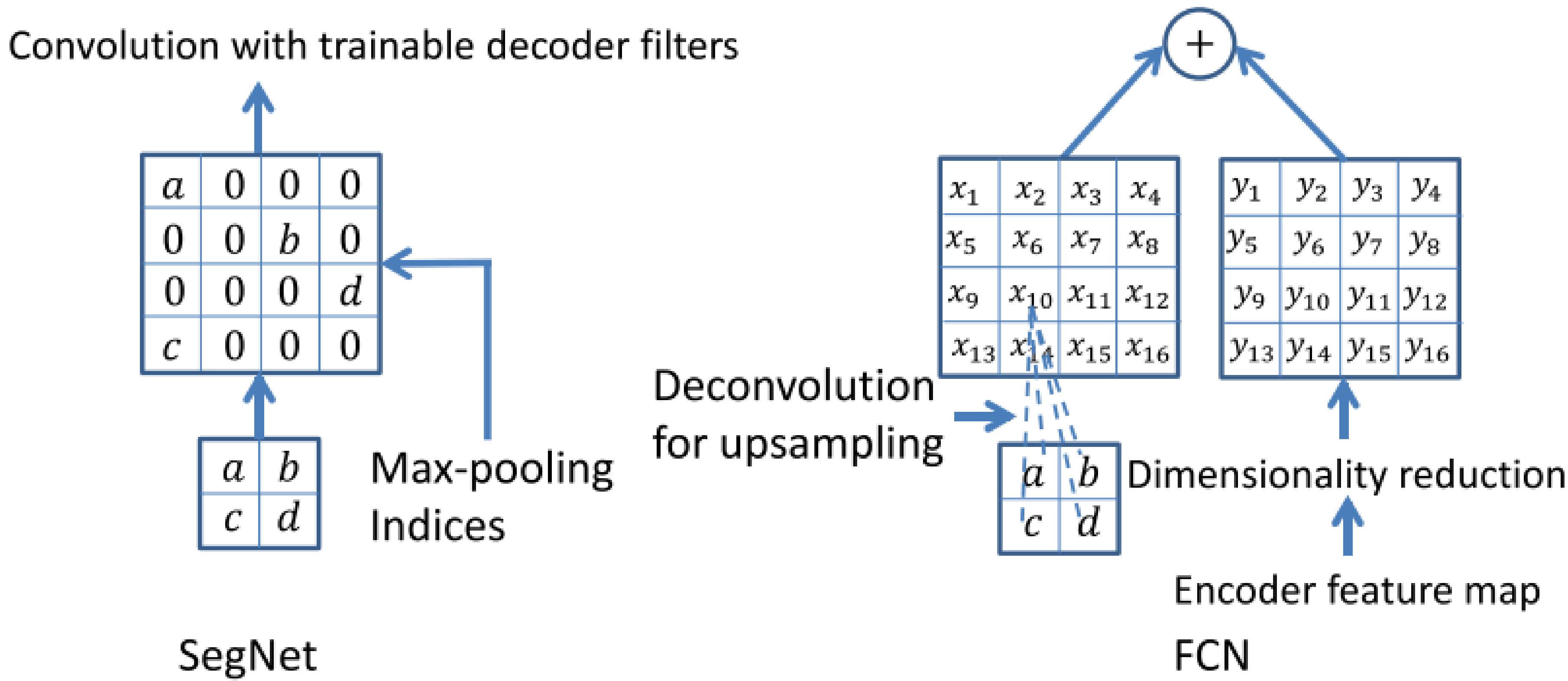
ENet – Design Choices:

SegNet Decoding Technique

- Fully Convolutional Networks (**FCN**) upsample by learning to deconvolve the input feature map and adds the corresponding encoder feature map to produce the decoder output.
- **SegNet** uses the max pooling indices to upsample (without learning) the feature map(s) and convolves with a trainable decoder filter bank. This approach reduces memory requirements.

ENet – Design Choices:

SegNet Decoding Technique



ENet – Design Choices:

SegNet Decoder Size

- SegNet is a very symmetric architecture, as the encoder is an exact mirror of the encoder.
- Instead, ENet architecture consists of a large encoder, and a small decoder.
- This is motivated by the idea that the encoder should be able to provide information processing and filtering. Instead, the role of the decoder, is to upsample the output of the encoder, only fine-tuning the details.

ENet – Design Choices:

SpatialDropout Regularization

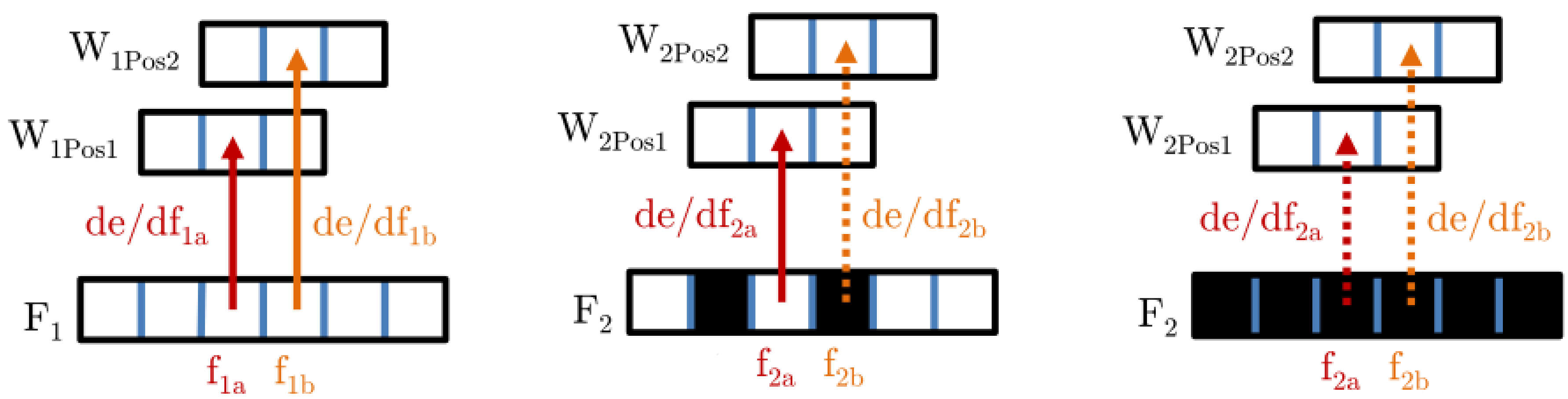
- Natural images exhibit strong spatial correlation, so in convolutional networks the feature map activations are also strongly correlated, and in this setting standard dropout fails.

J. Tompson, R. Goroshin, A. Jain, Y. LeCun, and C. Bregler, “Efficient object localization using convolutional networks,” in Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2015, pp. 648–656. (NYU)

A DNN for Semantic Segmentation of Road Maps from Remission Grid Maps

ENet – Design Choices:

SpatialDropout



Standard Dropout | SpatialDropout

F_1, F_2 = 1D feature maps

W_1, W_2 = 1D conv kernels

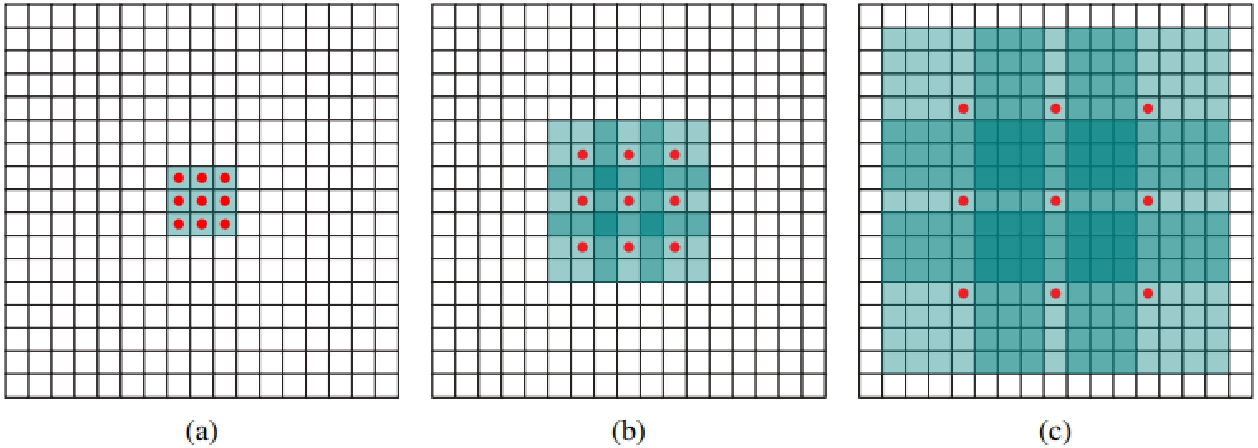
de/df = gradient contributions

■ = dropped-out

ENet – Design Choices:

Dilated Convolutions

- Increase the receptive field exponentially while the number of parameters grows linearly.



a) Kernel 3×3, receptive field 3×3
b) Kernel 3×3, dilation 2, receptive field 7×7
c) Kernel 3×3, dilation 4, receptive field 15×15

F. Yu and V. Koltun, “Multi-scale context aggregation by dilated convolutions,” arXiv preprint arXiv:1511.07122, 2015. (PRINCETON / INTEL)

A DNN for Semantic Segmentation of Road Maps from Remission Grid Maps

ENet – Design Choices:

Asymmetric Convolutions (or Flattened Convolutions)

- Consecutive sequence of one-dimensional kernels across all directions in space.
- Avoid redundancy of the parameters, especially weights of the convolutional symmetric kernels.
- Around two times speed-up during feedforward pass.
- Comparable performance as conventional convolutions.

J. Jin, A. Dundar, and E. Culurciello, “Flattened convolutional neural networks for feedforward acceleration,” arXiv preprint arXiv:1412.5474, 2014. (PURDUE)

The Use of Road Grid Maps in Autonomous Cars

Road grid maps have several relevant applications for autonomous cars:

- Get the distance from the car to the center of the nearest lane.
- Make decision of overtaking another car through a specific side of the lane (broken line marking).
- Build a graph of the mapped road network.
- Build RDDFs (Road Definition Data Files)

The Use of Road Grid Maps in Autonomous Cars

RDDF format in the IARA Software System:

Waypoints

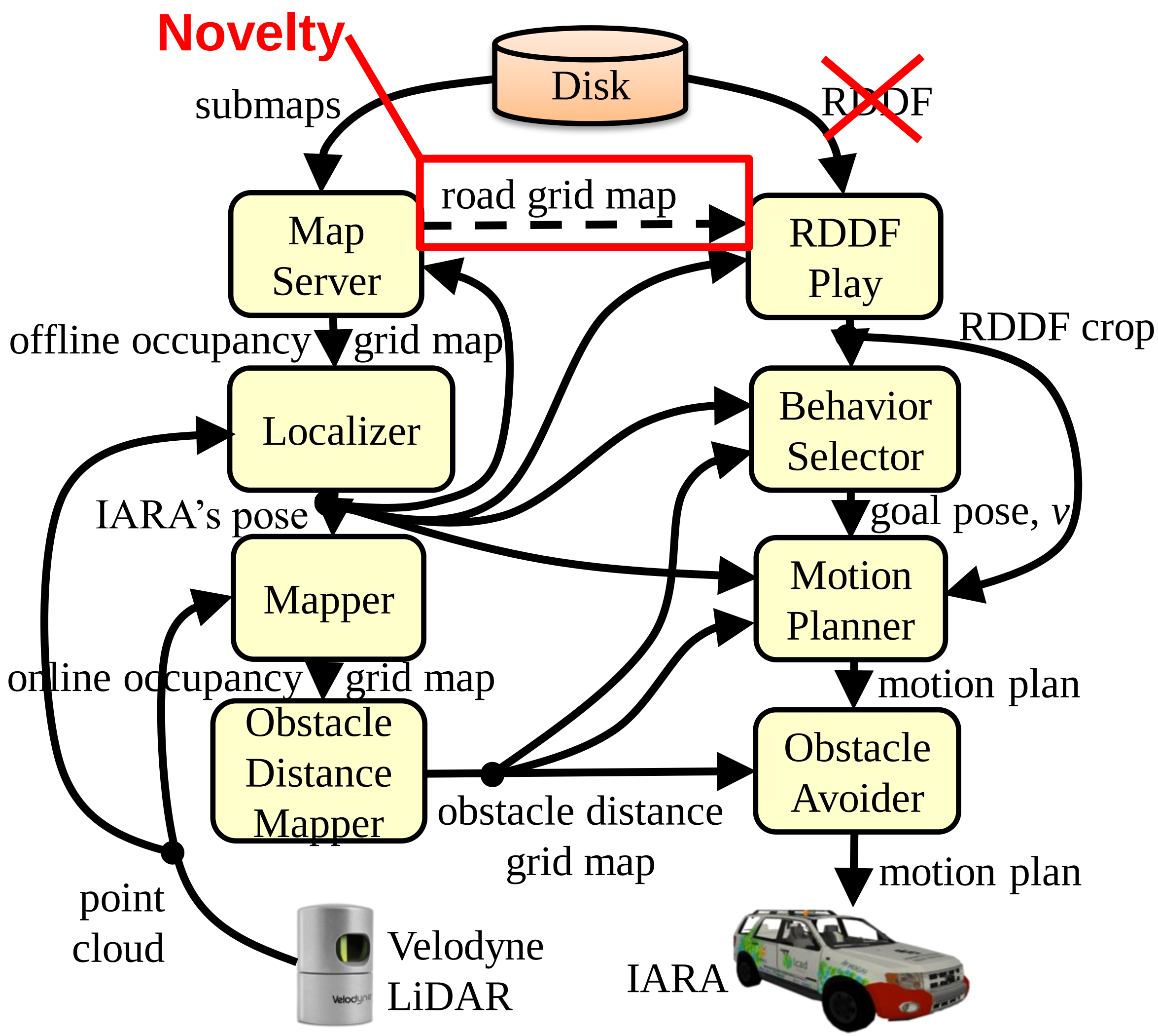
Latitude (m)	Longitude (m)	Yaw (rad)	Velocity (m/s)	Steering (rad)	Timestamp (s)
7757673.765872	-363604.529653	0.687338	2.814645	0.010953	1502316065.290736
7757674.193567	-363604.220413	0.689274	3.087685	0.013366	1502316065.490572
7757674.716373	-363603.802926	0.689999	3.306929	0.007510	1502316065.690336

Road Annotations

Description	Type	Code	Yaw (rad)	Latitude (m)	Longitude (m)	Radius (m)
SPEED_LIMIT	7	8	-2.015	7753405.4	-364951.6	0.0
BUMP	6	0	0.103	7757355.4	-364102.4	-1.4439133
TRAFFIC_SIGN	9	17	0.451	7757077.6	-363696.0	12.0

The Use of Road Grid Maps in Autonomous Cars

The IARA Software System:



The Use of Road Grid Maps in Autonomous Cars

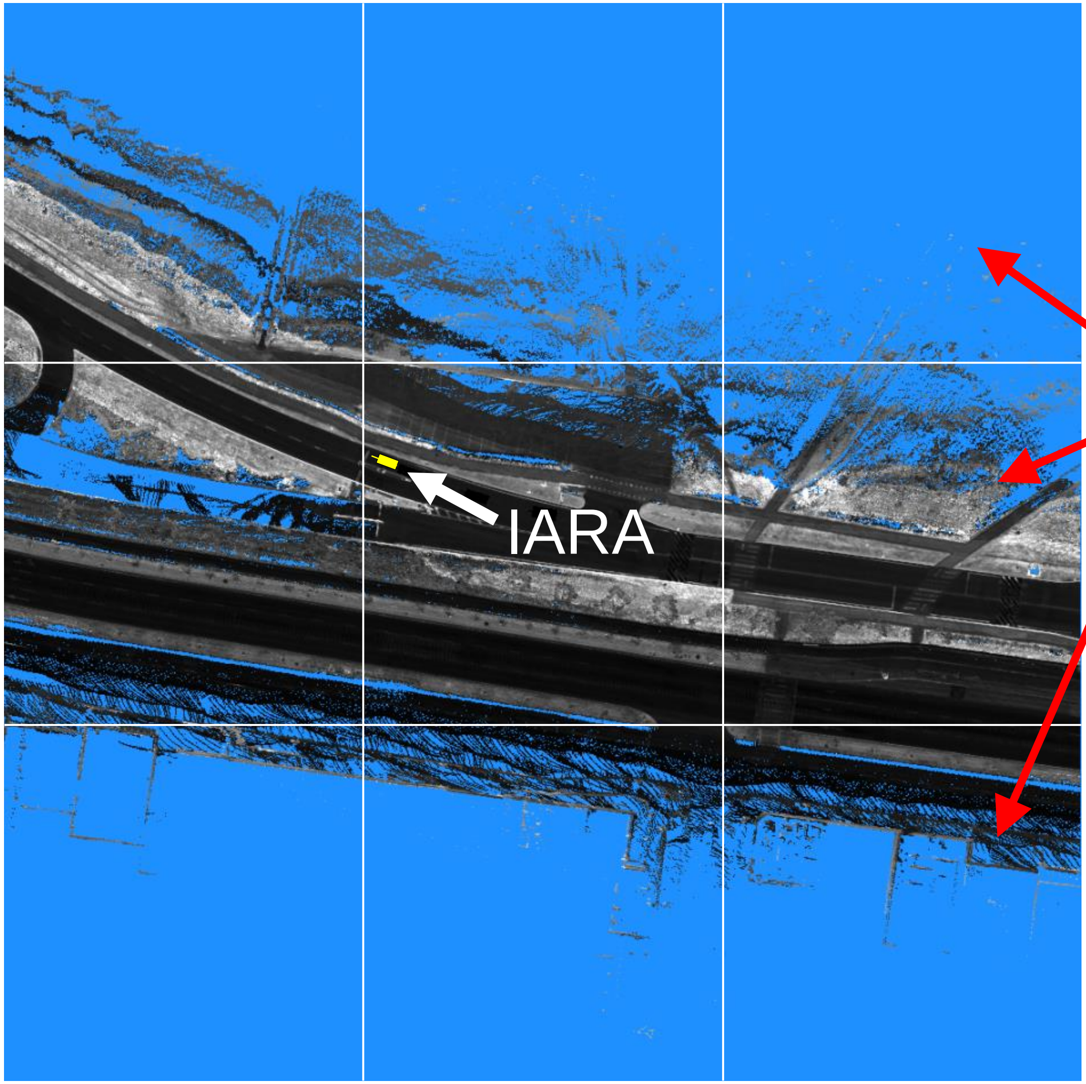
The IARA Software System

- Each grid map of size 210 m × 210 m is actually stored as nine submaps (3 × 3) of size 70 m × 70 m.
- Every time IARA crosses the limits of the central submap, new submaps are fetched from disk and a new grid map is built and published by Map Server.

The Use of Road Grid Maps in Autonomous Cars

The IARA Software System

New submaps will be fetched



These submaps will be discarded

The Use of Road Grid Maps in Autonomous Cars

Computing RDDFs from Road Grid Maps

Algorithm 1: compute_rddf_from_road_grid_map()

Input: *pose*, *road_grid_map*
Output: *RDDF_crop*

1: *rddf.wps_ahead* ← get_waypoints_ahead(*pose*, *road_grid_map*, *150*)
2: *rddf.wps_behind* ← get_waypoints_behind(*pose*, *road_grid_map*, *50*)
3: *RDDF_crop* ← smooth_rddf_using_conjugate_gradient(*rddf*)
4: **return** (*RDDF_crop*)

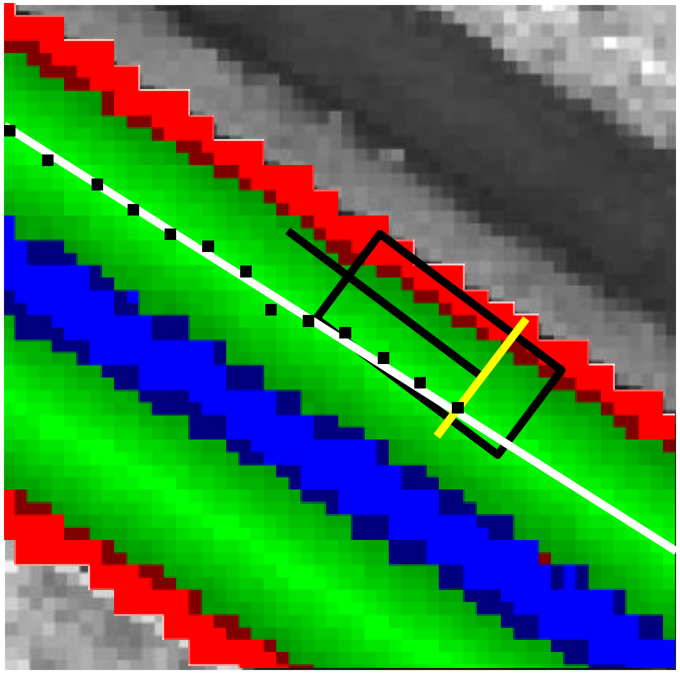
smooth_rddf_using_conjugate_gradient() solves an optimization problem using the same objective function used by Dolgov, Thrun, Montemerlo & Diebel (2010), but considering only the term that measures the smoothness of the path (i.e. the displacement of the waypoints).

The Use of Road Grid Maps in Autonomous Cars

Computing RDDFs from Road Grid Maps

Algorithm 2: get_waypoints_ahead()

Input: *pose*, *road_grid_map*, *num_waypoints*
Output: *wps_ahead*
1. *next_pose* ← *pose*
2. for *i* = *0* to *num_waypoints* – *1*
3. *wps_ahead[i]* ← get_lane_central_pose (*next_pose*, *road_grid_map*)
4. *next_pose* ← add_distance_to_pose (*wps_ahead[i]*, *0.5*)
5. *next_pose.yaw* ← *atan2* (*next_pose.y* – *wps_ahead[i].y*, *next_pose.x* – *wps_ahead[i].x*)
6. return (*wps_ahead*)



Methodology

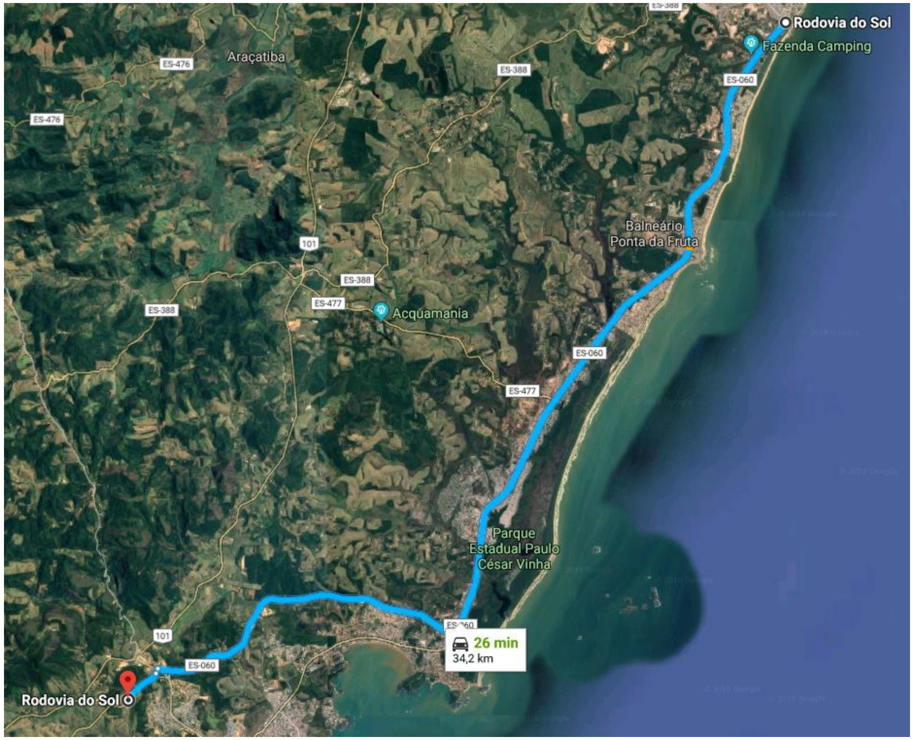
IARA – Intelligent Autonomous Robotic Automobile

Methodology

Datasets

Locations

The ring road of UFES
(3.7 km)

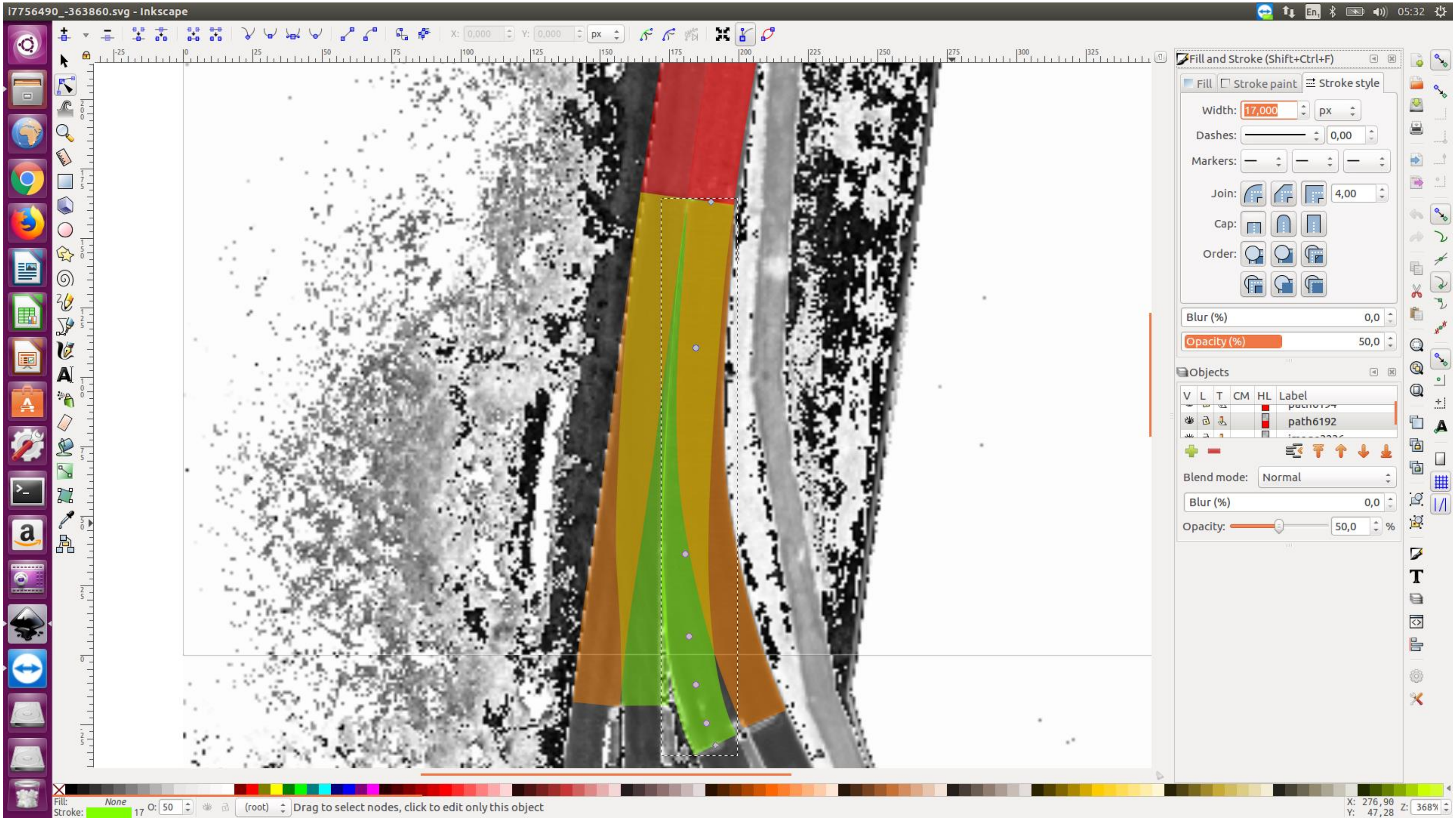


Rodovia do Sol Highway
(34.2 km)

Methodology

Datasets – Ground Truth Annotation

Inkscape – vector graphics software



Methodology

Datasets

Characteristics	UFES	Highway
Purpose	Training (80%) & Test (20%)	Test (100%)
Data size	120 x 120 cells	120 x 120 cells
Samples	110,544	3,556
Crops	658	3,556
Spacing	5 m	12 m
Rotations	24 x 15 degrees	–
Translations	7 x 0.5 m across	–

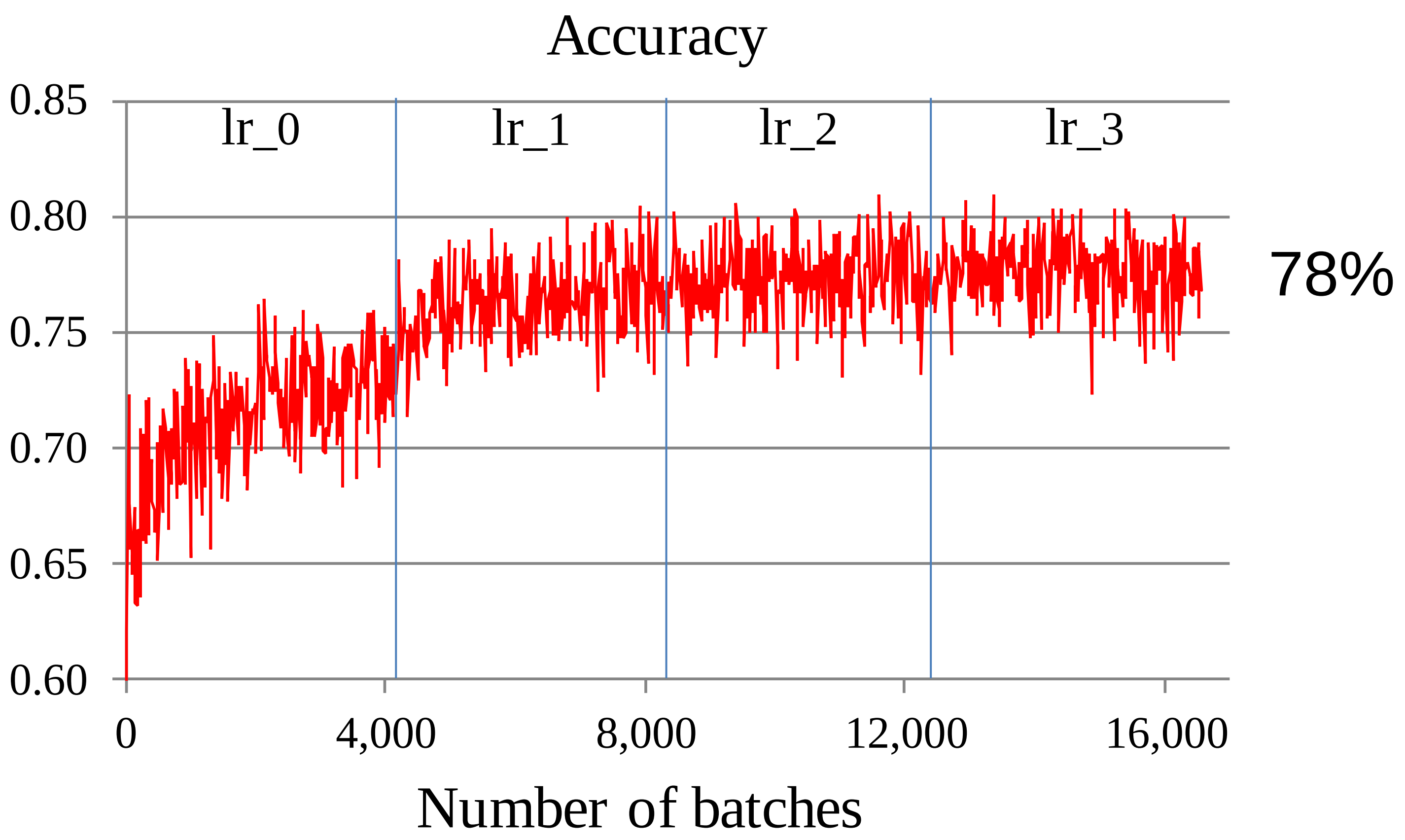
Methodology

ENet DNN – Training Parameters

Parameters	Setting
Batch size	16
Number of batches	16,569 (3 epochs x 88,368 samples / 16 samples per batch)
Learning rate	0.005, reduced by a factor of 10 every 4,142 batches
Solver type	ADAM
Momentum	0.9 (first) and 0.999 (second)
Weight decay	0.0002

Experiments

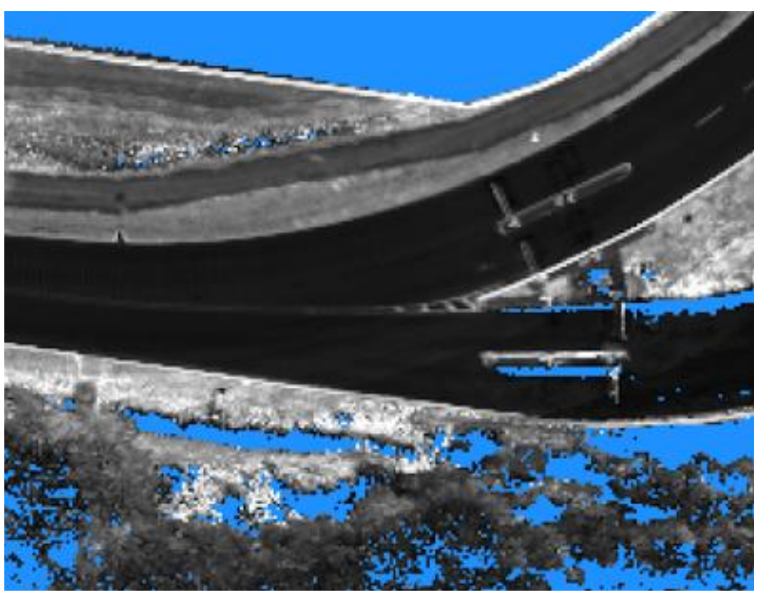
ENet encoder-decoder accuracy during training

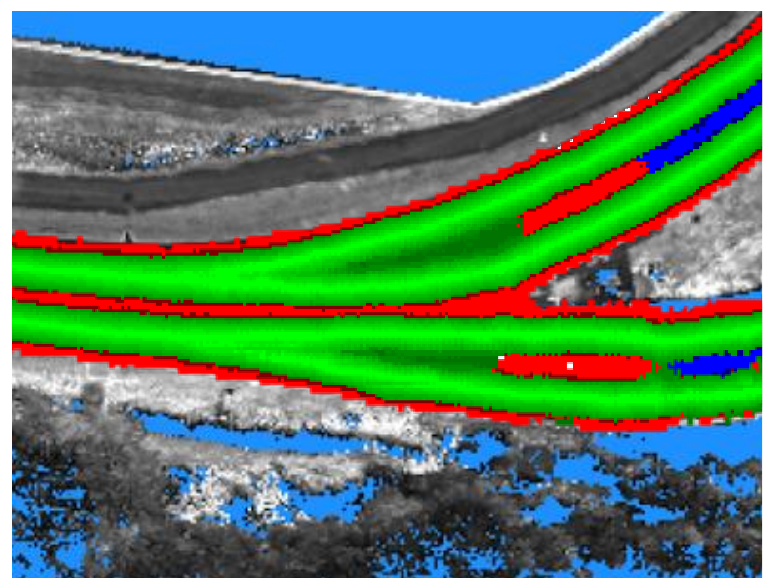


Experiments

ENet accuracy on the UFES test dataset

- 83.7%
- During test, SpatialDropout is removed, thus increasing the performance.
- Example of ENet inference on UFES test dataset

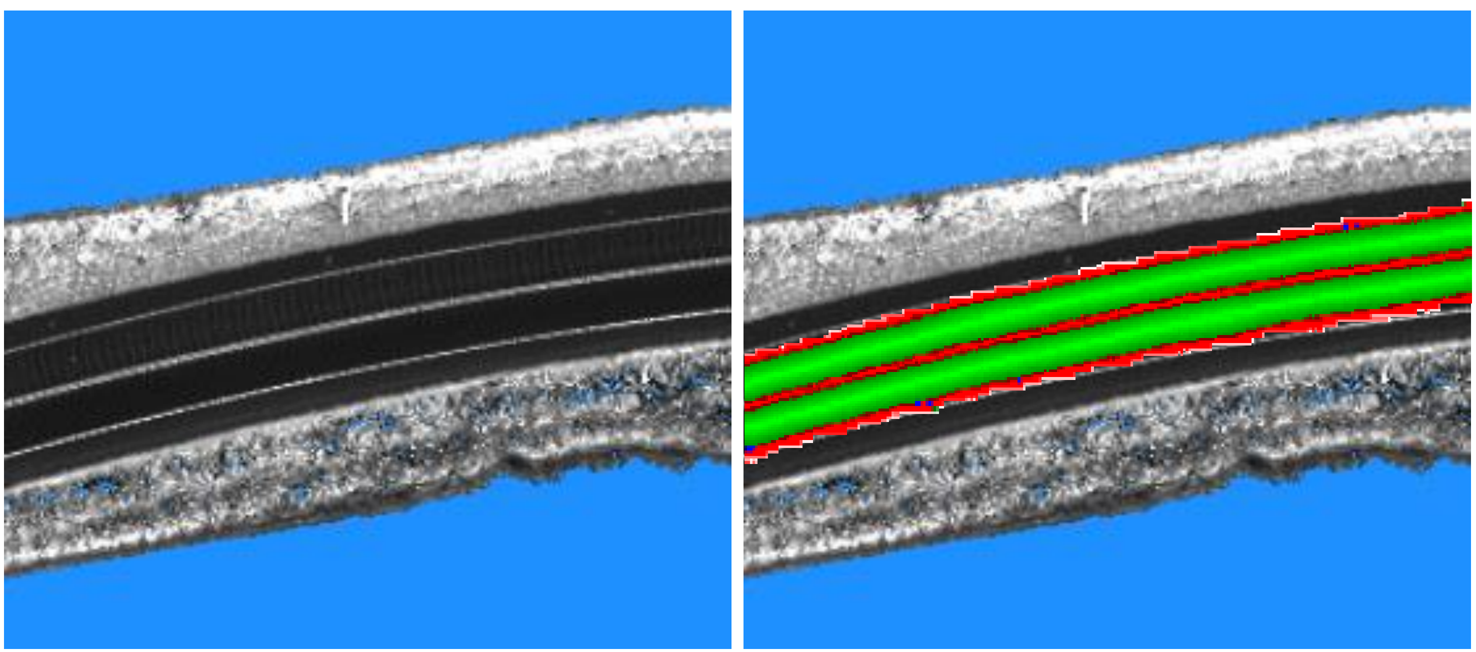




Experiments

ENet accuracy on the Highway dataset

- 64.1%
- Road never seen during training.
- Example of ENet inference on Highway dataset



Experiments

Capability to compute proper RDDFs from road grid maps

- Test #1: Set IARA to drive autonomously using a RDDF extracted from the manually annotated road grid map ground truth of UFES ring road.
- IARA had shown an equivalent or better autonomous navigation performance then the previous method.
- No human intervention was needed.

Experiments

Capability to compute proper RDDFs from road grid maps

- Test #2: ENet was used to generate the road grid map of UFES ring road and we set IARA to drive autonomously using a RDDF extracted from this road grid map.
- IARA had shown an equivalent performance to that of the first experiment.
- No human intervention was needed.

Conclusion

- The inference made by the trained DNN was capable to find existing lanes not marked in the ground truth.
- The experimental results suggest the a DNN segmentation accuracy of 83.7% in the UFES dataset is good enough for autonomous driving with our approach.

Conclusion

Future work

- Try other DNN architectures.
- Extract full road networks from road grid maps and store them as graphs.
- Examine online tools such as OpenStreetMap, Google Maps or Waze, as alternatives for making online high-level path planning decisions.