GHOST

Lane Detection

Ben Lippmeier

benl@gh.st

AI in Robotics Meetup

20th April 2022

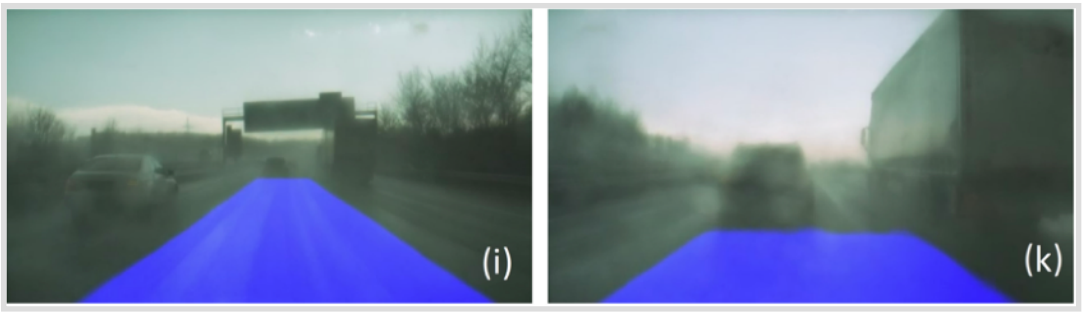
"Lane" is a human social abstraction

LaneAF: Robust Multi-Lane Detection with Affinity Fields, Abualsaud et al, 2021

... full sunlight, sun overhead, no shadows, sky is blue, road is grey, markers are white,
all the same shape, road is straight, car is already in center of lane, not raining, no traffic...

Paint on the road or which way to drive?

Segmenting lane markers in image space
U-Net Conv. Nets. for Biomedical Image Seg.
Ronneberger, Fischer, Brox, 2015



Segmenting the driveable corridor
Where can I drive?
Michalke, Wüst, et al, 2020

Fitting curves to semantic lane lines
PolyLaneNet: Lane Est. with Deep Poly. Regress.
Tabelini, Berriel, et al, 2020

Paint on the road or which way to drive?

Fitting affinitiy fields then reconstructing lane lines
LaneAF: Robust Multi-Lane Det. with Affinity Fields
Abualsaud et al, 2021

Possibly drivable paths
Variational End-to-End Navigation and Localization
Amini, Rosman, Karaman, Rus, 2019

(...no lane markers in this picture...)

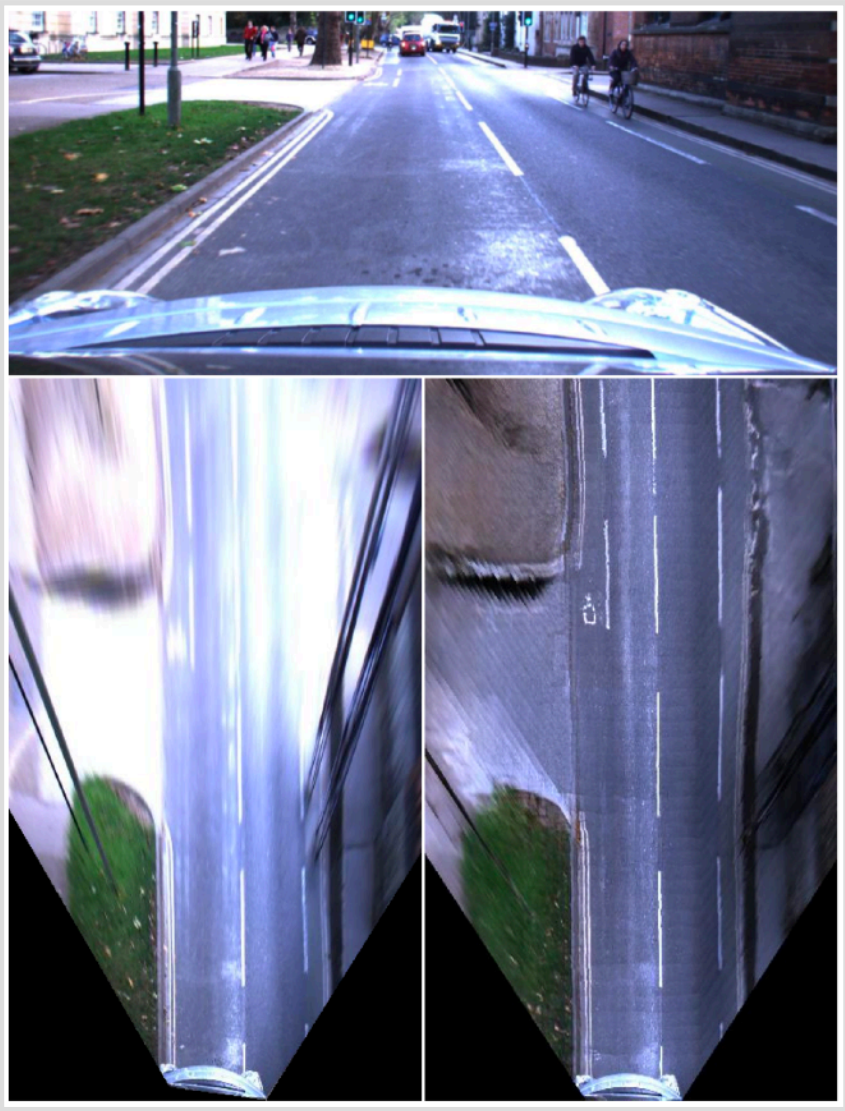
Segmenting paint on the road

- Many backbone networks can be used, UNet, DANet, ResNet...
- Ground truth is segmentation of individual markers on the road.
- Individual lane markers move relative to car speed.
- Simple segmentation networks are prone to false positives.
- Single frame segmentation has no temporal coherence

Segmenting paint on the road

- Simple segmentation networks are prone to false positives.
- This is a base UNet style model with a small number of channels.
- Lane markers are bright things next to not-so-bright things.
- .. so are shadows.

Inverse Perspective Mapping (IPM)

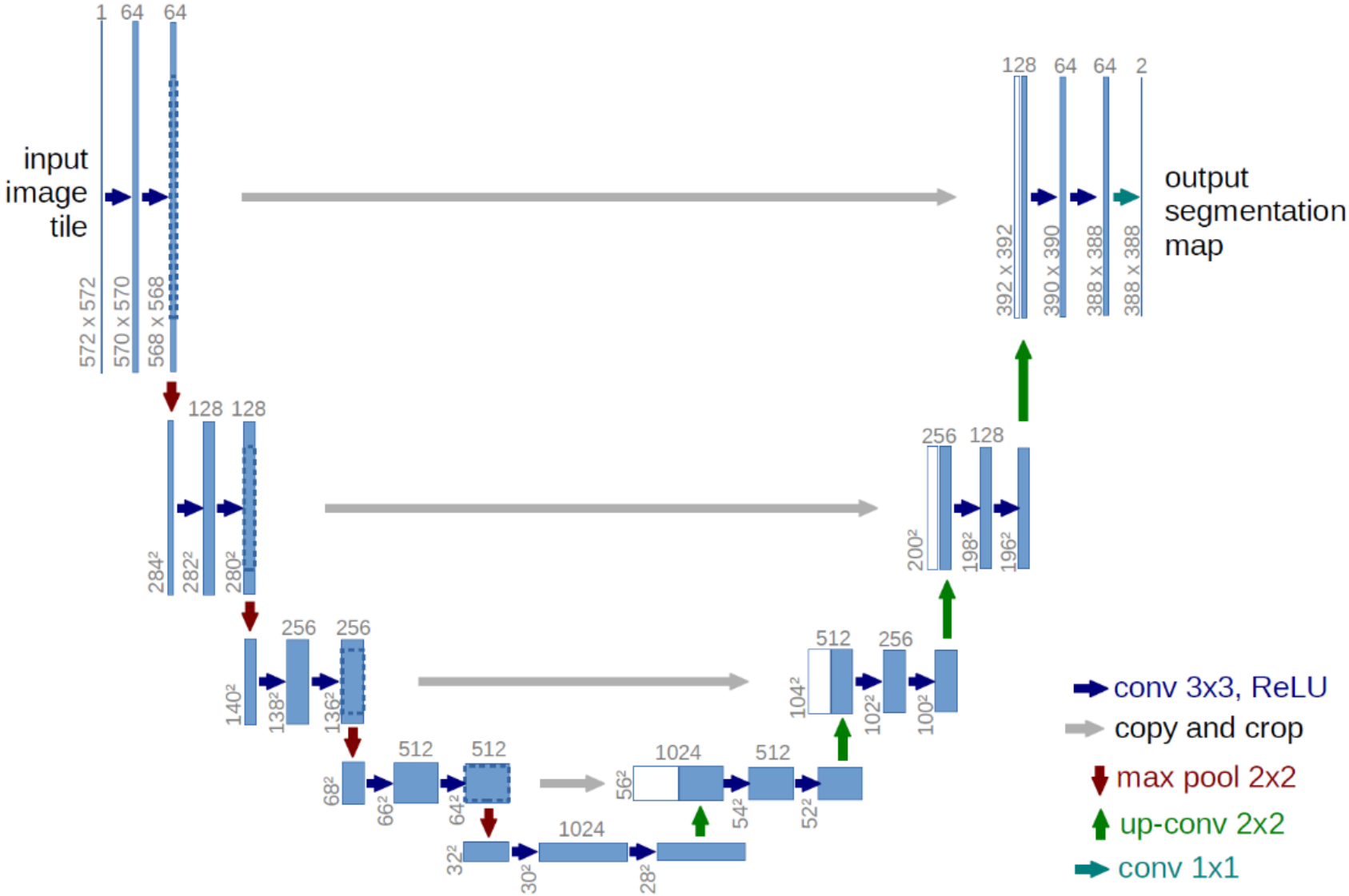


The Right (Angled) Perspective:
Improving the Understanding of Road Scenes
Using Boosted Inverse Perspective Mapping

Bruls, Porav, Kunze, Newman, 2019

Basic U-Net architecture

U-Net Conv. Nets. for Biomedical Image Seg.
Ronneberger, Fischer, Brox, 2015



Ghost Train Definition

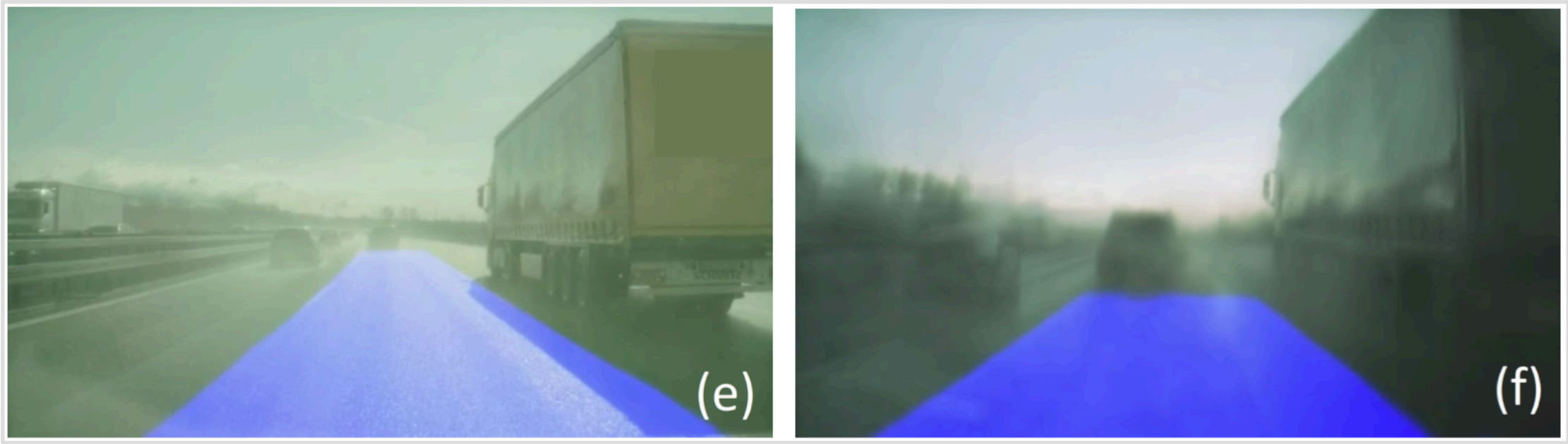
```
class UNet4Bilinear(
  startChannels:  Int,
  store:          Store.OpenCL)
{
  val c = startChannels

  val down1 = this.conv2d(3,    1*c, 3, padding = 1, layerName = "la-down1")
  val down2 = this.conv2d(1*c,  2*c, 3, padding = 1, layerName = "lb-down2")
  val down3 = this.conv2d(2*c,  4*c, 3, padding = 1, layerName = "lc-down3")
  val down4 = this.conv2d(4*c,  8*c, 3, padding = 1, layerName = "ld-down4")

  val neck1 = this.conv2d(8*c,  8*c, 3, padding = 1, layerName = "le-neck1")
  val neck2 = this.conv2d(8*c,  8*c, 3, padding = 1, layerName = "lf-neck2")

  val comb4 = this.conv2d(16*c, 8*c, 3, padding = 1, layerName = "lg-comb4")
  val comb3 = this.conv2d(12*c, 4*c, 3, padding = 1, layerName = "lh-comb3")
  val comb2 = this.conv2d( 6*c, 2*c, 3, padding = 1, layerName = "li-comb2")
  val comb1 = this.conv2d( 3*c, 1*c, 3, padding = 1, layerName = "lj-comb1")

  val score = this.conv2d(1*c,  2,   1, layerName = "lk-score")
```

```
def eval(input: Array4[Float]): Array4[Float] = {

    val x0 = input                    // 3

    val x1 = relu(down1.eval(x0))     // 1*c
    val d1 = pool(x1)

    val x2 = relu(down2.eval(d1))     // 2*c
    val d2 = pool(x2)

    val x3 = relu(down3.eval(d2))     // 4*c
    val d3 = pool(x3)

    val x4 = relu(down4.eval(d3))     // 8*c
    val d4 = pool(x4)

    val n1 = relu(neck1.eval(d4))     // 8*c
    val n2 = relu(neck2.eval(n1))     // 8*c

    val c4 = relu(comb4.eval(concatChannels(x4, up(n2)))) // 16*c -> 8*c
    val c3 = relu(comb3.eval(concatChannels(x3, up(c4)))) // 12*c -> 4*c
    val c2 = relu(comb2.eval(concatChannels(x2, up(c3)))) //  6*c -> 2*c
    val c1 = relu(comb1.eval(concatChannels(x1, up(c2)))) //  3*c -> 1*c

    val s1 = score.eval(c1)
    val s2 = logSoftmax(s1, dim = 2)
    s2
  }
```

```
def up(arr: Array4[Float]): Array4[Float] =
  upsampleBilinear(
    2 * arr.shape.row, 2 * arr.shape.col,
    UpsampleBilinear.CornerAlignment.Center,
    arr)
```

```
def pool (a: Array4[Float]): Array4[Float] =
  maxpool(2, 2, dropLeftover = false, a)
```

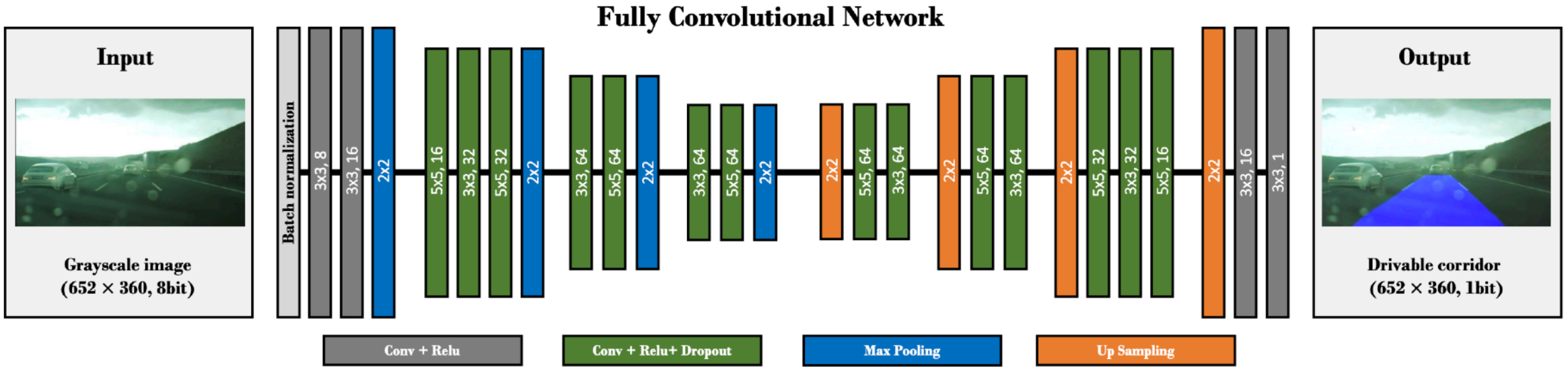
Segmenting the drivable corridor



Where can I drive? Michalke, Wüst, et al, 2020

- Segment the part of the road that can be driven into, not the paint.
- Can combine collision avoidance with lane detection.
- Model learns that drivable areas tend to have a particular shape.
- The corridor in image space does not move relative to car speed.

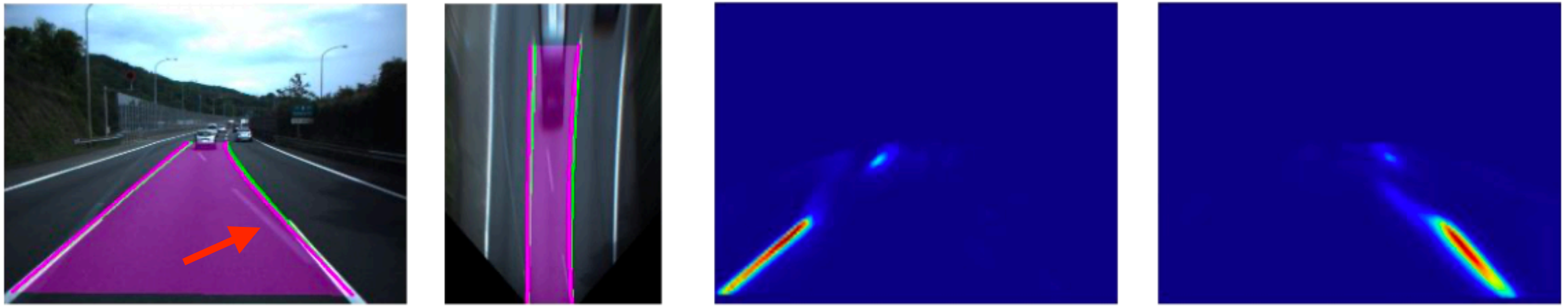
Segmenting the drivable corridor



Where can I drive? Michalke, Wüst, et al, 2020

- This is a Simple Auto Encoder (SAE), not using skip connections.
- Forcing information through the bottleneck helps regularize the output.
- Other approaches could include pyramid pooling, feature fusion etc.

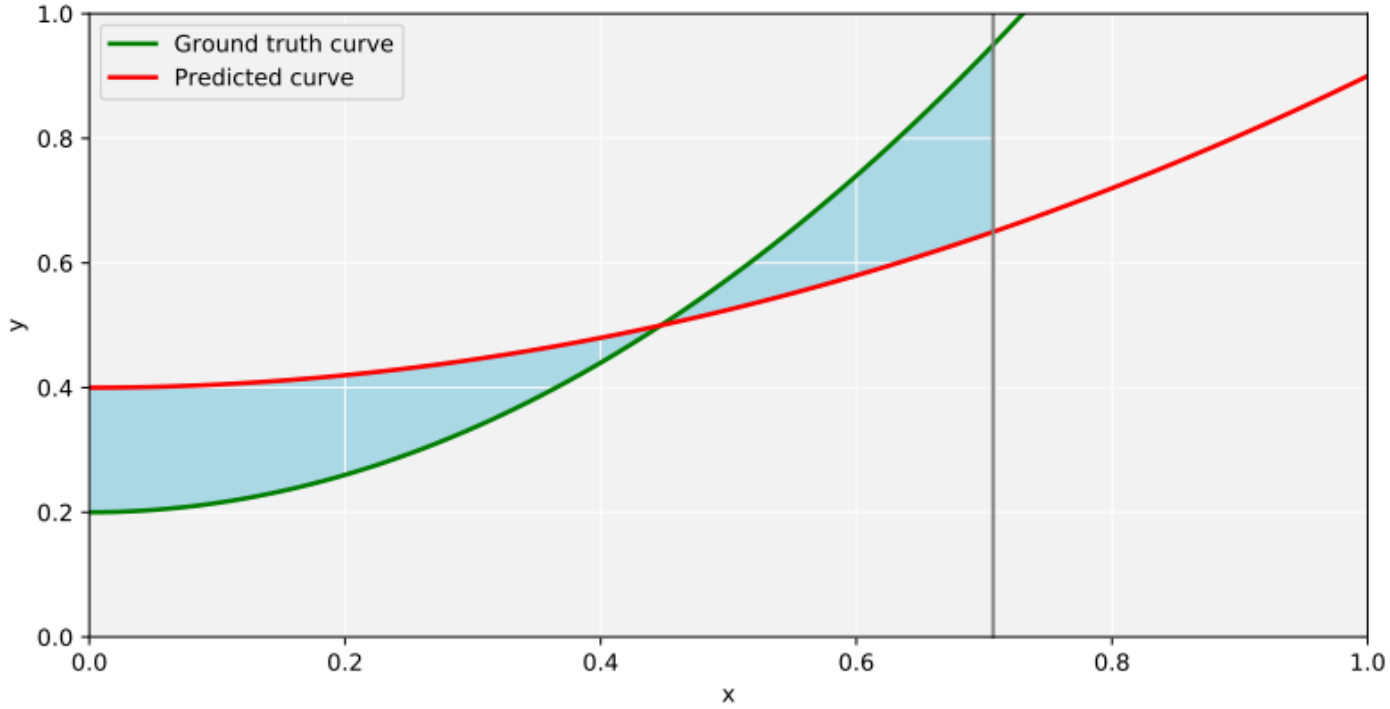
Fitting curves to semantic lane lines



End-to-end Lane Detection through Differentiable Least-Squares Fitting. Gansbeke, Brabandere et al, 2019

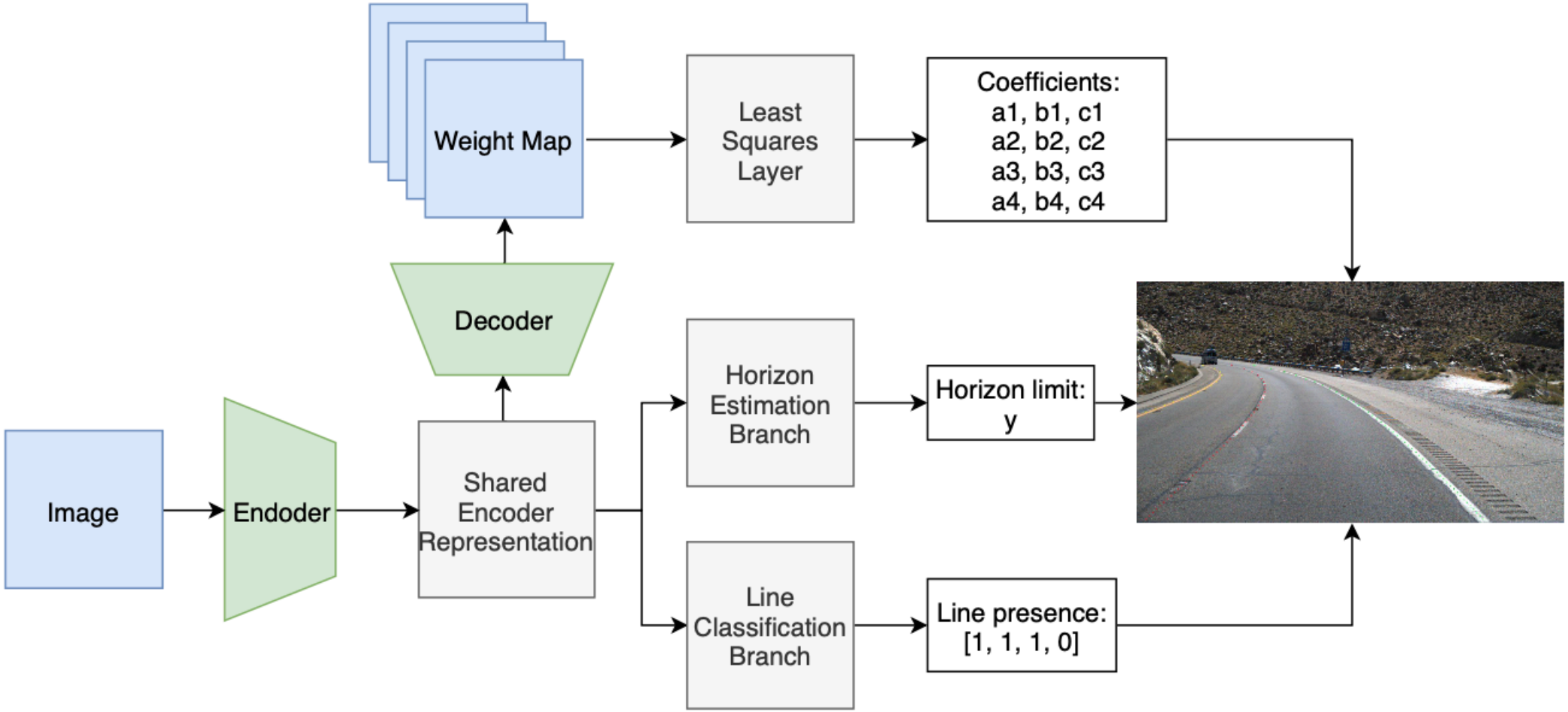
- Two stage network first segments markers then fits curves to them
- Network is trained end-to-end, ground truth is polynomial curves
- Uses a differentiable operator for least squares curve fitting.
- Gradient information from the curve fitting aids segmentation.
- System requires a fixed lane geometry.

Fitting curves to semantic lane lines



- Loss measures sum of mean square error between predicted curve and ground truth, up to a fixed distance from the car.
- Loss is computed in the ortho/top-down view.
  Transform between perspective and ortho is differentiable.

Fitting curves to semantic lane lines



Differentiable Linear Least Squares Curve Fitting

Given $X\beta = Y$ with $X \in \mathbb{R}^{m\times n}, \beta \in \mathbb{R}^{n\times 1}, \text{and } Y \in \mathbb{R}^{m\times 1}$

the least squares fit is: $\beta = \left(X^T X\right)^{-1} X^T Y.$

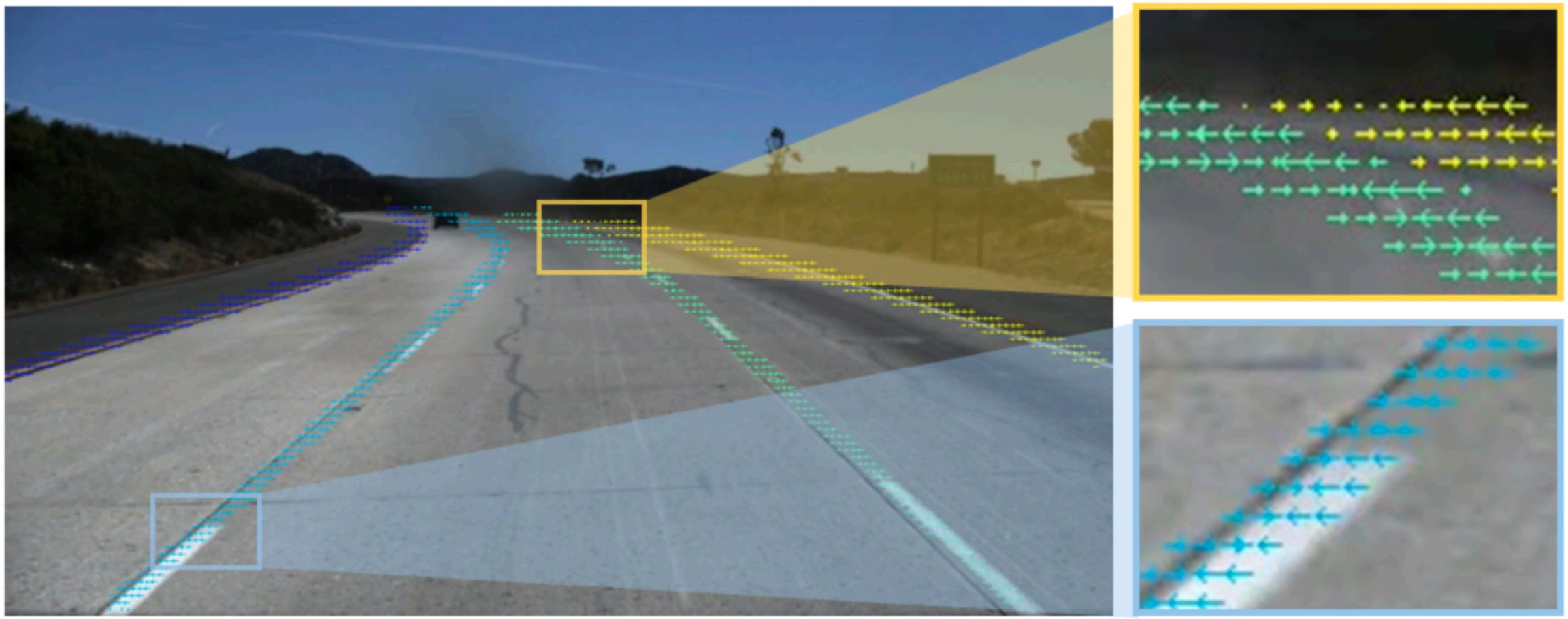
Given a segmentation map, interpret each point as a triple (x,y,w) with weight w.

Use weighted least squares fit:

$$WX\beta = WY.$$

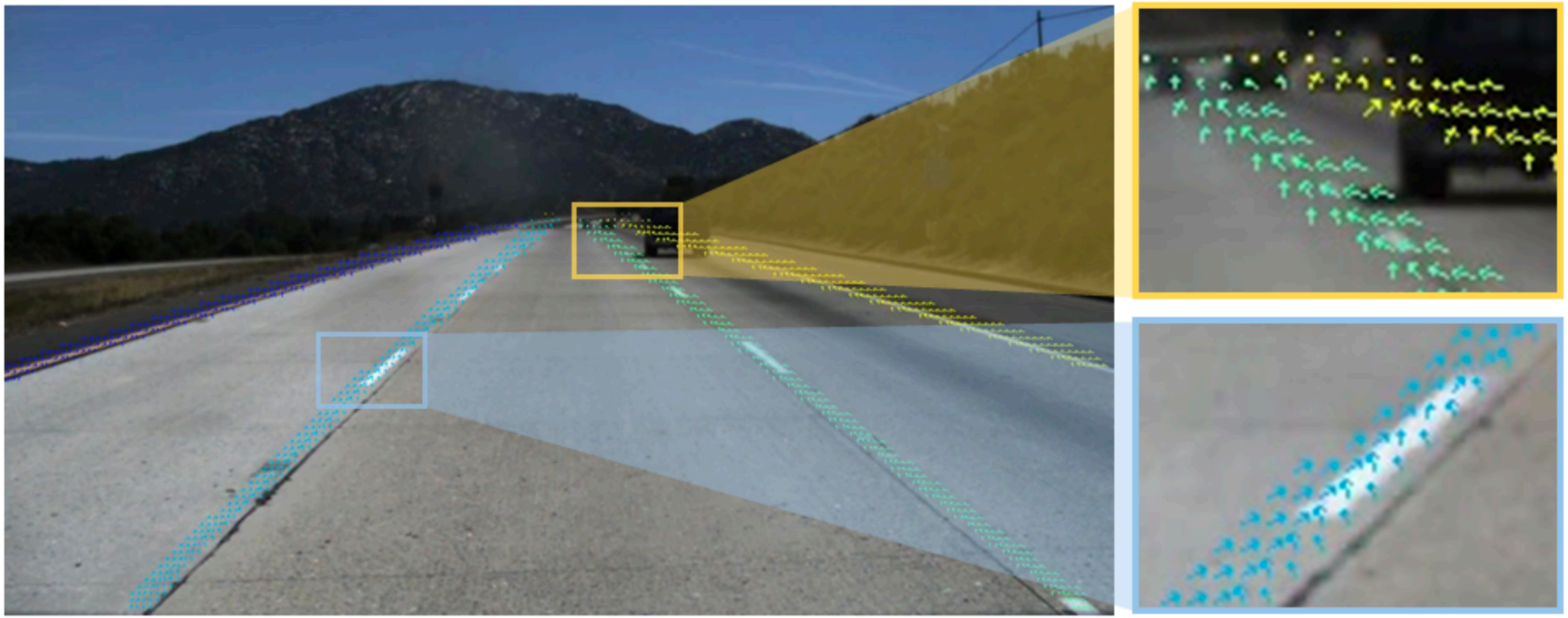
.. which can be rearranged to the same form as the first equation.

Per-pixel Affinity Fields



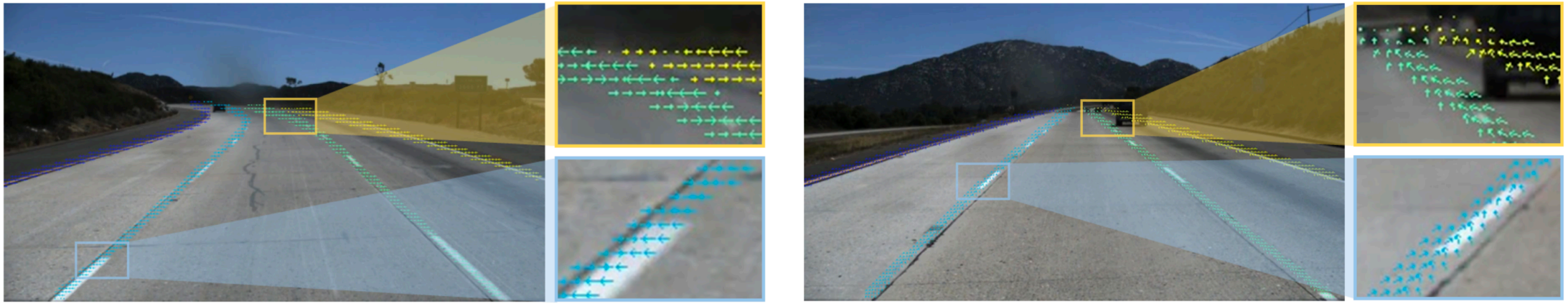
LaneAF: Robust Multi-Lane Detection with Affinity Fields
Abualsaud, Liu, Lu, Situ et al, 2021

Per-pixel Affinity Fields



LaneAF: Robust Multi-Lane Detection with Affinity Fields
Abualsaud, Liu, Lu, Situ et al, 2021

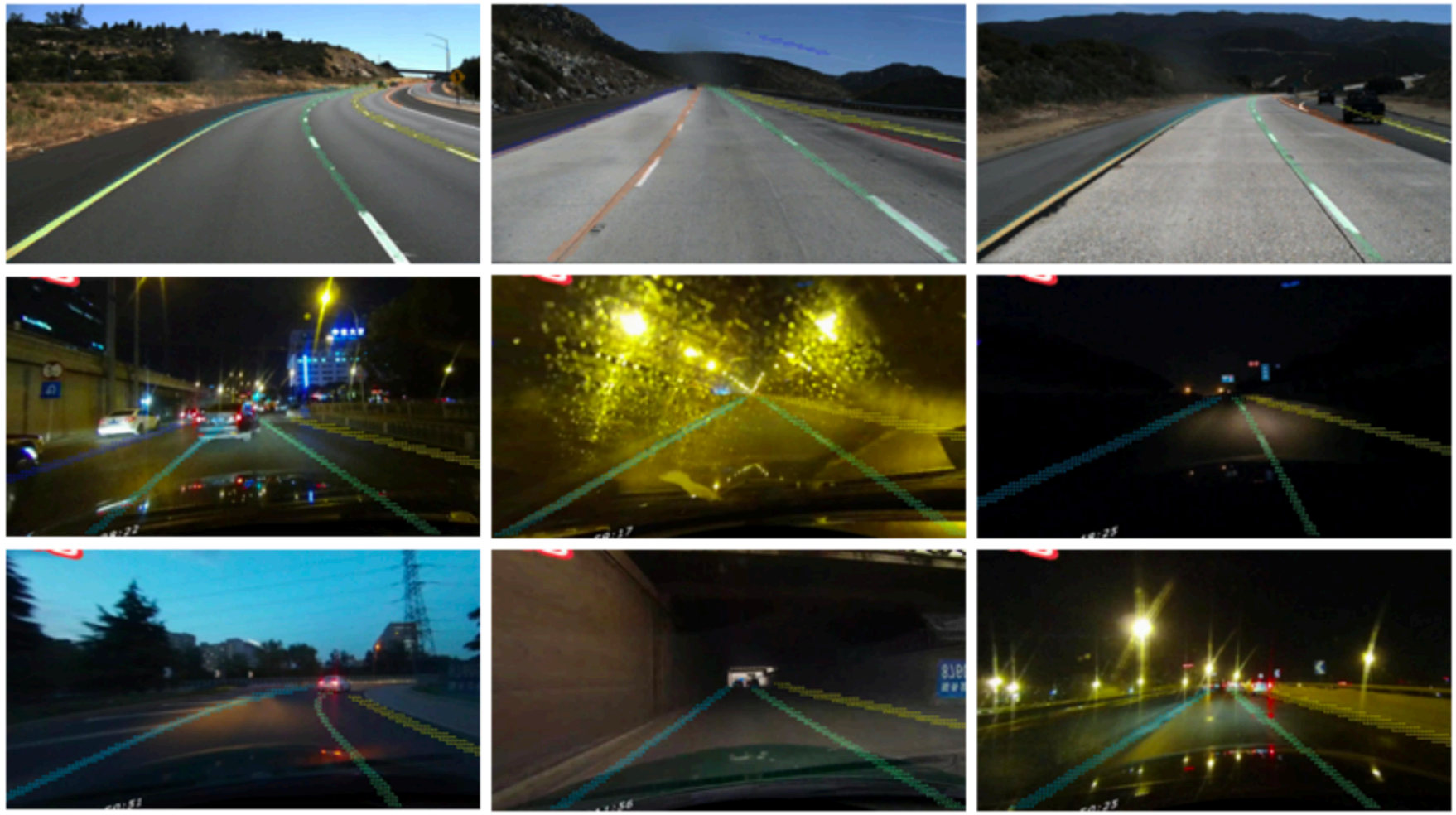
Per-pixel Affinity Fields



LaneAF: Robust Multi-Lane Detection with Affinity Fields
Abualsaud, Liu, Lu, Situ et al, 2021

- Affinity field is learned using ground truth as lane lines.
- Lane lines are reconstructed bottom-up pass over the field.
- Handles arbitrary lane geometry, including splits and joins.
- Output is still a segmentation mask, rather than geometric parameters that can be passed directly to steering control.

Adversarial Robustness vs Learned Bias



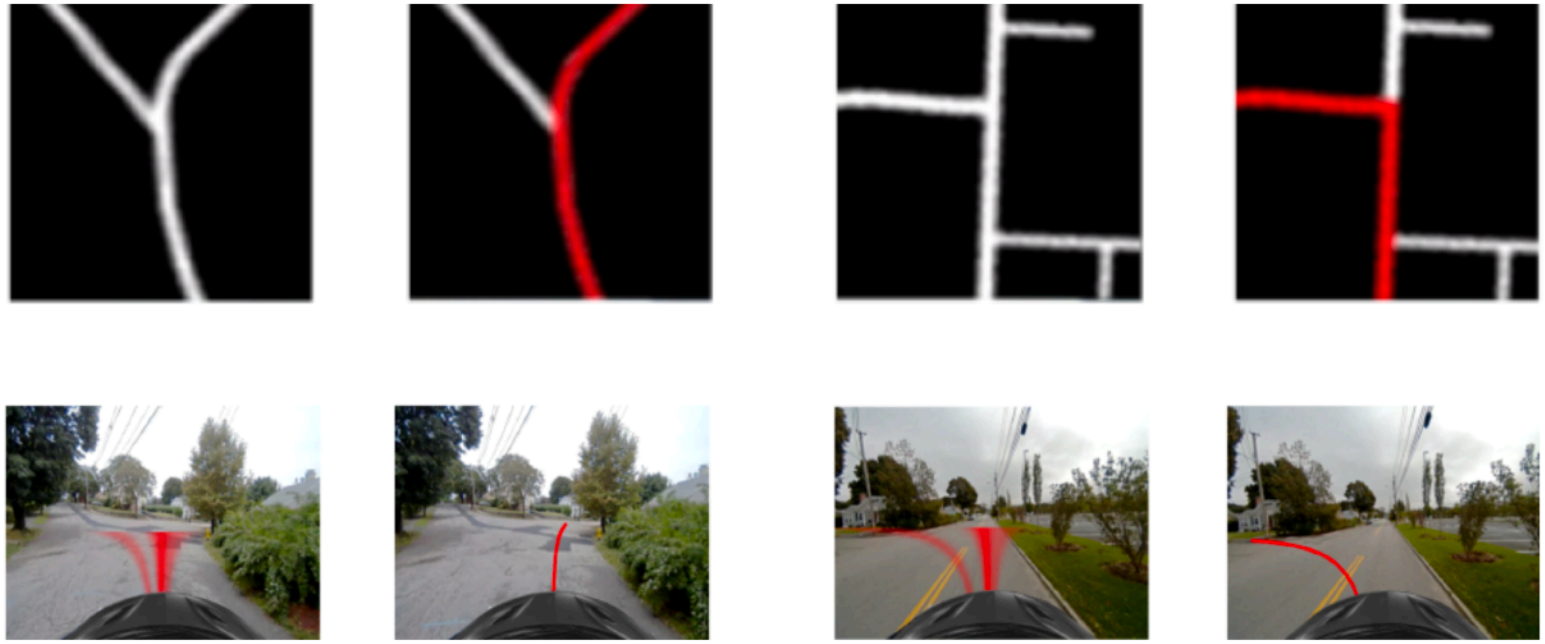
LaneAF: Robust Multi-Lane Detection with Affinity Fields
Abualsaud, Liu, Lu, Situ et al, 2021

Possibly Drivable Paths

Variational End-to-End Navigation and Localization
Amini, Rosman, Karaman, Rus, 2019

- Produce a proability distribution over possible paths.
- Paths that we're not sure about get lower proability.
- We don't always need to supply a valid path.

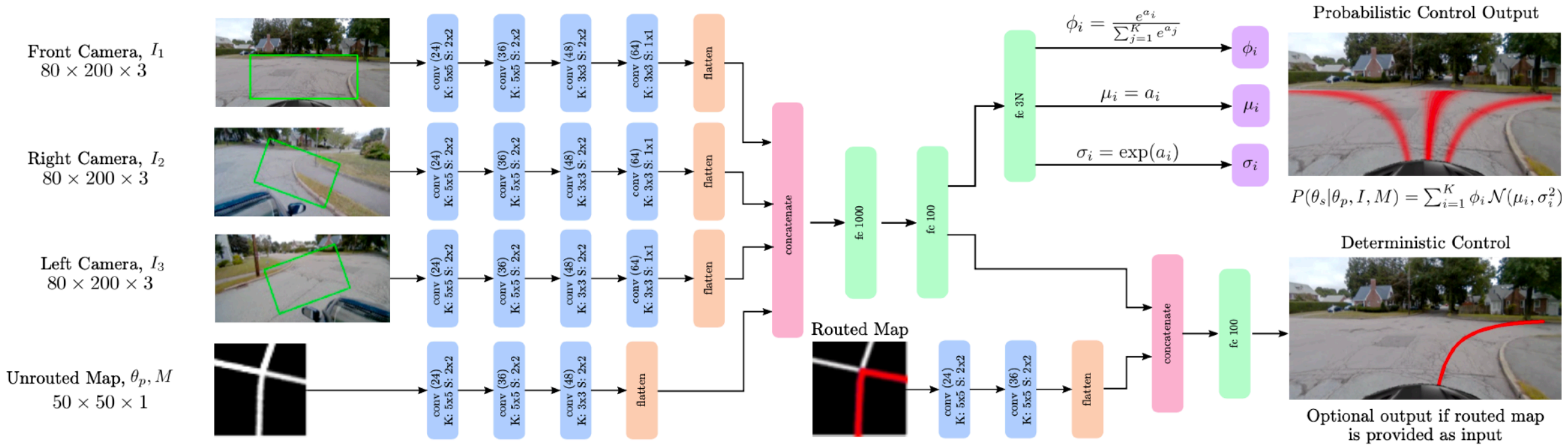
Possibly Drivable Paths



Variational End-to-End Navigation and Localization
Amini, Rosman, Karaman, Rus, 2019

- Learn a Gaussian Mixture Model over possible driving paths.
- Desired route can be selected by provding a top-down picture.

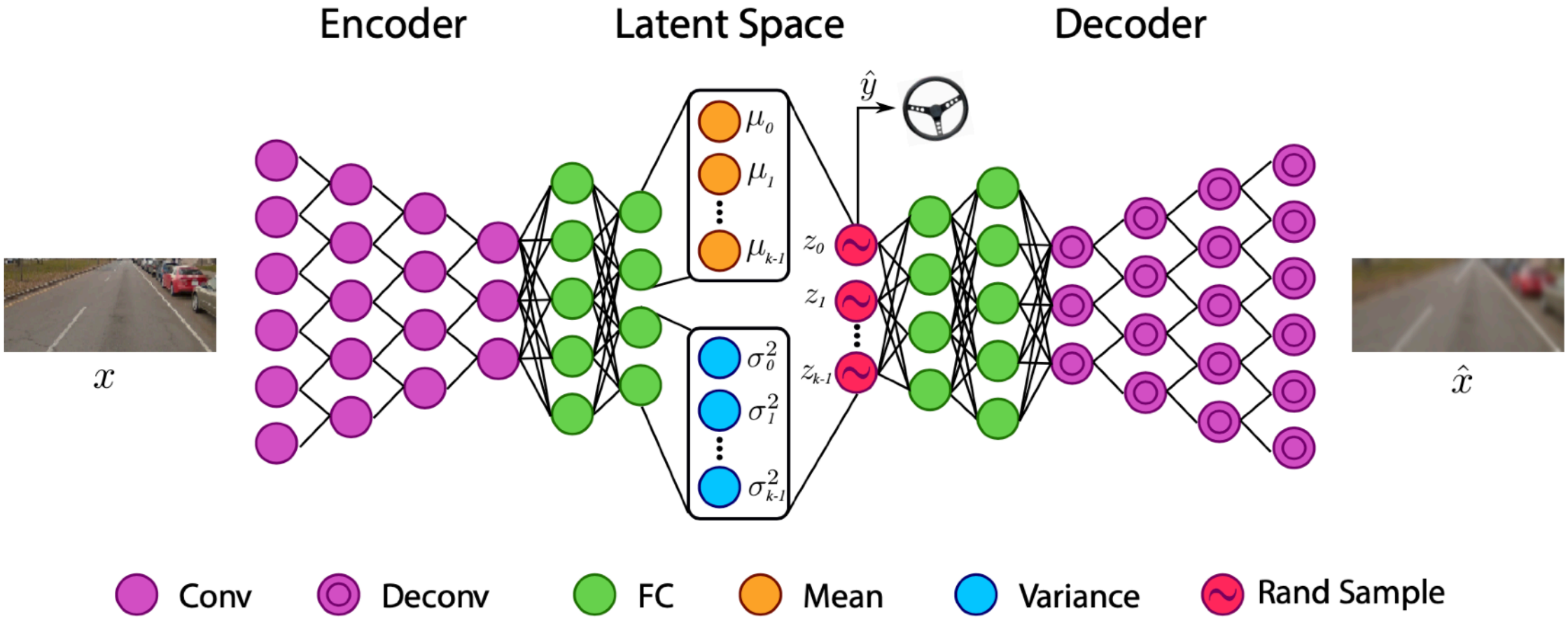
Possibly Drivable Paths



Variational End-to-End Navigation and Localization
Amini, Rosman, Karaman, Rus, 2019

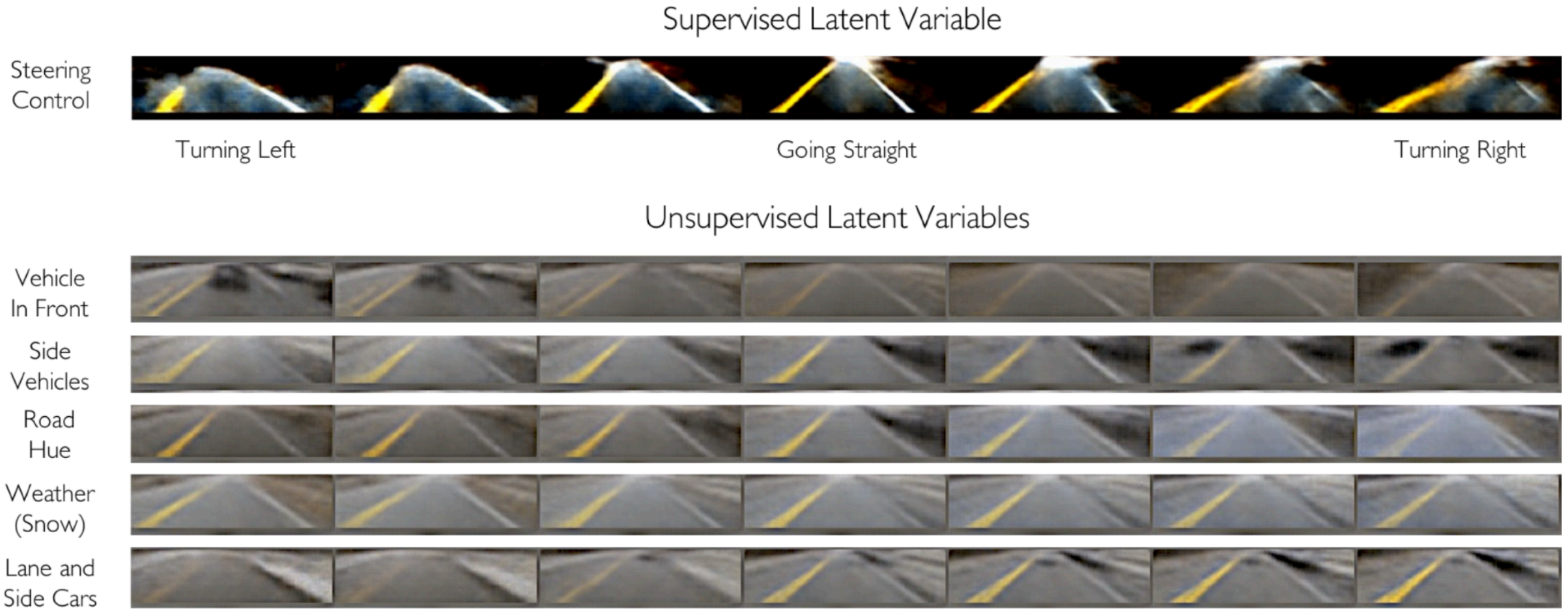
- Learned Gaussian Mixture Model has a fixed three clusters / lanes.
- Model considers navigation rather than obstacle detection.

End-to-end driving with Variational Autoencoders



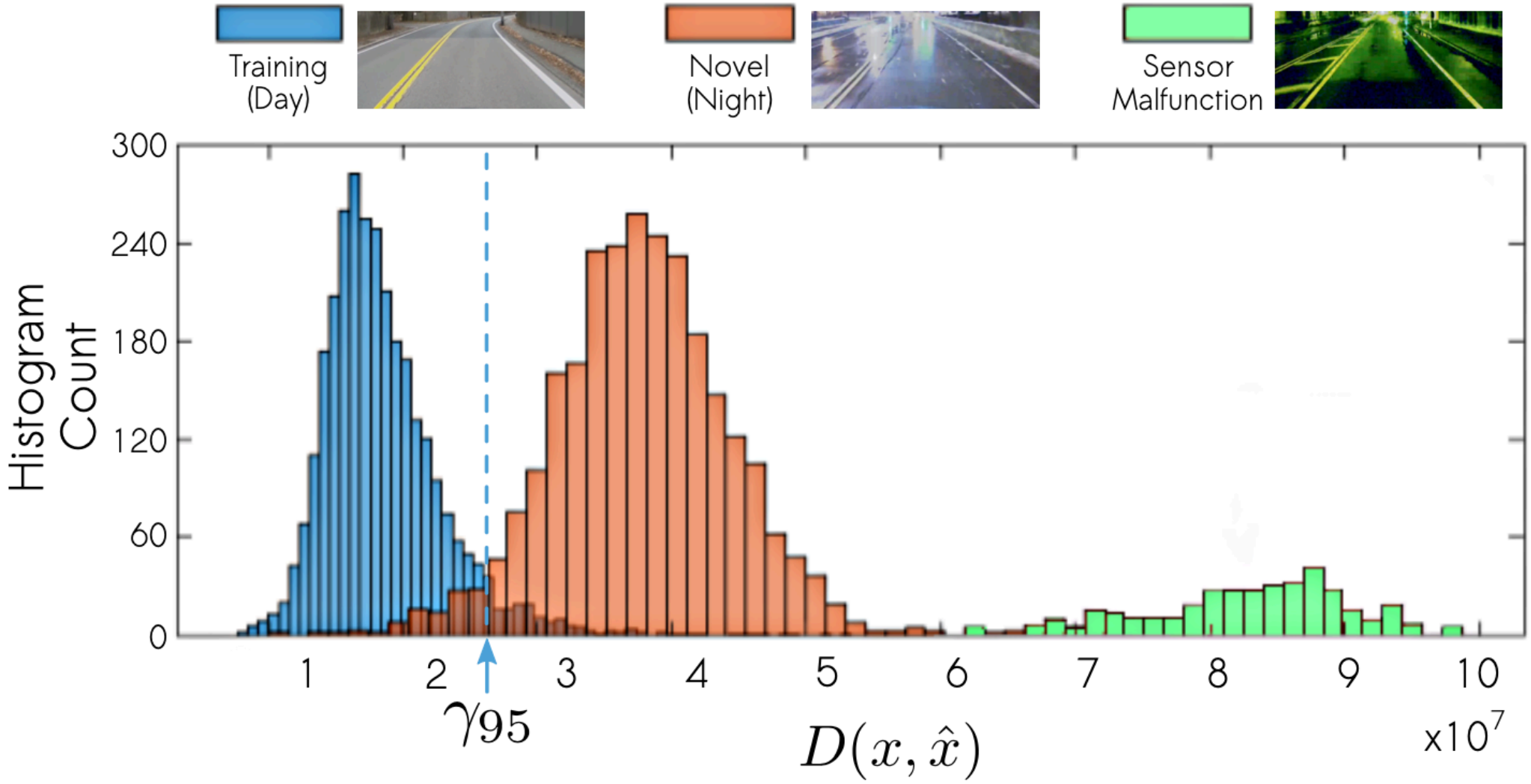
Variational Autoencoder for End-to-End Control of Autonomous Driving with Novelty Detection...
Amini, Schwarting, Rosman, Araki, Karaman, Rus, 2018

End-to-end driving with Variational Autoencoders



Variational Autoencoder for End-to-End Control of Autonomous Driving with Novelty Detection...
Amini, Schwarting, Rosman, Araki, Karaman, Rus, 2018

Propagating Uncertaing back into Image Space



Propagating Uncertaing back into Image Space

Variational Autoencoder for End-to-End Control of Autonomous Driving with Novelty Detection...
Amini, Schwarting, Rosman, Araki, Karaman, Rus, 2018

Lessons

- Appropriate approach depends on needs of the overall system.

- Detecting paint on the road is not the same as driving a car.

- One-shot end-to-end driving is possible:
  camera to wheels in a single network,
  but hard to debug if it does not perform well.

- Moving from L2/L3+ to L4 (no human fallback) is a big step up.
  In such a system it is more important to detect the uncommon cases than handle the common cases.