Ego-Lane Estimation by Modeling Lanes and Sensor Failures

HIGHWAY-LIKE SCENARIOS

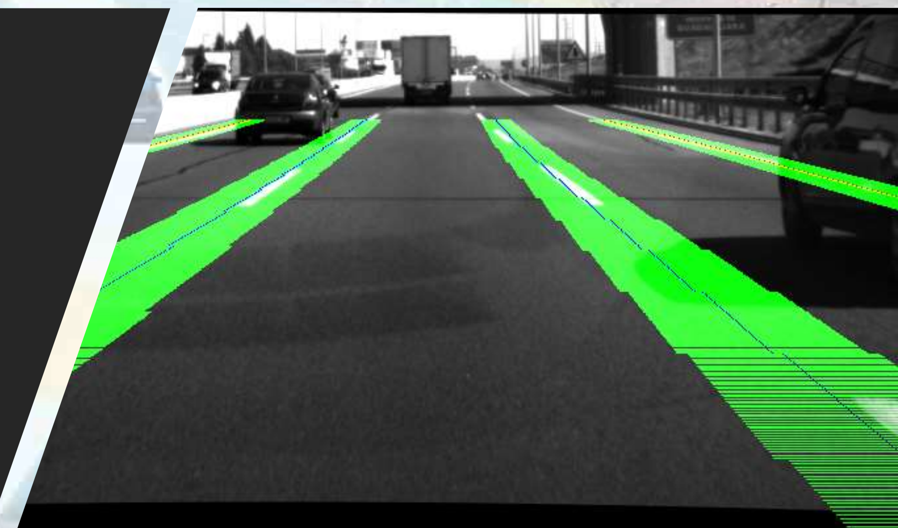
A.L.Ballardini D.Cattaneo D.G.Sorrenti *Università degli Studi di Milano - Bicocca, Italy*

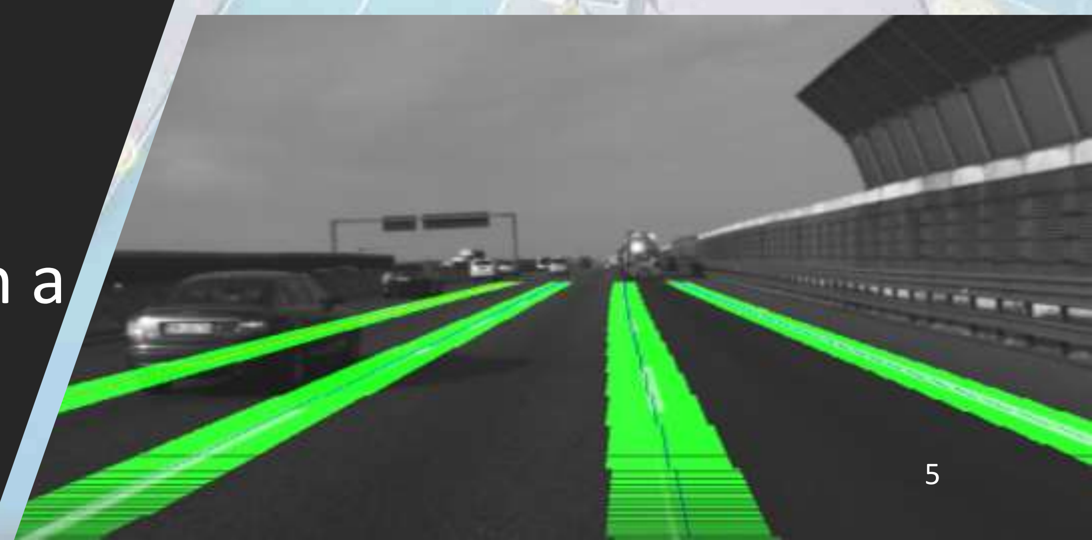
R.Izquierdo, I.Parra, M.A. Sotelo *University of Alcala, Alcalá de Henares, Spain*

Tuesday, October 17th, 2017

Proposed Approach Overview

- Given a rough localization estimate (road segment)
- Leverage a line detector & tracker (we do not investigate this topic)
- Exploit a **H**idden **M**arkov **M**odel with a *transient failure model*
- Infer: vehicle’s ego-lane in a probabilistic fashion






From Road *Lines* to Road *Lanes*

In an ideal scenario, we could directly infer the vehicle ego-lane from the road lines

Possible issues

- Cluttering elements (other cars)
- Faded Road Markings
- Illumination Issues

Limit the uncertainties

Given partial line detections, we can exploit their distances (lateral offset *wrt* vehicle) to limit the ego-lane uncertainty

Only leftmost line detected due to illumination issues

Lanes compatible w.r.t. line distance

01

Exploit *consecutive* yet partial observations *over time*

02

Exploit a HMM approach with 2*n-states* corresponding to the *n-lanes*

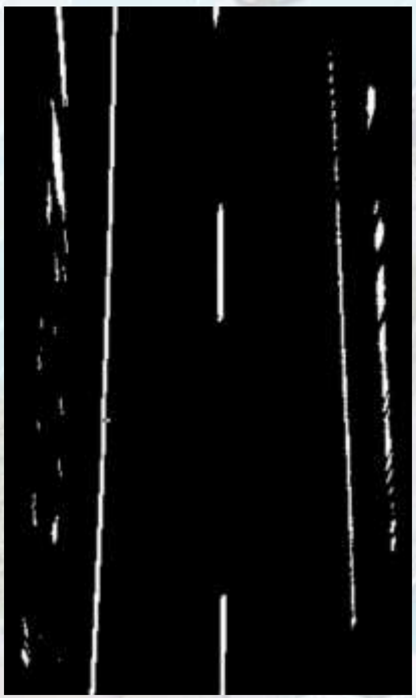
03

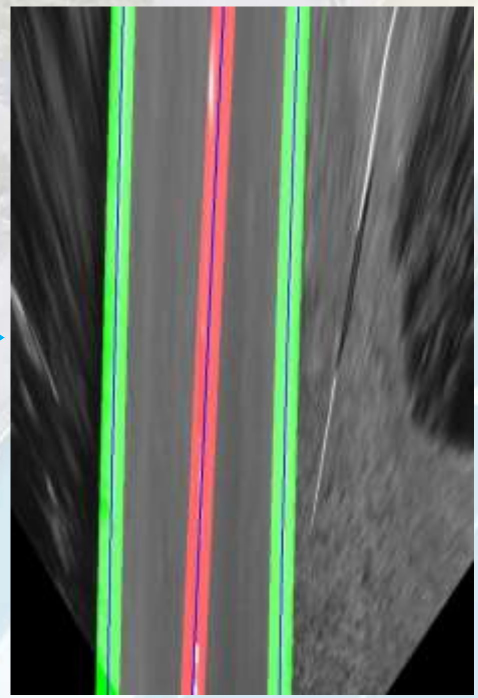
“n” is the number of lanes retrieved from OpenStreetMap

Track the Vehicle ego-lane

Line Detector & Tracker Interface

$$\text{output} = \{\ \text{line}_1, <\text{property}_{1\ldots k}> \\ \text{line}_2, <\text{property}_{1\ldots k}> \\ \ldots\ \}$$



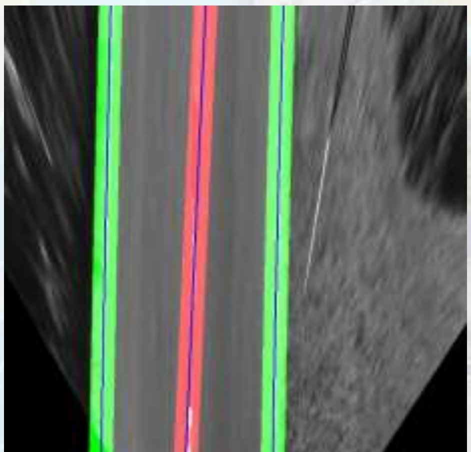


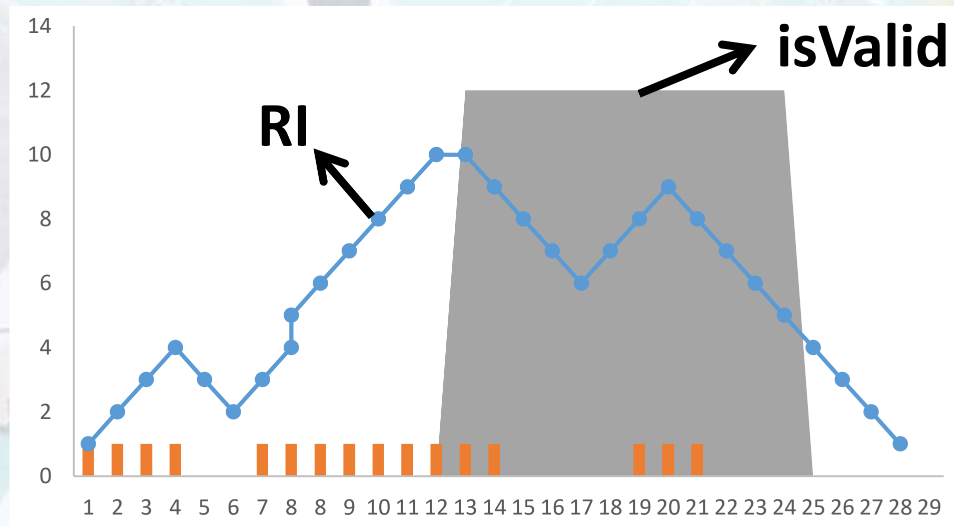
1. **Find Contours in image (BEV)**
2. **Fit Geometric Primitives on Contours (clothoids, polylines…)**
3. **Track Geometric Primitives**
    1. **lateral offset from vehicle [m]**
    2. **is line continuous? (y/n)**

Temporal Line Reliability

Reliability Index

line detection ratio over the last k-frames





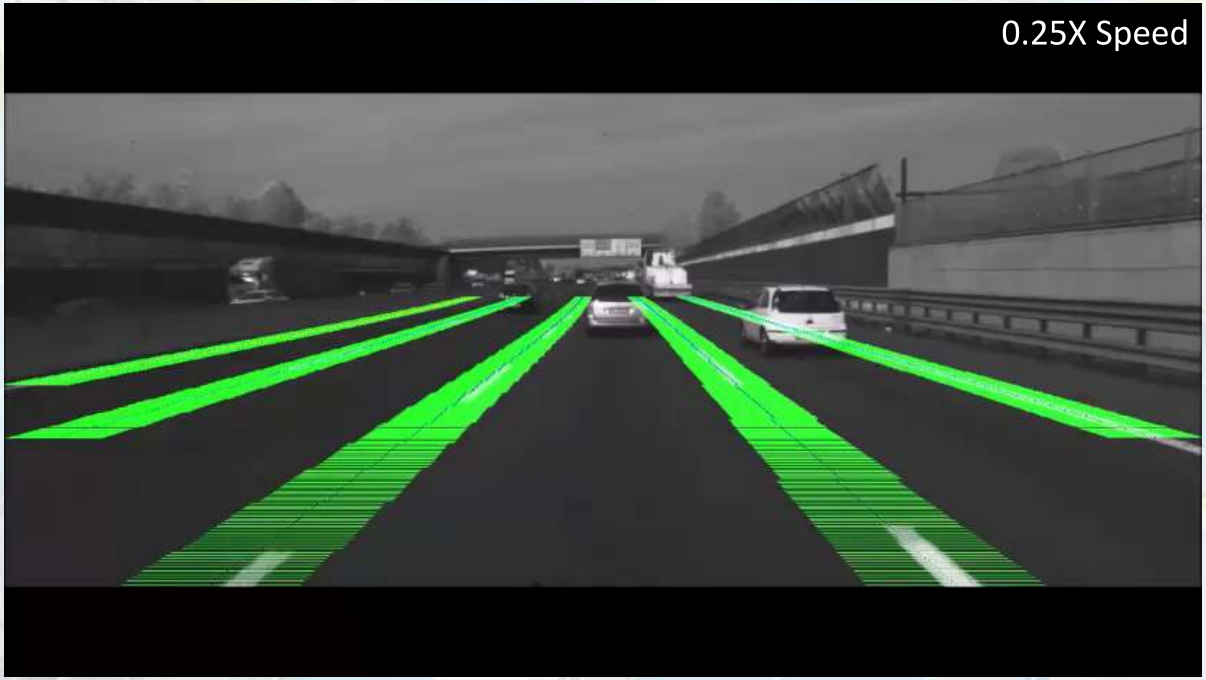
Reliability Index of line1 8/**10**, isValid, continuous

Reliability Index of line2 5/**10**

Reliability Index of line3 10/**10**, isValid

We add the Reliability Index and the isValid properties to the Line properties vector

Sensor Output
(Line Detector & Tracker + Line Reliability)



These results can be used to infer the ego-lane using the simple geometric considerations shown before

The HMM Model
State Definition

HMM ($n, \sigma_1, \sigma_2, P_1, P_2, BV, w$)

- *n* — *<u>number of lanes</u>, retrieved from OpenStreetMap*
- σ_1, σ_2 — *parameters used for the lane transition model generation*
- P_1, P_2 — *How likely to stay in SensorOK/SensorBad state (prediction phase)*
- *BV* — *Bonus Value for continuous lines (gives richer information)*
- *w* — *Inertia used in the Sensor Matrix to propagate the SensorBad state*

The HMM allow us to represent and track over time the probability of being on each lane and having either a *<u>properly operating</u>* or a *<u>faulty</u>* Sensor

$$X_t = \langle Lane_1SensorOK \dots Lane_nSensorOK\ ,\ Lane_1SensorBad \dots Lane_nSensorBad \rangle$$

$$|X_t| = |Sensor| * |ego\_lane|$$

The HMM Tracking Model
State Prediction

$$X_t = \langle Lane_{1\ldots n}\ SensorOK\ ;\ Lane_{1\ldots n}\ SensorBad \rangle$$

$$\overline{X_{t+1}} = X_t \cdot StateTransitionMatrix$$

BTM	*Lane*$_1$	*Lane*$_2$	*Lane*$_{\ldots}$	*Lane*$_n$
Lane$_1$	$\mathcal{N}(1,1,\sigma)$	$\mathcal{N}(2,1,\sigma)$	…	$\mathcal{N}(n,1,\sigma)$
Lane$_2$	$\mathcal{N}(1,2,\sigma)$	…	…	…
Lane$_{\ldots}$	…	…	…	…
Lane$_4$	$\mathcal{N}(1,n,\sigma)$	…	…	$\mathcal{N}(n,n,\sigma)$

The State Transition Matrix is built using a Basic Transition Matrix, which in turn is based on the following consideration:

the probability of moving to another lane is normally distributed, the average is on the current lane, σ is a model parameter

The HMM Tracking Model

State Prediction

$$\overline{X_{t+1}} = X_t \cdot StateTransitionMatrix$$

BTM	$Lane_1$	$Lane_2$	$Lane_{...}$	$Lane_n$
$Lane_1$	$\mathcal{N}(1,1,\sigma)$	$\mathcal{N}(2,1,\sigma)$	…	$\mathcal{N}(n,1,\sigma)$
$Lane_2$	$\mathcal{N}(1,2,\sigma)$	…	…	…
$Lane_{...}$	…	…	…	…
$Lane_4$	$\mathcal{N}(1,n,\sigma)$	…	…	$\mathcal{N}(n,n,\sigma)$

BTM	$Lane_1$	$Lane_2$	$Lane_{...}$	$Lane_n$
$Lane_1$	$\mathcal{N}(1,1,\sigma)$	$\mathcal{N}(2,1,\sigma)$	…	$\mathcal{N}(n,1,\sigma)$
$Lane_2$	$\mathcal{N}(1,2,\sigma)$	…	…	…
$Lane_{...}$	…	…	…	…
$Lane_4$	$\mathcal{N}(1,n,\sigma)$	…	…	$\mathcal{N}(n,n,\sigma)$

$$STM_{2n\times 2n} = \left(\begin{array}{c|c} A_{n\times n} & B_{n\times n} \\ \hline C_{n\times n} & D_{n\times n} \end{array}\right)$$

BTM	$Lane_1$	$Lane_2$	$Lane_{...}$	$Lane_n$
$Lane_1$	$\mathcal{N}(1,1,\sigma)$	$\mathcal{N}(2,1,\sigma)$	…	$\mathcal{N}(n,1,\sigma)$
$Lane_2$	$\mathcal{N}(1,2,\sigma)$	…	…	…
$Lane_{...}$	…	…	…	…
$Lane_4$	$\mathcal{N}(1,n,\sigma)$	…	…	$\mathcal{N}(n,n,\sigma)$

BTM	$Lane_1$	$Lane_2$	$Lane_{...}$	$Lane_n$
$Lane_1$	$\mathcal{N}(1,1,\sigma)$	$\mathcal{N}(2,1,\sigma)$	…	$\mathcal{N}(n,1,\sigma)$
$Lane_2$	$\mathcal{N}(1,2,\sigma)$	…	…	…
$Lane_{...}$	…	…	…	…
$Lane_4$	$\mathcal{N}(1,n,\sigma)$	…	…	$\mathcal{N}(n,n,\sigma)$

- A $\rightarrow$ (Lane Transition from SensorOK to SensorOK) * P_1
- B $\rightarrow$ (Lane Transition from SensorOK to SensorBad) * $(1-P_1)$
- C $\rightarrow$ (Lane Transition from SensorBad to SensorOk) * $(1-P_2)$
- D $\rightarrow$ (Lane Transition from SensorBad to SensorBad) * P_2

The HMM Tracking Model

State Prediction

$$\overline{\boldsymbol{X_{t+1}}} = \boldsymbol{X_t} \cdot \boldsymbol{StateTransitionMatrix}$$

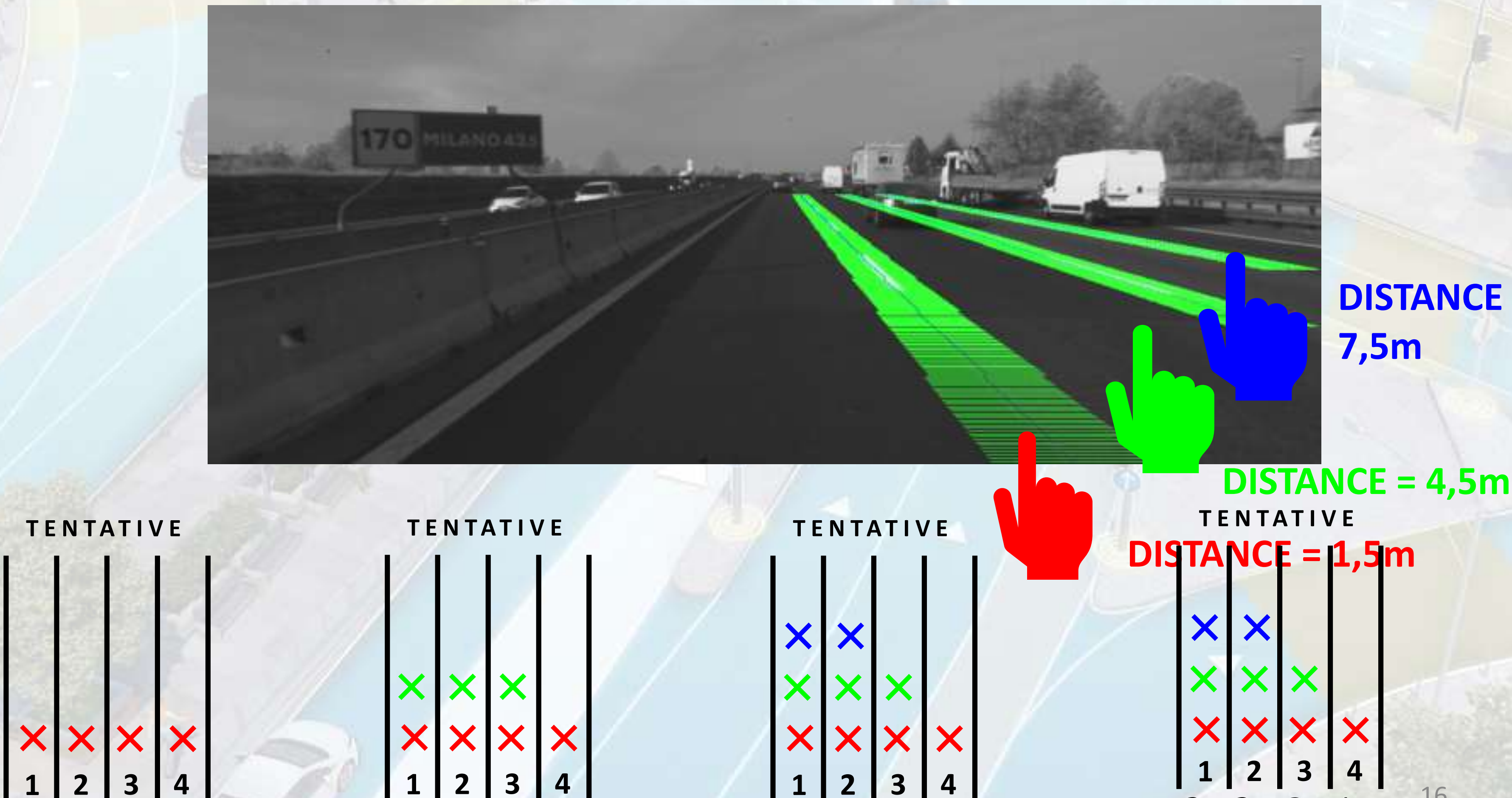
- A $\rightarrow$ (Lane Transition from SensorOK to SensorOK) * P_1
- B $\rightarrow$ (Lane Transition from SensorOK to SensorBad) * $(1-P_1)$
- C $\rightarrow$ (Lane Transition from SensorBad to SensorOk) * $(1-P_2)$
- D $\rightarrow$ (Lane Transition from SensorBad to SensorBad) * P_2

Idea:
the sensor mainly gives long runs of correct outputs, so P_1 is “large” (and $(1-P_1)$ is “small”)
when the sensor makes mistakes for a short period of time, so $(1-P_2)$ is “large” (and P_2 is “small”)

Updating the Belief 1

Counting Scheme

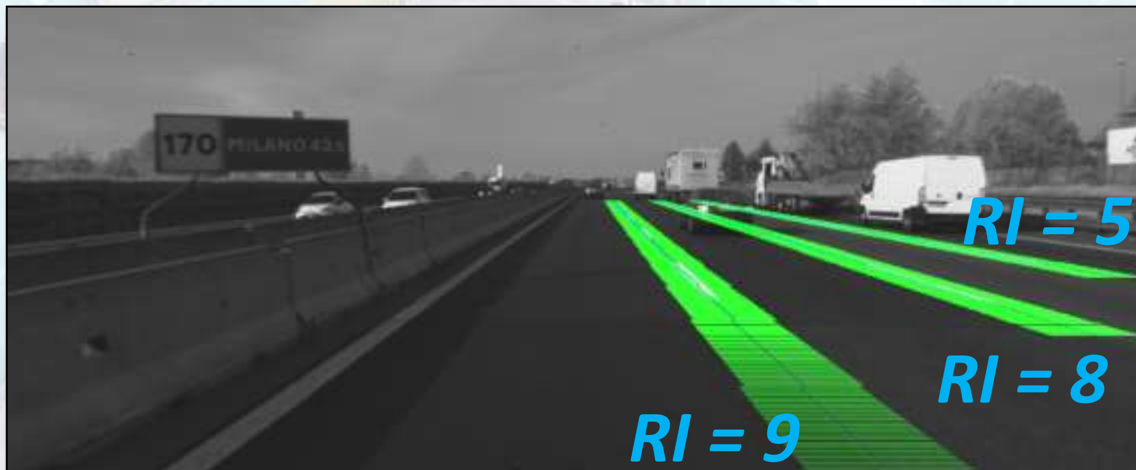
To update the belief exploiting the output of the Line Detector & Tracker we use an ad-hoc sensor model which uses the Line Properties (*LateralOffset*; *ReliabilityIndex*; *isValid*; *Continuous*)



P={ $^3/_9$ $^3/_9$ $^2/_9$ $^1/_9$ }

Updating the Belief 2
Exploiting the Sensor Reliability

To update the belief exploiting the output of the Line Detector & Tracker we use an ad-hoc sensor model which uses the Line Properties (Lateral Offset ;Reliability Index ; isContinuous)



$$SensorScoreOK = \frac{\sum_1^m isValid_i * RI_i}{10 * (n+1)} = \frac{9+8+5}{50} = 0.44$$

$$SensorScoreBad = 1 - 0.44 = 0.66$$

The normalizer equals the maximum number than RI can take times the maximum number of lines

Updating the Belief 3

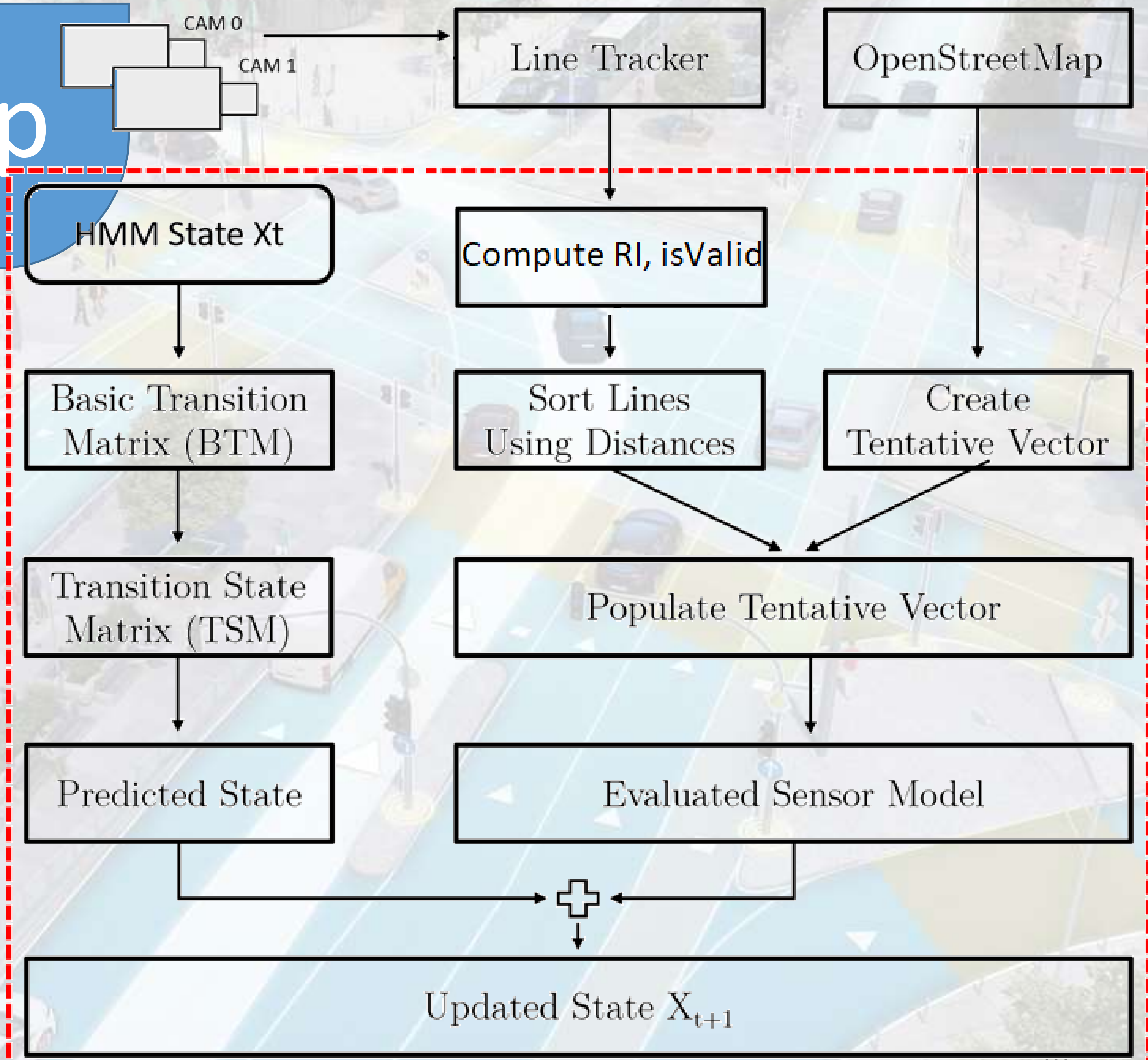
To correctly deal with either a *properly operating* or *faulty* sensor the HMM model includes different strategies for the two cases

$$S_1 = \overbrace{SensorScoreOK}^{\textit{Sensor Reliability}} \cdot \overbrace{tentative}^{\textit{Counting Scheme}}$$

$$S_2 = SensorScoreBad \cdot [(tentative \cdot w) + \overline{\boldsymbol{X}_{\boldsymbol{t}+\boldsymbol{1}}} \cdot (1 - w)]$$

$$Z = (S_1 \mid S_2)$$

Recap


CAM 0
CAM 1
Line Tracker
OpenStreetMap
HMM State Xt
Compute RI, isValid
Basic Transition Matrix (BTM)
Sort Lines Using Distances
Create Tentative Vector
Transition State Matrix (TSM)
Populate Tentative Vector
Predicted State
Evaluated Sensor Model
Updated State X_{t+1}

Experimental Configuration

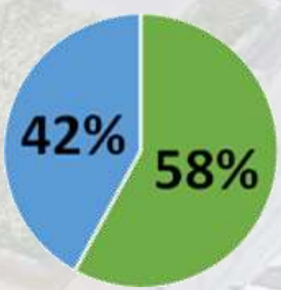
We verified the improvements of our model using two datasets recorded in real driving conditions.

Differently from KITTI datasets here we have hundreds of lane transitions

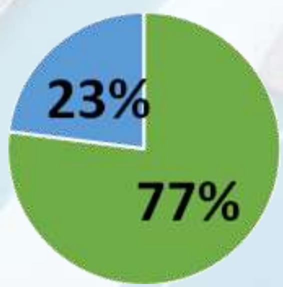
A4 Highway, Milan-Bergamo, Italy
4-Lanes Configuration

A2 Highway, Alcalá de Henares, Spain
3-Lanes Configuration

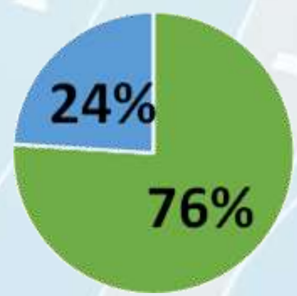
Detector Only



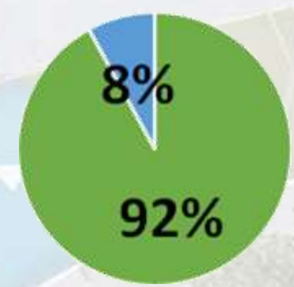
Our Model



Detector Only



Our Model



Wrong Lane Estimate *Correct Lane Estimate*

Results 1

Detected Ego Lane Dispersion

	Detectory Only	Our Model
Correct Lane	5276	6978
Offset 1	3744	2762
Offset 2	779	212
Offset 3	153	0

Line Detector Only

	1	2	3	4	Support	Recall
GT Lane 1	2230	320	21	3	2574	0.866
GT Lane 2	904	2005	275	16	3200	0.627
GT Lane 3	373	1666	927	5	2971	0.312
GT Lane 4	150	369	574	114	1207	0.094
Total	3657	4360	1797	138		
Precision	0.61	0.46	0.516	0.826		
F1 Score	0.7158	0.53	0.389	0.17		

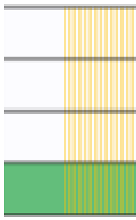
Proposed Model

	1	2	3	4	Support	Recall
GT Lane 1	2080	432	62	0	2574	0.808
GT Lane 2	246	2477	476	1	3200	0.774
GT Lane 3	13	871	2082	5	2971	0.701
GT Lane 4	0	136	732	339	1207	0.281
Total	2339	3916	3352	345		
Precision	0.889	0.633	0.621	0.983		
F1 Score	0.847	0.696	0.659	0.437		

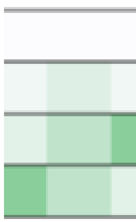
Support-Column: how many GT lanes. Total-Row: how many ego-vehicle detections over the *n-lane.*

Results 2

Ground Truth →



Line Detector & Tracker Only →



Proposed Model →

Thank you

Dataset will be available for download at http://www.ira.disco.unimib.it/ego-lane-estimation-by-modeling-lanesand-sensor-failures

or just scan qr code

Updated Version of the Paper also on our website soon

Results 2