Self-Driving Cars

Lecture 8 – Road and Lane Detection

Prof. Dr.-Ing. Andreas Geiger

Autonomous Vision Group

University of Tübingen / MPI-IS

EBERHARD KARLS
UNIVERSITÄT
TÜBINGEN

ellis
European Laboratory for Learning and Intelligent Systems

Agenda

8.1 Introduction

8.2 Road Segmentation

8.3 Lane Marking Detection

8.4 Lane Detection

8.5 Lane Tracking

8.1
Introduction

Road and Lane Detection

State-of-the-Art Commercial Systems:

- **Lane departure warning**
  Driver is warned when leaving the current lane

- **Lane keeping support**
  Driver is assisted with minimal steering interventions

- **Lane keeping/lane departure protection**
  Car keeps the current lane autonomously

Road and Lane Detection

https://www.youtube.com/watch?v=OQkdvi55woA

Representations

Road and Lane Detection: Representations

Road and Lane Detection:

- Navigate without detailed global map by sensing “drivable areas” in vicinity of car
- Multiple cues available (geometric, semantic, and man-made cues)
- Input: image / Lidar/ radar. Multiple possible **output representations**

Road and Lane Detection: Representations

Road Segmentation:

- Classify each pixel in the image as either road or non-road
- Purely semantic cue (“where is a vehicle allowed to drive?”)
- Doesn’t consider “reachability” / geometry, no information about lanes

Road and Lane Detection: Representations

Driving Corridor Prediction:

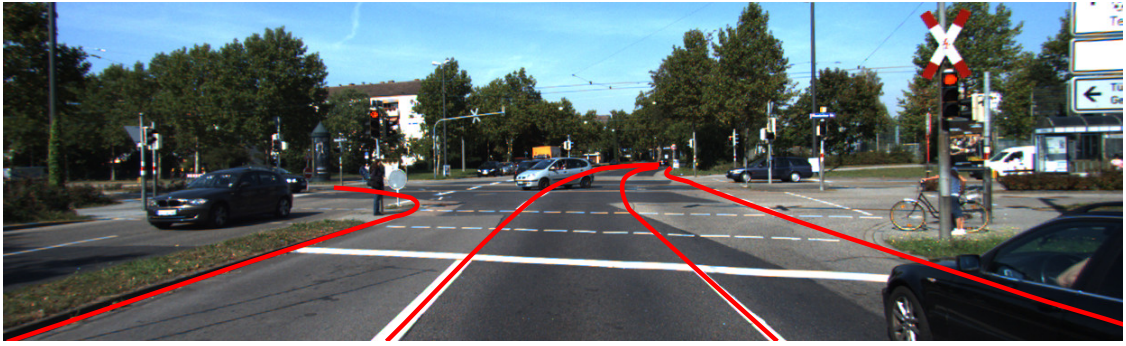
- Estimate the corridor ahead within which the vehicle is supposed to drive
- May or may not consider obstacles (green) within the driving corridor
- Multiple driving corridors, the relevant one depends on the high-level plan

Road and Lane Detection: Representations

Driving Corridor Prediction:

- Estimate the corridor ahead within which the vehicle is supposed to drive
- May or may not consider obstacles (green) within the driving corridor
- Multiple driving corridors, the relevant one depends on the high-level plan

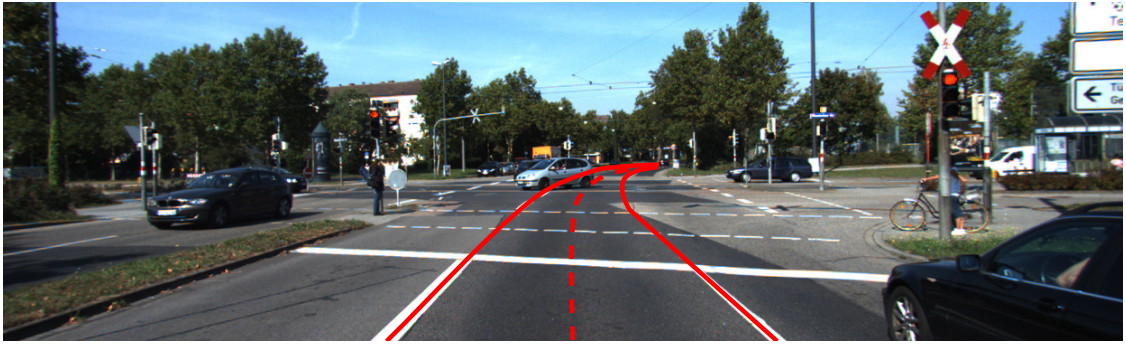
Road and Lane Detection: Representations



Lane Marking Detection:

- Roads are often subdivided into lanes using lane markings
- Detect lane markings/road curbs and fit parametric model to them
- Works well for highways, but in the city lane markings are often not reliable

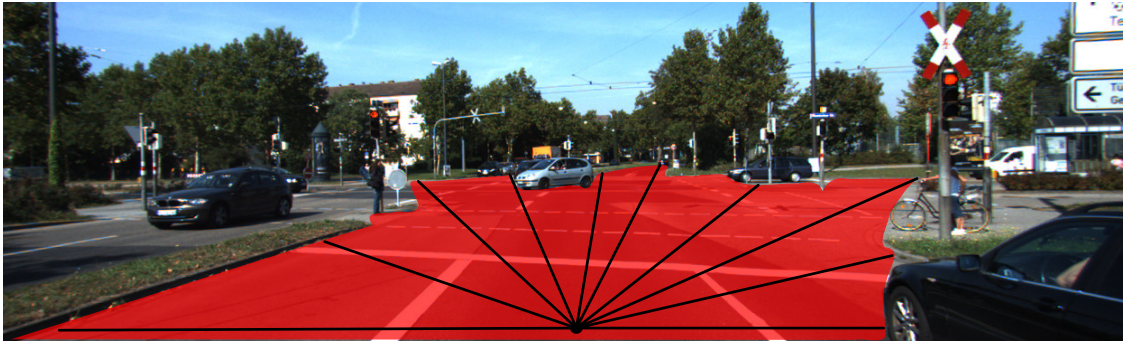
Road and Lane Detection: Representations



Lane Detection:

- Lane detection groups two lane markings into one lane
- Yields information about lane width and centerline (dashed)
- Navigation relevant lane depends on high-level plan

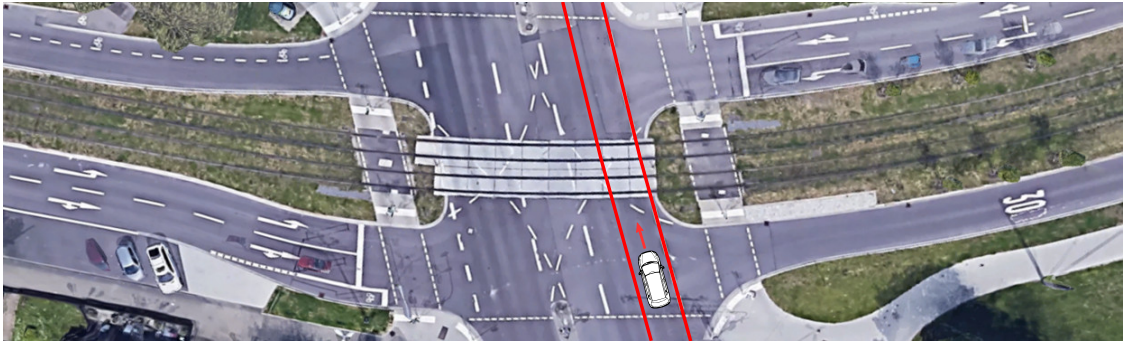
Road and Lane Detection: Representations



Freespace Estimation:

- Estimate places that can be reached without collision
- Purely geometric cue (“which places can be directly reached?”)
- Doesn’t require appearance cues (lane markings, road texture)

Road and Lane Detection: Representations



2D vs. 3D Representation:

- Estimating quantities in the 2D image domain is not directly useful
- Must be mapped into 3D (or bird's eye view) where vehicle is controlled
- Given an estimate of the road surface (e.g., 3D plane), this is possible

Ambiguities

Road and Lane Detection: Ambiguities

Observations can be ambiguous:

- Road Segmentation: which parts are considered road? (vs. sidewalk etc.)
- Freespace Estimation: is the sidewalk drivable? (height threshold)
- Lane Boundary Detection: neither lane markings nor curbs in this image

Road and Lane Detection: Ambiguities

Observations can be ambiguous:

- This is an easier case (similar to highway driving)
- Road segmentation, lane boundaries and driving corridor clearly defined
- Freespace Estimation: is the grass field on the right traversable or not?

Road and Lane Detection: Ambiguities

Observations can be ambiguous:

- Temporary lane markings during road construction: old markings often visible
- Which lane marking counts? Which ones are relevant (here: bicycle lane?)
- Lane markings often hard to detect (visibility), demaged or missing

Road and Lane Detection: Ambiguities

Observations can be ambiguous:

- Lane markings missing, dedicated to other objects, or not within field of view
- Other traffic relevant scene elements (blue sign)
- Freespace includes sidewalk in this case

Road and Lane Detection: Ambiguities

Observations can be ambiguous:

- Navigation in parking lots not possible based on lane markings alone
- Accessibility of entryway (left) depends on height of curb (subtle cue)
- Curbs can be of arbitrary height, how to detect them robustly?

Road and Lane Detection: Ambiguities

Observations can be ambiguous:

- Rural streets often unmarked, lane detection very difficult
- Detection of road signs relevant to determine allowed driving directions
- Sometimes difficult to detect if car is parked or not (affecting driving corridor)

Road and Lane Detection: Ambiguities

Observations can be ambiguous:

- Often very unstructured environments which are hard to capture parametrically
- Scene elements sometimes resemble lane markings, but are not
- Some areas not designated for driving, yet not clearly marked (here: poles)

Road and Lane Detection: Ambiguities

Observations can be ambiguous:

- Lane markers often missing at intersections
- Tram railroad tracks can be confused with lane markings
- Structured but at the same time very unstructured, holistic reasoning required

Road and Lane Detection: Ambiguities

Observations can be ambiguous:

- Often navigation based solely on lane markings not a good idea
- In particular, construction sites are often badly structured
- Construction signage changes street layout, lanes partially obstructed

Road and Lane Detection: Ambiguities

Observations can be ambiguous:

- Here no lane markings but implicitly two lanes
- However, both lanes narrow down into one based on very subtle cues
- Requires combination of structured planning and reactive control

Road and Lane Detection: Ambiguities

Observations can be ambiguous:

- Sometimes lane markings can have very different semantics
- Allowed driving directions not always easy to infer
- Heavy occlusions might obstruct the sensor field of view

Overview

	Advantages	**Disadvantages**
Road Segment.	• Works in unstructred env. • Doesn't rely on accurate geometry	• Semantic ambiguities • Doesn't output path information
Driving Corridor	• More directly relevant to control • Also comprises obstacle inform.	• Depends on high-level plan • Ambiguous/difficult to estimate
Lane Marking	• Relatively easy inference task • Compact parametric models	• Markings missing/deteriorated • Only in structured env. (highway)
Lane	• Directly yields path / centerline • Compact parametric models	• Ambiguous/difficult to estimate • Only in structured env. (highway)
Freespace Est.	• Works in unstructred env. • Doesn't require semantics	• Relies on accurate geometry • Doesn't output path information

8.2
Road Segmentation

Deep Convolutional Image Segmentation

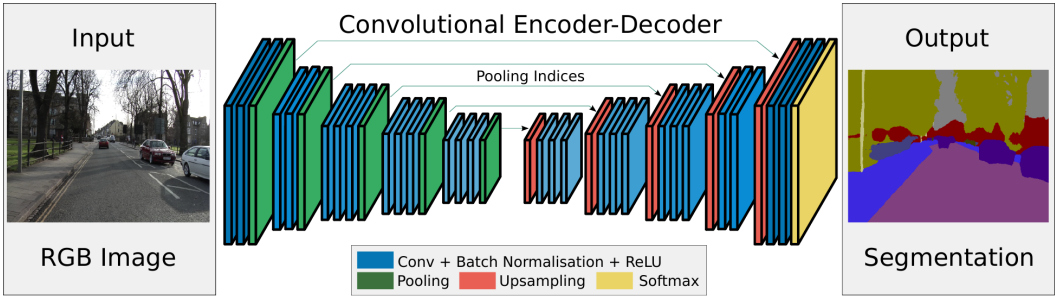
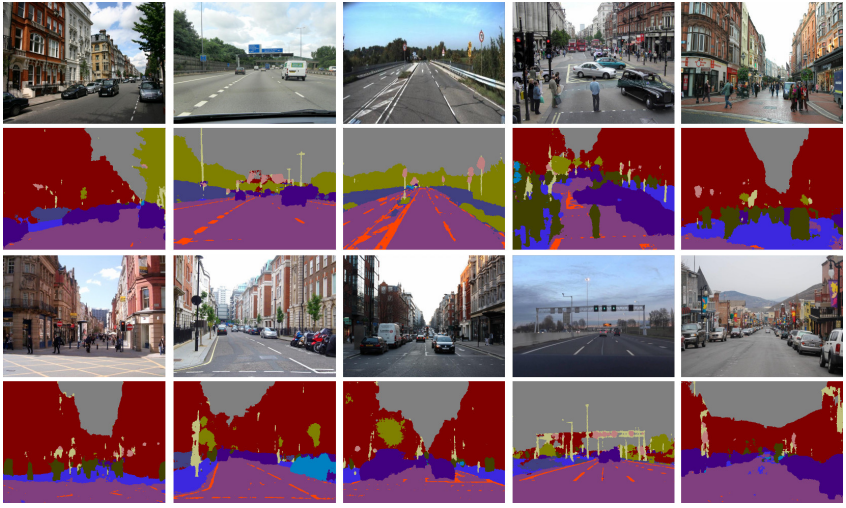


Fig. 2. An illustration of the SegNet architecture. There are no fully connected layers and hence it is only convolutional. A decoder upsamples its input using the transferred pool indices from its encoder to produce a sparse feature map(s). It then performs convolution with a trainable filter bank to densify the feature map. The final decoder output feature maps are fed to a soft-max classifier for pixel-wise classification.

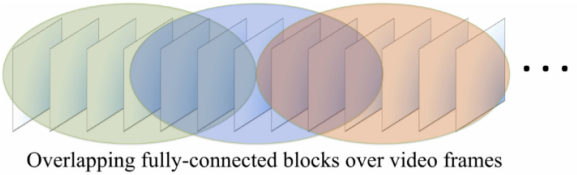
- Use convolution, pooling and upsampling layers to predict per-pixel class label
- Trained using cross-entropy loss. Semantic classes: sky, building, car, road, . . .

Deep Convolutional Image Segmentation



Badrinarayanan, Kendall and Cipolla: SegNet: A Deep Convolutional Encoder-Decoder Architecture for Image Segmentation. PAMI, 2017.

Deep Convolutional Image Segmentation



$$P(\mathbf{x}|\mathbf{P}) = \frac{1}{Z(\mathbf{P})} \exp\big(-E(\mathbf{x}|\mathbf{P})\big)$$

$$E(\mathbf{x}|\mathbf{P}) = \sum_{\mathbf{p}} \psi^u_{\mathbf{p}}(x_{\mathbf{p}}) + \sum_{(\mathbf{p},\mathbf{q})\in\mathcal{E}} \psi^p_{\mathbf{p},\mathbf{q}}(x_{\mathbf{p}}, x_{\mathbf{q}})$$

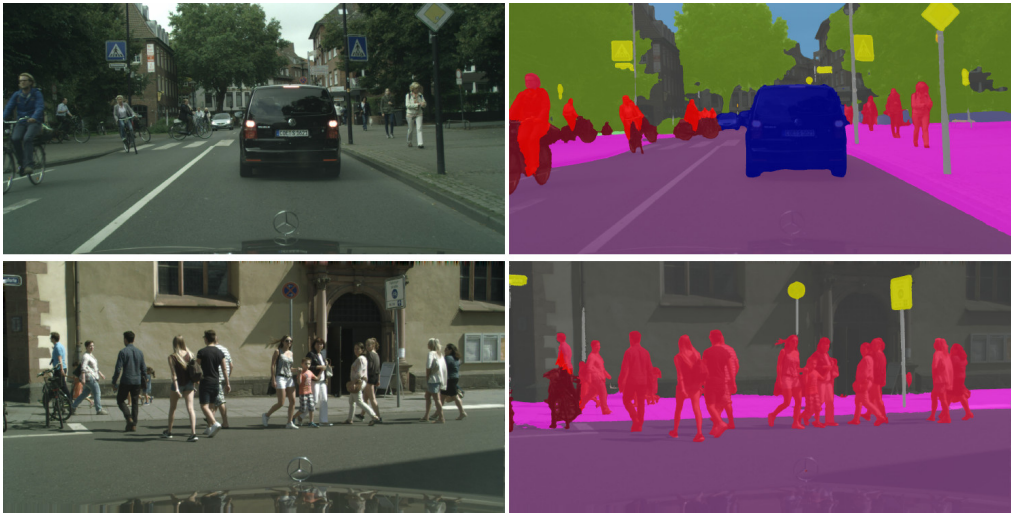
$$\psi^p_{\mathbf{p},\mathbf{q}}(x_{\mathbf{p}}, x_{\mathbf{q}}) = \mu(x_{\mathbf{p}}, x_{\mathbf{q}}) \sum_{m=1}^{M} w^m \kappa^m(\mathbf{f}_{\mathbf{p}}, \mathbf{f}_{\mathbf{q}})$$

$$\kappa^m(\mathbf{f}_{\mathbf{p}}, \mathbf{f}_{\mathbf{q}}) = \exp\left(-\frac{\|\mathbf{f}_{\mathbf{p}} - \mathbf{f}_{\mathbf{q}}\|^2}{\sigma_m^2}\right)$$

- Formulate semantic segmentation as optimization problem over multiple frames
- Temporal and spatial smoothness constraints via fully connected CRF model

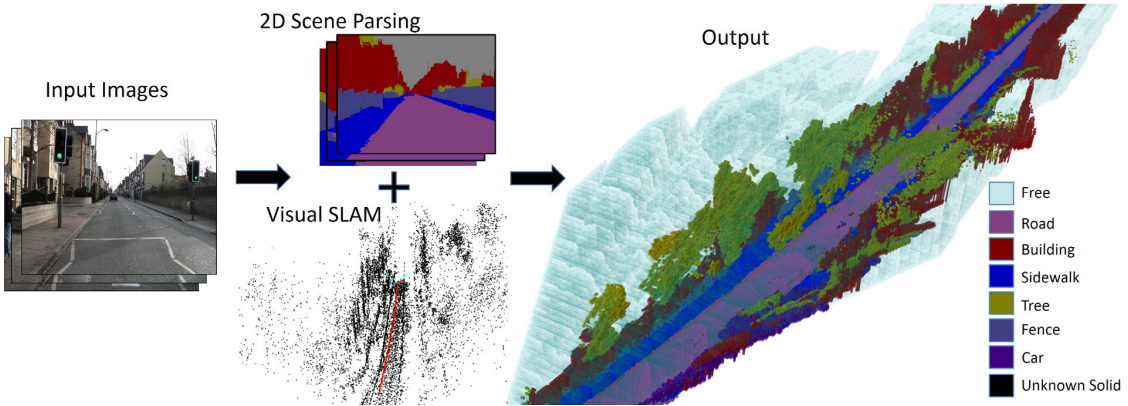
Kundu, Vineet and Koltun: Feature Space Optimization for Semantic Video Segmentation. CVPR, 2016.

Deep Convolutional Video Segmentation



Kundu, Vineet and Koltun: Feature Space Optimization for Semantic Video Segmentation. CVPR, 2016.

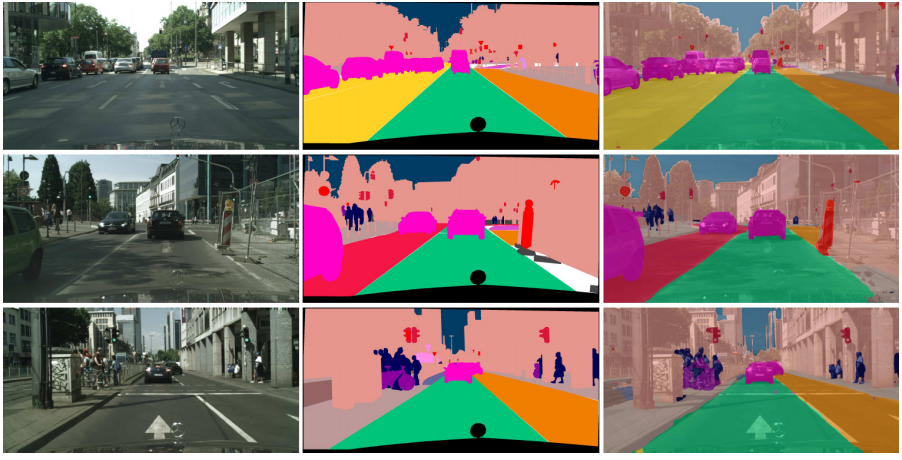
Joint Semantic Segmentation and 3D Reconstruction



- Joint semantic segmentation and monocular multi-view 3D reconstruction

Kundu, Li, Dellaert, Li and Rehg: Joint Semantic Segmentation and 3D Reconstruction from Monocular Video. ECCV, 2014.

Driving Corridor Prediction



- Fine-grained semantic segmentation (ego/parallel/opposite lanes)

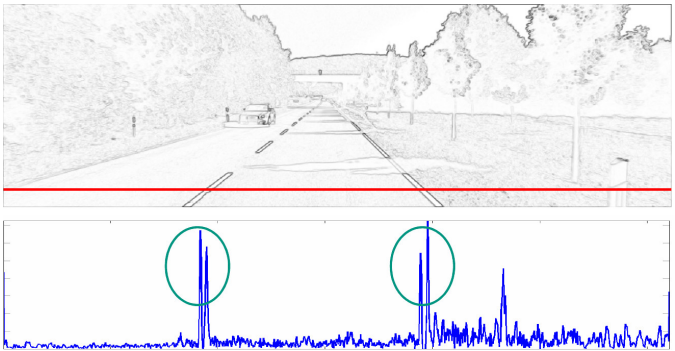
Meyer, Salscheider, Orzechowski and Stiller: Deep Semantic Lane Segmentation. IROS, 2018.

8.3
Lane Marking Detection

Detection of Lane Markings

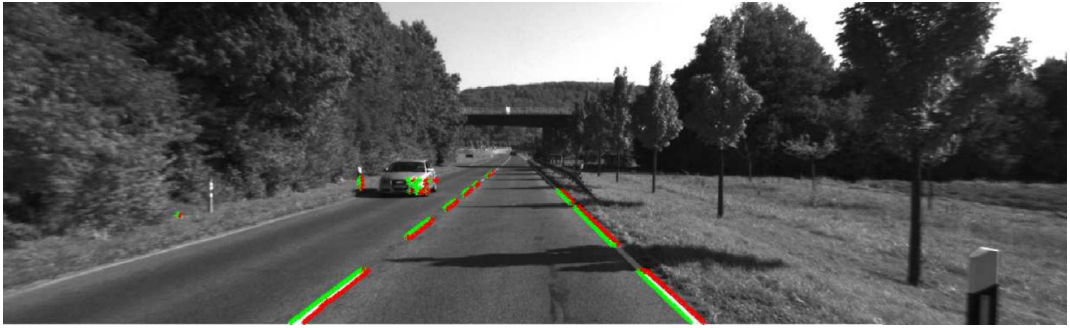
- Gradient Image (e.g., convolution with horizontal kernel $k = [-1 \;\; 0 \;\; +1]$)
- Alternatives: steerable filters, templates, learned features

Detection of Lane Markings



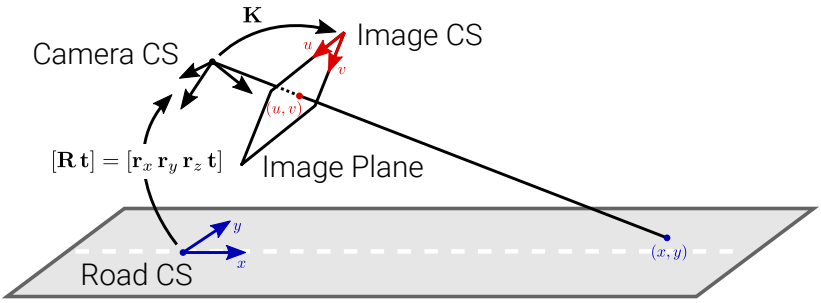
- Search for large gradients along each image row
- If depth available: filter points not on ground plane (e.g., by plane fitting)
- Remove isolated points or points for which no opposite gradient exists in vicinity

Detection of Lane Markings



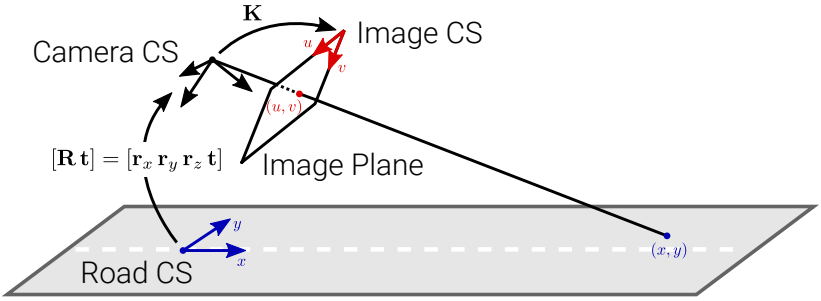
- Green: detected left boundary of lane marking
- Red: detected right boundary of lane marking
- False positives can be rejected using heuristics or robust fitting (e.g., RANSAC)
- Note: not all lane markings detected!

Inverse Perspective Mapping



- IPM maps each pixel on the image plane to the respective ground plane point
- IPM is only possible if either per pixel depth or ground model (e.g., plane) known
- How to obtain ground plane (i.e., extrinsics $[\mathbf{R}\ \mathbf{t}]$)?
- Assume fixed plane or fit plane/curve surface to depth measurements

Inverse Perspective Mapping



Given $[\mathbf{R}\ \mathbf{t}]$, a pixel in the image can be mapped to the ground plane:

$$\begin{bmatrix} u \\ v \\ 1 \end{bmatrix} \propto \mathbf{K} \begin{bmatrix} \mathbf{R} & \mathbf{t} \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix} \overset{z=0}{\Rightarrow} \begin{bmatrix} u \\ v \\ 1 \end{bmatrix} \propto \mathbf{K} \begin{bmatrix} \mathbf{r}_x & \mathbf{r}_y & \mathbf{t} \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} \Rightarrow \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} \propto \begin{bmatrix} \mathbf{r}_x & \mathbf{r}_y & \mathbf{t} \end{bmatrix}^{-1} \mathbf{K}^{-1} \begin{bmatrix} u \\ v \\ 1 \end{bmatrix}$$

Inverse Perspective Mapping: Example

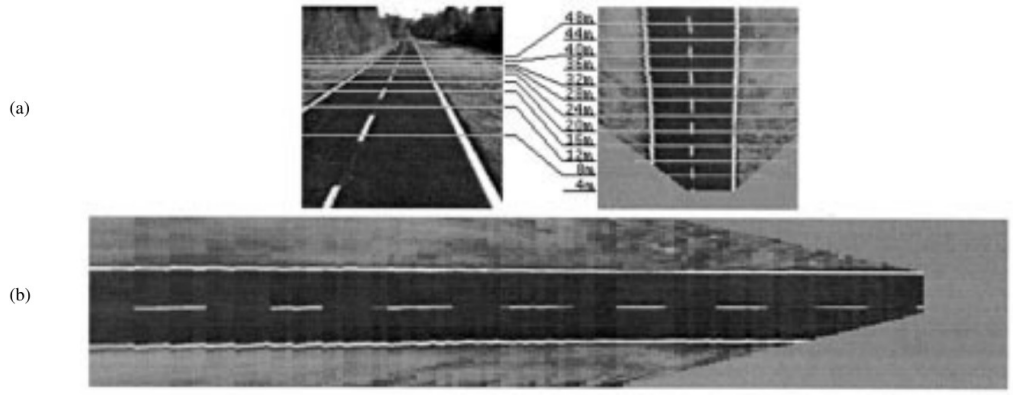
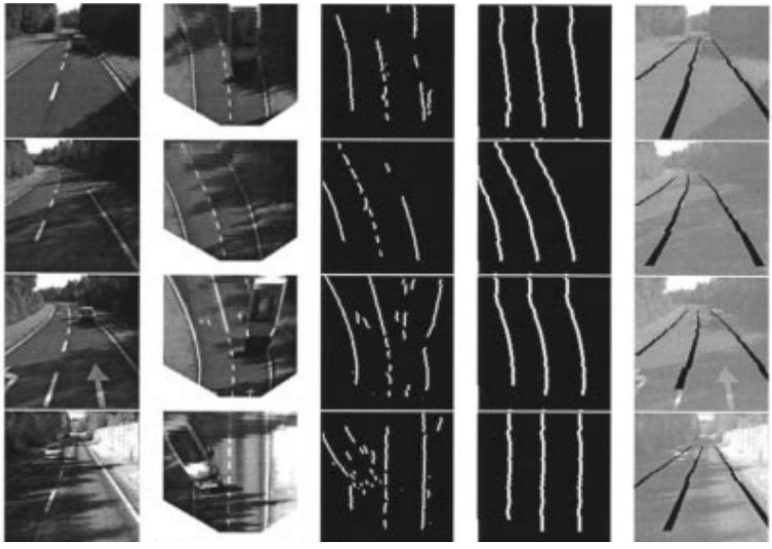


Fig. 8. (a) Horizontal calibration of the MOB-LAB vision system. (b) Rotated version of the remapped image considering an aspect ratio of 1 : 1.

Bertozzi and Broggi: GOLD: a parallel real-time stereo vision system for generic obstacle and lane detection. TIP, 1998.

Inverse Perspective Mapping: Example



Bertozzi and Broggi: GOLD: a parallel real-time stereo vision system for generic obstacle and lane detection. TIP, 1998.

Parametric Lane Marking Estimation

From Pixels to Curves:

- Lane marking detection alone is not sufficient
- Only returns a set of pixels in image or road plane space
- In order to be useful for navigation, this needs to be transformed into a more semantically meaningful parametric model
- Often low-dimensional parametric models (lines, polynomials, bezier curves, splines) are used

Remarks:

- This is a multi-model fitting problem (1 model per lane marking)
- Outliers must be handled (model must be robust to noise)

Parametric Lane Marking Estimation

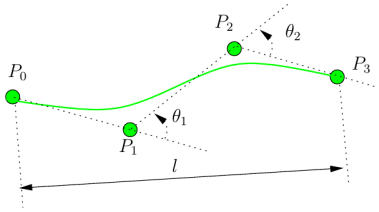


Fig. 7. Spline score computation.

Algorithm 1 RANSAC Spline Fitting

for $i = 1$ to $numIterations$ **do**
  $points$=getRandomSample()
  $spline$=fitSpline($points$)
  $score$=computeSplineScore($spline$)
  if $score > bestScore$ **then**
    $bestSpline = spline$
  end if
end for

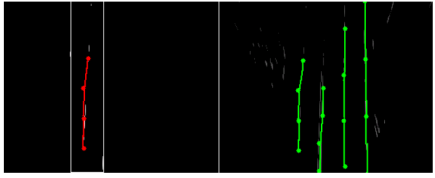


Fig. 8. RANSAC Spline fitting. Left: one of four windows of interest (white) obtained from previous step with detected spline (red). Right: the resulting splines (green) from this step

Fig. 10. Post-processing splines. Left: splines before post-processing in blue. Right: splines after post-processing in green. They appear longer and localized on the lanes.

Aly: Real time detection of lane markers in urban streets. IV, 2008.

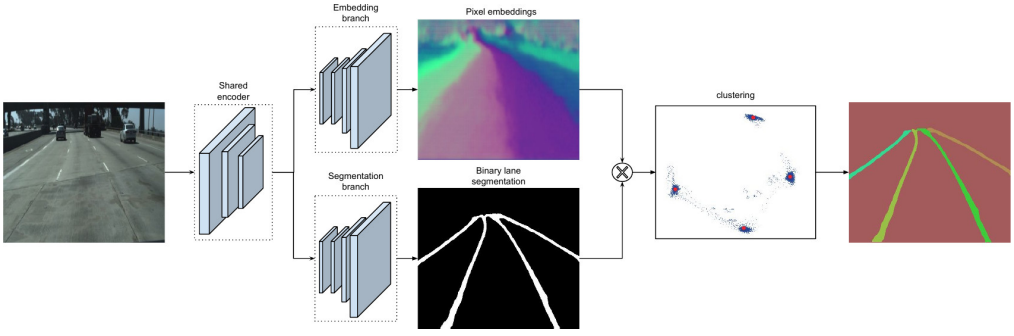
Parametric Lane Marking Estimation

Aly: Real time detection of lane markers in urban streets. IV, 2008.

End-to-End Lane Marking Detection

How to detect multiple lanes with deep learning?

- Problem: Standard cross-entropy loss (used, e.g., for segmentation) not enough
- Solution: Predict per-pixel feature embeddings and cluster them in feature space



Neven et al.: Towards End-to-End Lane Detection: An Instance Segmentation Approach. IV, 2018.

End-to-End Lane Marking Detection

Loss Function:

$$L_{var} = \frac{1}{C} \sum_{c=1}^{C} \frac{1}{N_c} \sum_{i=1}^{N_c} \left[\|\mu_c - x_i\| - \delta_{\mathrm{v}} \right]_+^2$$

$$L_{dist} = \frac{1}{C(C-1)} \sum_{c_A=1}^{C} \sum_{c_B=1, c_A \neq c_B}^{C} \left[\delta_{\mathrm{d}} - \|\mu_{c_A} - \mu_{c_B}\| \right]_+^2$$

- Clustering loss
- Learns an **embedding** where pixels
  - .. belonging to the same lane are pulled together
  - .. belonging to different lanes are pushed apart

Neven et al.: Towards End-to-End Lane Detection: An Instance Segmentation Approach. IV, 2018.

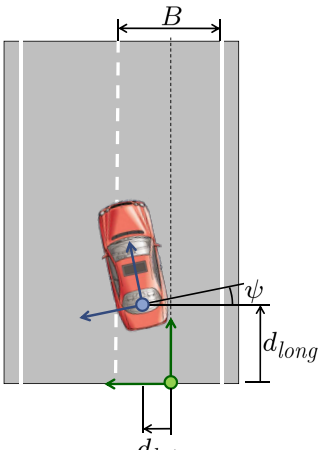
End-to-End Lane Marking Detection: Results

Fig. 5. Visual results. *Top row*: ground-truth lane points. *Middle row*: LaneNet output. *Bottom row*: final lane predicts after lane fitting.

Neven et al.: Towards End-to-End Lane Detection: An Instance Segmentation Approach. IV, 2018.

8.4
Lane Detection

Model for Straight Lanes



Vehicle coordinates: rear axle center

We consider the vehicle in blue

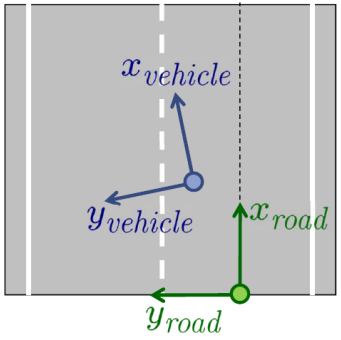
Geometric model: (straight=highway scenario)

- Can we fit a single model for the entire lane?
- Lane width $B \in \mathbb{R}^+$
- Lateral vehicle offset wrt. centerline $d_{lat} \in \mathbb{R}$
- Longitudinal position $d_{long} \in \mathbb{R}$
- Yaw angle $\psi \in [-\pi, \pi]$

State:

$$\mathbf{s} = \begin{bmatrix} B \\ d_{long} \\ d_{lat} \\ \psi \end{bmatrix}$$

Model for Straight Lanes



Transformation vehicle → road coordinates:

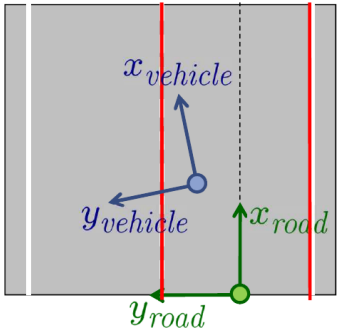
$$\begin{bmatrix} x_{road} \\ y_{road} \end{bmatrix} = \begin{bmatrix} d_{long} \\ d_{lat} \end{bmatrix} + \begin{bmatrix} \cos(\psi) & -\sin(\psi) \\ \sin(\psi) & \cos(\psi) \end{bmatrix} \begin{bmatrix} x_{vehicle} \\ y_{vehicle} \end{bmatrix}$$

Transformation road → vehicle coordinates:

$$\begin{bmatrix} x_{vehicle} \\ y_{vehicle} \end{bmatrix} = \begin{bmatrix} \cos(\psi) & \sin(\psi) \\ -\sin(\psi) & \cos(\psi) \end{bmatrix} \left(\begin{bmatrix} x_{road} \\ y_{road} \end{bmatrix} - \begin{bmatrix} d_{long} \\ d_{lat} \end{bmatrix} \right)$$

Slide Credits: Martin Lauer

Model for Straight Lanes



Position of lane markings:

$$\begin{bmatrix} x_{road} \\ y_{road} \end{bmatrix} = \begin{bmatrix} \ell \\ \pm\frac{B}{2} \end{bmatrix}$$

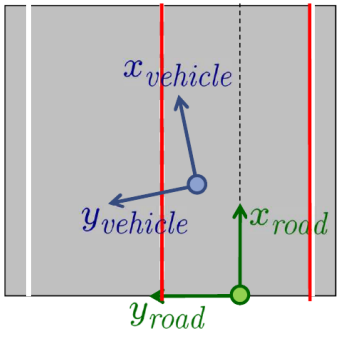
- with path length $\ell \in \mathbb{R}$ and lane width $B \in \mathbb{R}^+$

In vehicle coordinate system:

$$\begin{bmatrix} x_{vehicle} \\ y_{vehicle} \end{bmatrix} = \begin{bmatrix} \cos(\psi) & \sin(\psi) \\ -\sin(\psi) & \cos(\psi) \end{bmatrix} \begin{bmatrix} \ell - d_{long} \\ \pm\frac{B}{2} - d_{lat} \end{bmatrix}$$

Slide Credits: Martin Lauer

Model for Straight Lanes



Repeated from last slide:

$$\begin{bmatrix} x_{vehicle} \\ y_{vehicle} \end{bmatrix} = \begin{bmatrix} \cos(\psi) & \sin(\psi) \\ -\sin(\psi) & \cos(\psi) \end{bmatrix} \begin{bmatrix} \ell - d_{long} \\ \pm\frac{B}{2} - d_{lat} \end{bmatrix}$$

Eliminating ℓ and assuming $\psi \approx 0$ yields:

$$y_{vehicle} = -\psi \cdot x_{vehicle} \pm \frac{B}{2} - d_{lat}$$

Slide Credits: Martin Lauer

Model for Straight Lanes

Results in simple regression problem per frame:

- Given N^L points on the left marking (x_i^L, y_i^L) and
  Given N^R points on the right marking (x_j^R, y_j^R)
- Minimize:

$$
\begin{aligned}
B^*, d_{lat}^*, \psi^* &= \underset{B, d_{lat}, \psi}{\operatorname{argmin}} \sum_i (e_i^L)^2 + \sum_j (e_j^R)^2 \\
e_i^L &= \left(-\psi \cdot x_i^L + \frac{B}{2} - d_{lat} \right) - y_i^L \\
e_j^R &= \left(-\psi \cdot x_j^R - \frac{B}{2} - d_{lat} \right) - y_j^R
\end{aligned}
$$

- Closed form solution:

$$
\begin{bmatrix}
\frac{1}{4}(N^L + N^R) & \frac{1}{2}(-N^L + N^R) & \frac{1}{2}(-\sum x_i^L + \sum x_j^R) \\
\frac{1}{2}(-N^L + N^R) & N^L + N^R & \sum x_i^L + \sum x_j^R \\
\frac{1}{2}(-\sum x_i^L + \sum x_j^R) & \sum x_i^L + \sum x_j^R & \sum (x_i^L)^2 + \sum (x_j^R)^2
\end{bmatrix}
\begin{bmatrix} B \\ d_{lat} \\ \psi \end{bmatrix}
=
\begin{bmatrix}
\frac{1}{2}(\sum y_i^L - \sum y_j^R) \\
-\sum y_i^L - \sum y_j^R \\
-\sum x_i^L y_i^L - \sum x_j^R y_j^R
\end{bmatrix}
$$

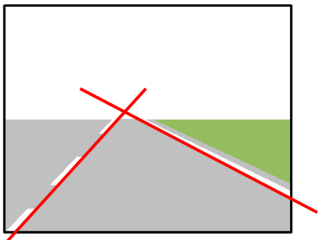
Model for Straight Lanes

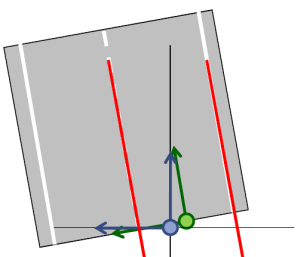
Summary:

- Detect lane markings in camera image
- Transform position of markings into vehicle coordinates via IPM
- Estimate pose parameters and lane width

Note:

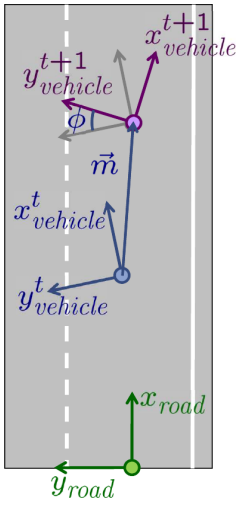
- Model parameter d_{long} is not observable, thus set to 0 in this simple model





8.5
Lane Tracking

Model for Straight Lanes: Incremental Localization



Incremental localization: (=parameter tracking over time)

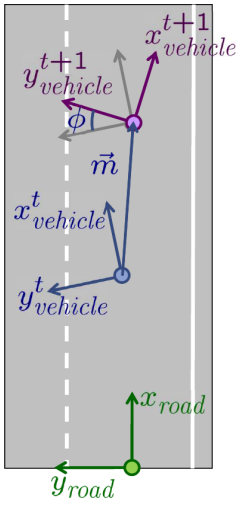
- **State:**

$$\mathbf{s} = \begin{bmatrix} B & d_{long} & d_{lat} & \psi \end{bmatrix}^T$$

- **State transition:**

$$\begin{aligned}
\begin{bmatrix} d^{t+1}_{long} \\ d^{t+1}_{lat} \end{bmatrix} &= \begin{bmatrix} d^{t}_{long} \\ d^{t}_{lat} \end{bmatrix} + \begin{bmatrix} \cos(\psi^t) & -\sin(\psi^t) \\ \sin(\psi^t) & \cos(\psi^t) \end{bmatrix} \mathbf{m} \\
&\approx \begin{bmatrix} d^{t}_{long} \\ d^{t}_{lat} \end{bmatrix} + \begin{bmatrix} 1 & -\psi^t \\ \psi^t & 1 \end{bmatrix} \mathbf{m} \\
\psi^{t+1} &= \psi^t + \phi \\
B^{t+1} &= B^t \qquad \text{(constant road width)}
\end{aligned}$$

Model for Straight Lanes: Incremental Localization



Incremental localization: (=parameter tracking over time)

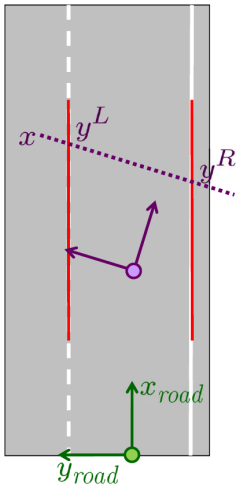
- **State:**

$$\mathbf{s} = \begin{bmatrix} B & d_{long} & d_{lat} & \psi \end{bmatrix}^T$$

- **State transition:**

$$\mathbf{s}^{t+1} = \underbrace{\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & -m_y \\ 0 & 0 & 1 & m_x \\ 0 & 0 & 0 & 1 \end{bmatrix}}_{\mathbf{A}^t} \mathbf{s}^t + \underbrace{\begin{bmatrix} 0 \\ m_x \\ m_y \\ \phi \end{bmatrix}}_{\mathbf{u}^t}$$

Model for Straight Lanes: Incremental Localization

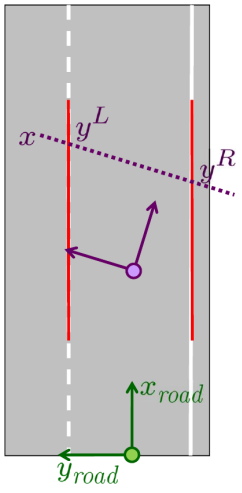


Incremental localization: (=parameter tracking over time)

- **Observations:**
  - N^L points on left marking (x_i^L, y_i^L)
  - N^R points on right marking (x_j^R, y_j^R)
- y^L and y^R can be determined from $\mathbf{s}$ and x:

$$y^L = -\psi \cdot x + \frac{B}{2} - d_{lat}$$
$$y^R = -\psi \cdot x - \frac{B}{2} - d_{lat}$$

Model for Straight Lanes: Incremental Localization

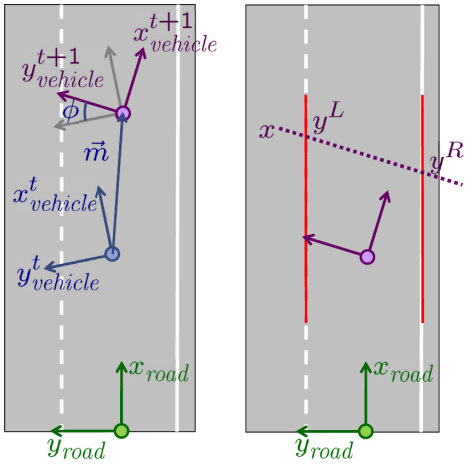


Incremental localization: (=parameter tracking over time)

- **Observations:**
  - N^L points on left marking (x_i^L, y_i^L)
  - N^R points on right marking (x_j^R, y_j^R)
- y^L and y^R can be determined from $\mathbf{s}$ and x:

$$\underbrace{\begin{bmatrix} y_1^L \\ \vdots \\ y_{N^L}^L \\ y_1^R \\ \vdots \\ y_{N^R}^R \end{bmatrix}}_{=\mathbf{y}^{t+1}} = \underbrace{\begin{bmatrix} \frac{1}{2} & 0 & -1 & -x_1^L \\ \vdots & \vdots & \vdots & \vdots \\ \frac{1}{2} & 0 & -1 & -x_{N^L}^L \\ -\frac{1}{2} & 0 & -1 & -x_1^R \\ \vdots & \vdots & \vdots & \vdots \\ -\frac{1}{2} & 0 & -1 & -x_{N^R}^R \end{bmatrix}}_{=\mathbf{C}^{t+1}} \underbrace{\begin{bmatrix} B \\ d_{long} \\ d_{lat} \\ \psi \end{bmatrix}}_{=\mathbf{s}^{t+1}}$$

Model for Straight Lanes: Incremental Localization



State space representation:

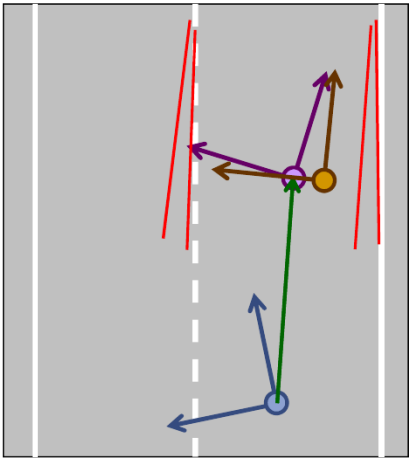
$$\mathbf{s}^{t+1} = \mathbf{A}^t \mathbf{s}^t + \mathbf{u}^t$$
$$\mathbf{y}^{t+1} = \mathbf{C}^{t+1} \mathbf{s}^{t+1}$$

Estimation:

- Linear dynamic model
- Linear measurement model
- Assuming Gaussian noise
- $\Rightarrow$ Kalman filter [Kalman, 1960]

Slide Credits: Martin Lauer

Model for Straight Lanes: Incremental Localization

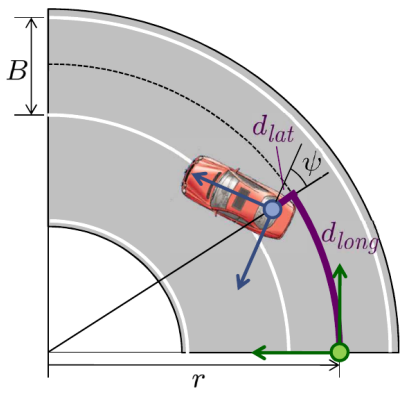


Algorithm:

- **Detection Step**
  Detect road markings in image
- **Inverse Perspective Mapping**
  Transform detected road markings into vehicle coordinates
- **Prediction Step**
  Determine how the vehicle is moving, e.g., using visual odometry
- **Innovation Step**
  Update state wrt. observed markings

Slide Credits: Martin Lauer

Model for Circular Lanes



Geometric model:

- Lane width $B \in \mathbb{R}^+$
- Signed radius $r \in \mathbb{R}$ or curvature $\kappa = \frac{1}{r}$
- Lateral vehicle offset $d_{lat} \in \mathbb{R}$
- Longitudinal position $d_{long} \in \mathbb{R}$
- Yaw angle $\psi \in [-\pi, \pi]$

State:

$$\mathbf{s} = \begin{bmatrix} B & r & d_{long} & d_{lat} & \psi \end{bmatrix}^T$$

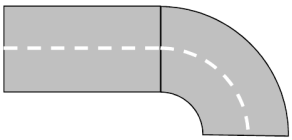
Model becomes non-linear:

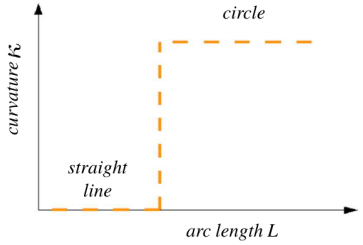
- Requires EKF, UKF or Particle Filter

Slide Credits: Martin Lauer

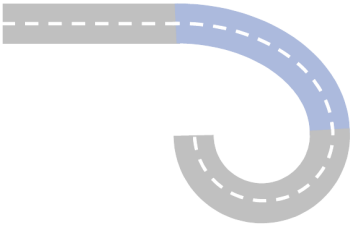
Circular trajectories are not enough ..

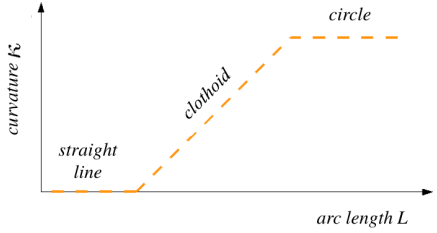
Circle-Straight Combination





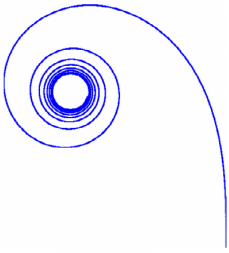
Clothoid

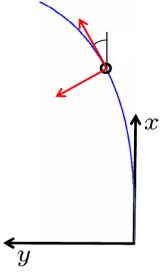




Slide Credits: Martin Lauer

Clothoid Model





Clothoid:

- Curve whose curvature linearly depends on the arc length:

$$\frac{1}{r(\ell)} = \kappa(\ell) = \kappa_0 + \kappa_1 \ell$$

- Yaw angle:

$$\psi(\ell) = \int_0^\ell \kappa(\lambda) d\lambda = \kappa_0 \ell + \frac{1}{2}\kappa_1 \ell^2$$

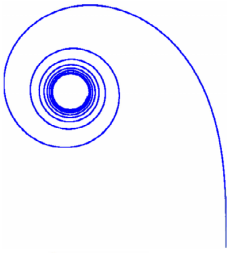
- Position:

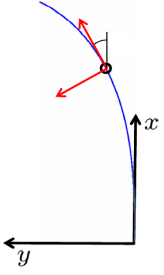
$$x(\ell) = \int_0^\ell \cos(\psi(\lambda))\, d\lambda$$

$$y(\ell) = \int_0^\ell \sin(\psi(\lambda))\, d\lambda$$

Slide Credits: Martin Lauer

Clothoid Model





Clothoid:

- For small yaw angles ψ we obtain:

$$x(\ell) = \int_0^\ell \cos(\psi(\lambda))\, d\lambda \approx \int_0^\ell 1\, d\lambda$$
$$y(\ell) = \int_0^\ell \sin(\psi(\lambda))\, d\lambda \approx \int_0^\ell \psi(\lambda)\, d\lambda$$

  with $\psi(\lambda) = \kappa_0\lambda + \frac{1}{2}\kappa_1\lambda^2$

- Thus, we obtain a 3rd order polynomial:

$$x(\ell) \approx \ell$$
$$y(\ell) \approx \frac{1}{2}\kappa_0\ell^2 + \frac{1}{6}\kappa_1\ell^3$$

Recursive 3-D Road and Relative Ego-State Recognition

***Abstract*—The general problem of recognizing both horizontal and vertical road curvature parameters while driving along the road has been solved recursively. A differential geometry representation decoupled for the two curvature components has been selected. Based on the planar solution of [7] and its refinements, a simple spatio-temporal model of the driving process allows us to take both spatial and temporal constraints into account effectively. The estimation process determines nine road and vehicle state parameters recursively at 25 Hz (40 ms) using four Intel 80286 and one 386 microprocessor. Results with the test vehicle (VaMoRs), which is a 5-ton van, are given for a hilly country road.**

[PAMI, 1992]

- Seminal paper by self-driving pioneer Ernst Dickmanns (Bundeswehr Univ. Munich)
- Visual recursive state estimation
- Uses clothoid model in 3D

Ernst D. Dickmanns

Dickmanns and Mysliwetz: Recursive 3-D road and relative ego-state recognition. PAMI, 1992.

Recursive 3-D Road and Relative Ego-State Recognition

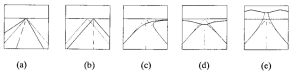


Fig. 2. Individual influences of the lateral vehicle state and of road parameters on the road image: (a) Lateral offset y_c; (b) heading offset ψ_c; (c) horizontal curvature right; (d) downward vertical curvature; (e) upward curvature.

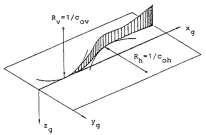


Fig. 3. Spatial road segment with horizontal and vertical curvature.

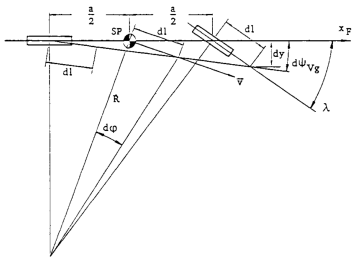


Fig. 8. Steering kinematics.

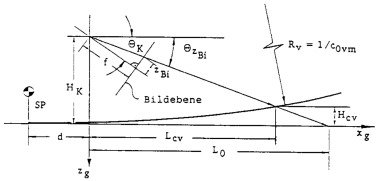


Fig. 6. Vertical road and mapping geometry.

- Models horizontal and vertical road curvature (left)
- Considers dynamic bicycle motion model (center)
- Uses a projection-based measurement model instead of IPM (right)

Dickmanns and Mysliwetz: Recursive 3-D road and relative ego-state recognition. PAMI, 1992.

Summary

- Road and lane detection are important for ADAS and self-driving
- They allow to localize locally with respect to the road without requiring a map
- Multiple different representations exist with advantages/disadvantages
- We discussed road segmentation and lane marking detection
- We have seen how lanes can be estimated from lane markings
- We discussed how we can incrementally track lanes over time
- We haven't addresses geometric freespace estimation in this lecture
- We will discuss this topic in the next lecture in the context of 3D reconstruction