Machine learning based automatic extrinsic calibration of an onboard monocular camera for driving assistance applications on smart mobile devices

Razvan Itu, Radu Danescu
Technical University of Cluj-Napoca, Romania
Image Processing and Pattern Recognition Research Center

12.09.2018

Objectives

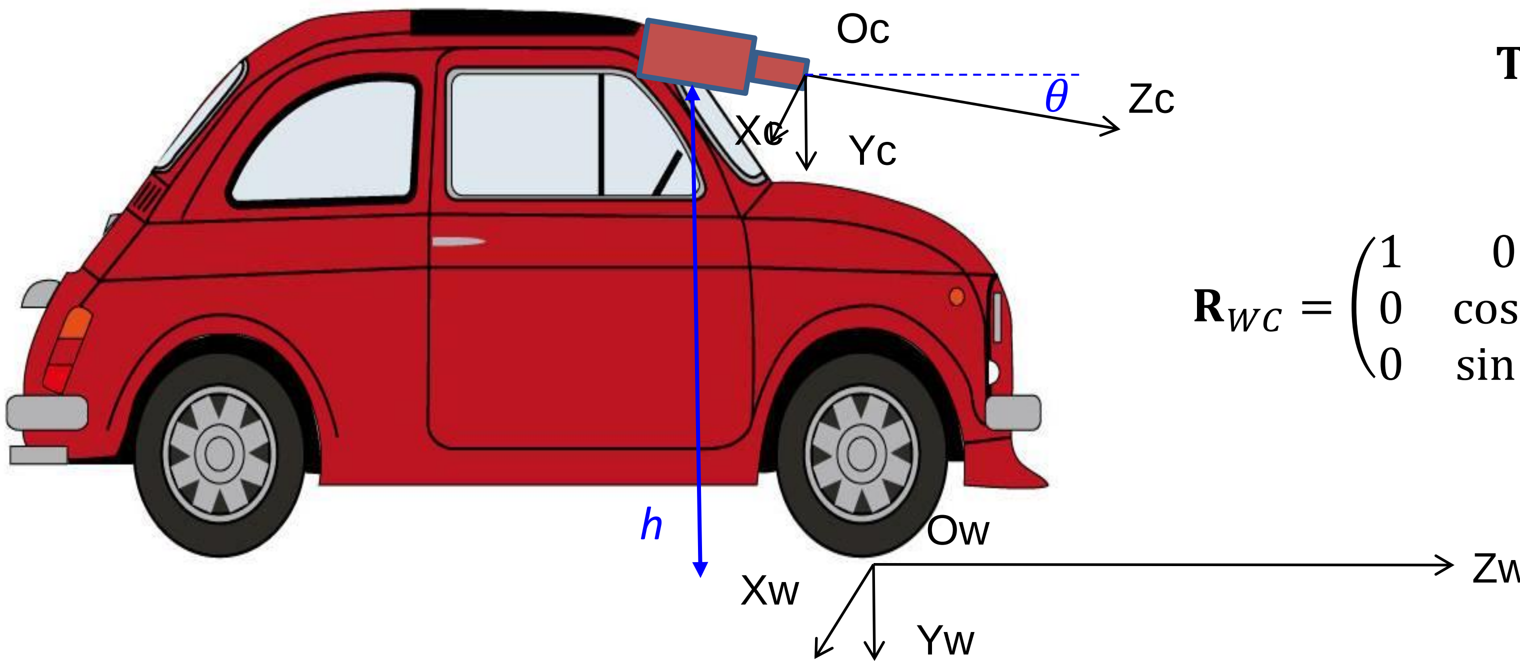
- **Estimation of the camera position and orientation with respect to a host vehicle reference frame, with no interaction from the user.**
- **The system should work in the absence of road markings, or when these markings are not visible (crowded city traffic).**
- **The detection of the calibration landmarks should be independent on preliminary extrinsic camera calibration.**

The calibration problem

- **Classical camera calibration uses the known correspondence between measured 3D coordinates and their position in the image.**
  - **Requires a controlled, artificial environment.**
- **Automatic calibration of real scenes is mostly based on vanishing points [Caprile, 1990].**
- **The vanishing points are detected using multiple voting techniques, such as the Gaussian sphere voting [Magee, 1984] or RANSAC techniques [Bazin, 2012], or based on CNNs [Itu, 2017].**
- **These techniques require structured scenes, with straight parallel and perpendicular lines ("Mahnattan World" assumption).**
- **Additiona sensors can be used for automatic calibration [Bileschi, 2009], [Levinson, 2013].**

Camera model

- **The camera and the vehicle (world) coordinate systems:**



$$\mathbf{T}_{CW} = \begin{pmatrix} 0 \\ -h \\ 0 \end{pmatrix}$$

$$\mathbf{R}_{WC} = \begin{pmatrix} 1 & 0 & 0 \\ 0 & \cos\theta & -\sin\theta \\ 0 & \sin\theta & \cos\theta \end{pmatrix}$$

Camera model

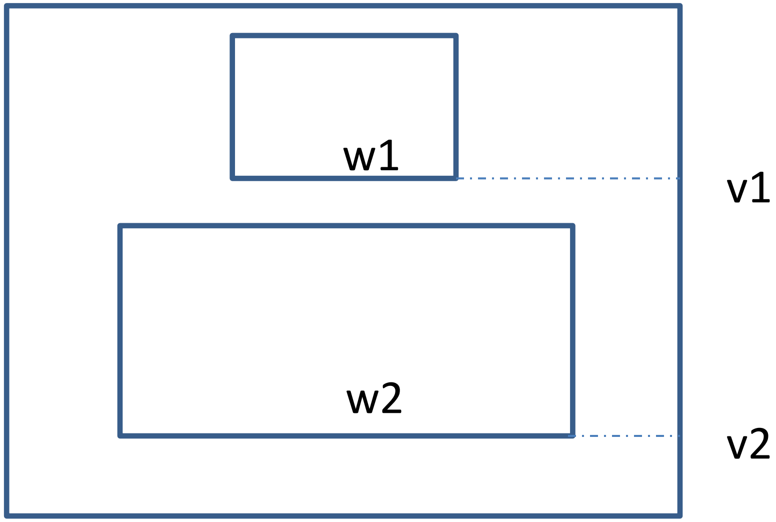
- **Projection of a 3D world point in the image space:**
  - ***u*** **– column coordinate relative to top left corner of the image**
  - ***v*** **– row coordinate relative to top left corner of the image**

$$\begin{pmatrix} u_s \\ v_s \\ s \end{pmatrix} = \mathbf{P} \begin{pmatrix} X_W \\ Y_W \\ Z_W \\ 1 \end{pmatrix}$$

$$\mathbf{P} = \mathbf{A}[\mathbf{R}_{WC}\ \mathbf{T}_{WC}] \qquad \mathbf{A} = \begin{pmatrix} f & 0 & W/2 \\ 0 & f & H/2 \\ 0 & 0 & 1 \end{pmatrix} \qquad \mathbf{T}_{WC} = -\mathbf{R}_{WC}\mathbf{T}_{CW}$$

Estimation problem statement

- **We consider the height of the camera and pitch angle to be the unknown state vector X to be estimated.**
- **The measurement vector will be composed of two image space widths of a known 3D structure in the road plane, for given image rows.**
  - **The known 3D structures can be lanes (or any painted structure on the road, of standard size), or vehicles.**



Estimation using EKF

- **The state vector is initialized with default values (0 degrees for pitch, any value for height).**
- **At any iteration, we assume that we have the image widths w1 and w2 of a 3D object of known size, for two row coordinates, v1 and v2.**
- **The following steps are performed:**
  - **Prediction of the measurement vector, using the projection equations, the known object size *L*, the given image lines and the camera intrinsic parameters**

$$Z'_k = g_{v1,v2,L}(X_k)$$

  - **Computation of the Jacobian of the transformation, Mk**
  - **Computation of the Kalman gain:**

$$K_k = P_k M^T \left(M_k P_k {M_k}^T + R\right)^{-1}$$

  - **Computation of the updated state vector**

$$X_k = X_{k-1} + K_k(Z - Z'_k)$$

Measurement data

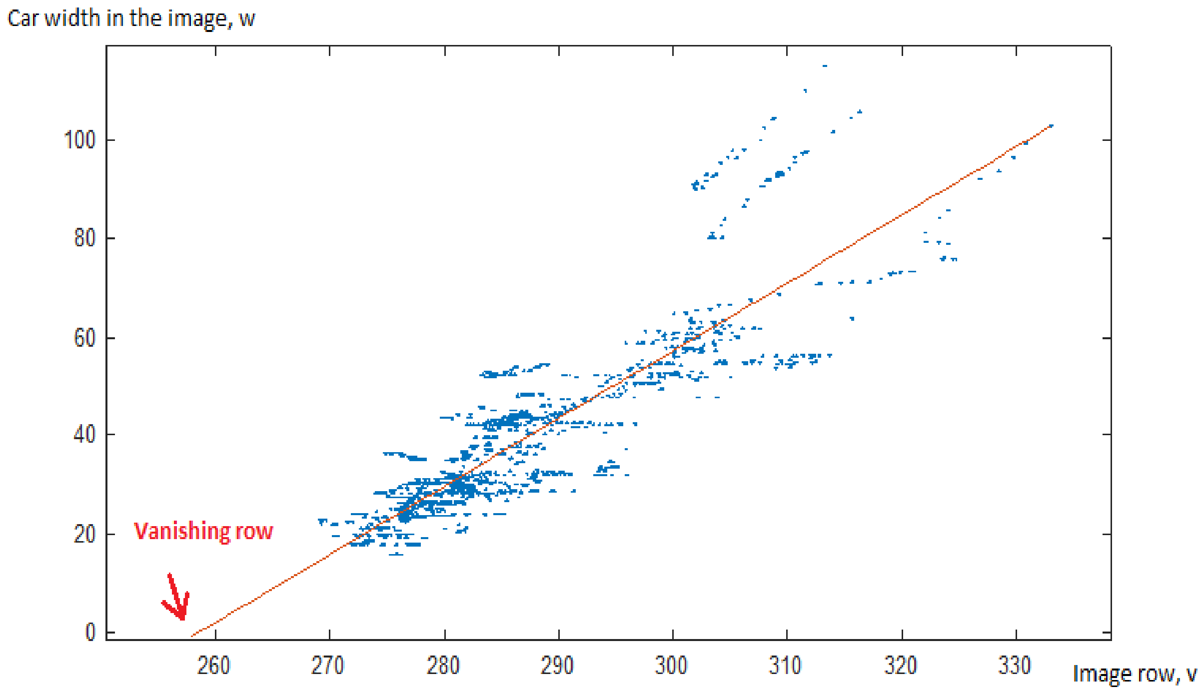
- **Vehicles are detected using the MobileNet CNN architecture.**
- **The network was trained on the KITTI and the Udacity datasets, for detecting passenger cars.**
  - **Completeness of the detection of obstacles is not of concern at this point. The detection results should be only useful for calibration.**

Measurement data

- **The MobileNet CNN architecture: single shot detector featuring a reduced number of parameters.**
- **The normal convolution operation is replaced by depthwise convolution followed by pointwise convolution.**
- **Training is done on a desktop machine using gradient descent and two loss functions: one for detection and one for classification (smoothed L1 and weighted sigmoid loss).**
- **The network input represents images resized to 300x300 px.**
- **The output of the network is formed by bounding boxes representing the location of passenger cars.**

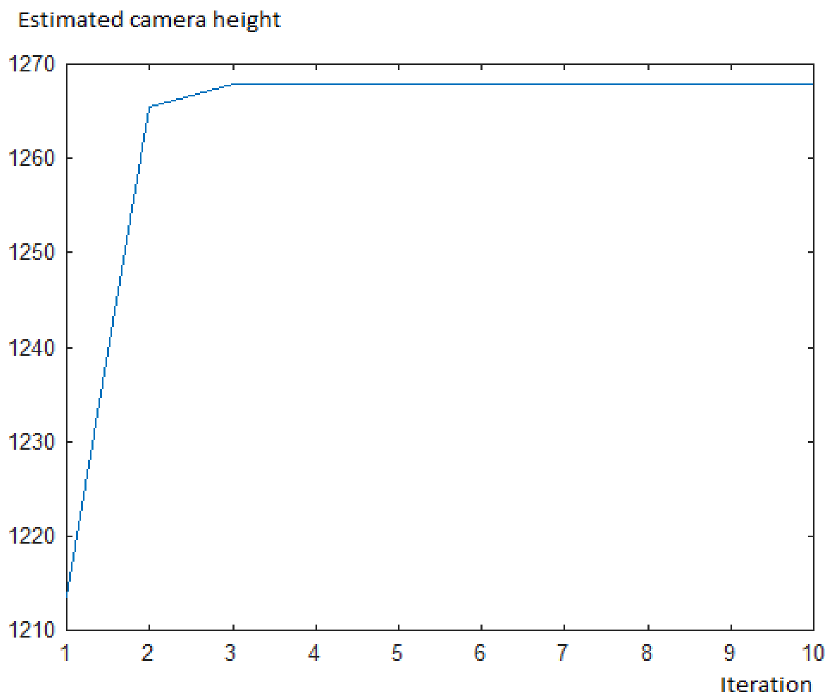
Measurement data

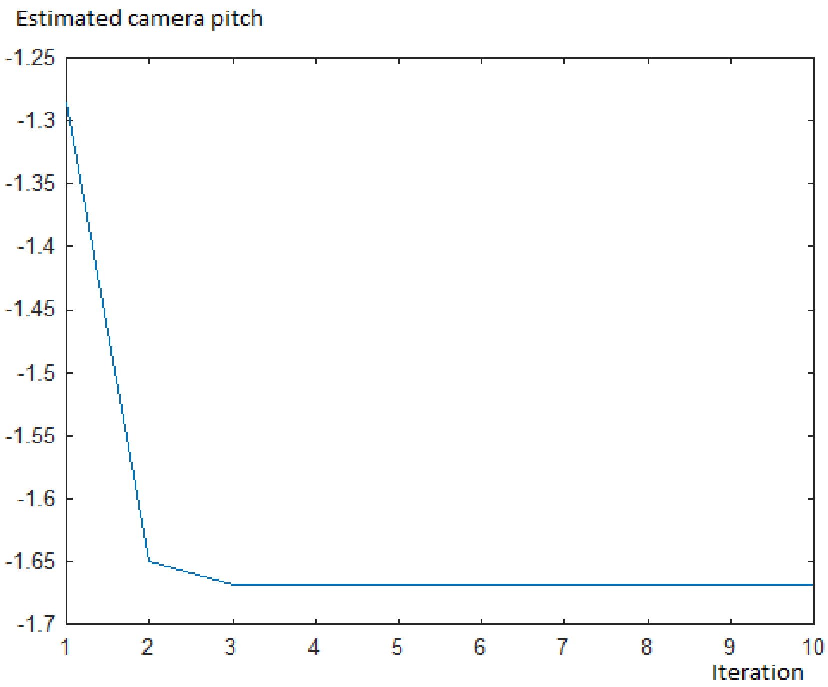
- **Car width versus image line (bottom line of the detected vehicle, the point of contact with the road), for a sequence of 8 minutes of driving:**
- **RANSAC is used to fit a line to the data, and two point on the line are used as measurement vector for the EKF.**



Estimation results

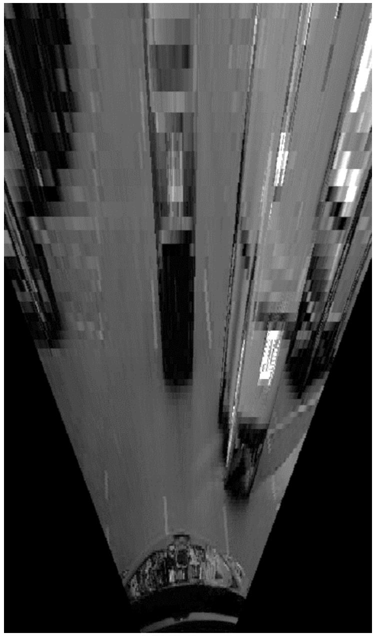
- **Height and pitch angle converge in 4-5 iterations**
  - **Height ground truth 1.25 m, pitch ground truth unknown, but must match the vanishing point in the image**





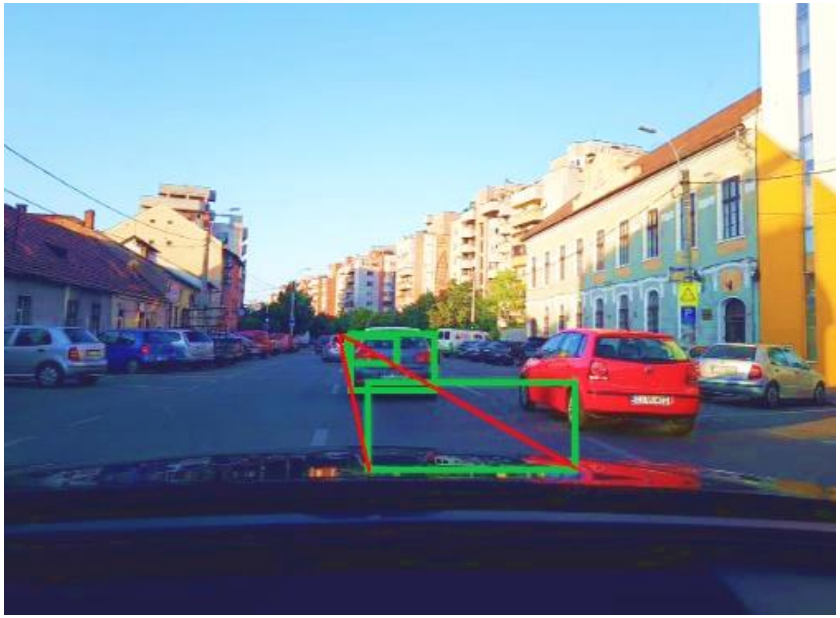
Estimation results

- **Checking the results via the projected horizon line and the Inverse Perspective Mapping:**
  - **Horizon line matches the real horizon.**
  - **The IPM image shows parallel lane markings, and a lane width of 3.25 m (Ground truth, 3.20 m).**



Yaw angle estimation

- **Yaw (heading) angle was assumed to be zero, but from the IPM image it can be seen that the assumption was wrong.**
- **The yaw angle is related to the *u* position of the vanishing point – the end point of the forward going vehicles' trajectories.**



Yaw angle estimation

- **No tracking is performed. Instead, obstacles are paired in consecutive frames on simple position constraints.**
- **From one pair, a trajectory is constructed and a potential vanishing point is determined.**
- **The median position of the vanishing points' *u* coordinate is chosen.**

Yaw angle estimation

- **Yaw angle is computed from the vanishing point:**

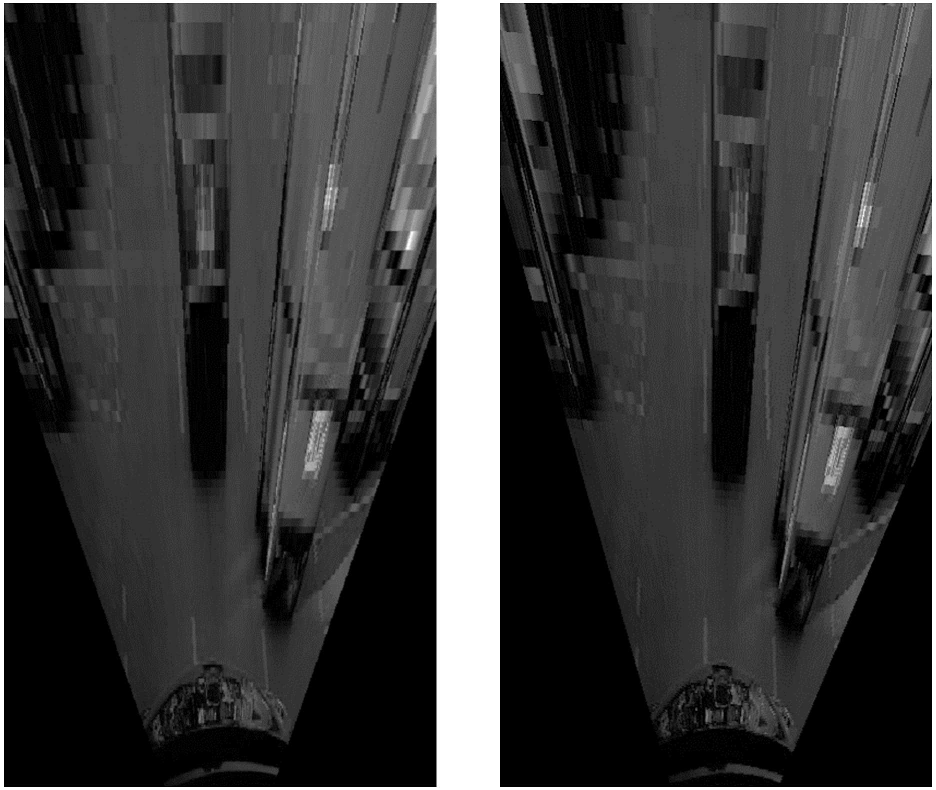
$$tan\Psi = \frac{u_0 - W/2}{f}$$

- **The rotation matrix is re-computed taking the yaw into consideration:**

$$R_{WC} = \begin{pmatrix} 1 & 0 & 0 \\ 0 & cos\theta & -sin\theta \\ 0 & sin\theta & cos\theta \end{pmatrix} \begin{pmatrix} cos\Psi & 0 & sin\Psi \\ 0 & 1 & 0 \\ -sin\Psi & 0 & cos\Psi \end{pmatrix}$$

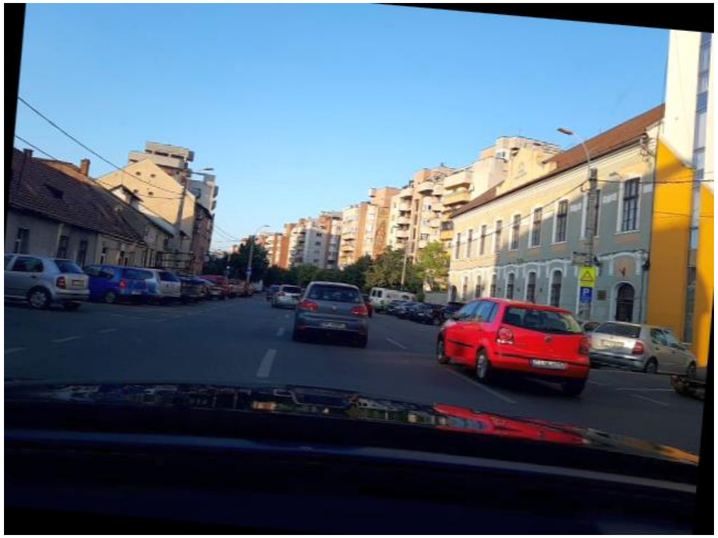
Yaw angle estimation result

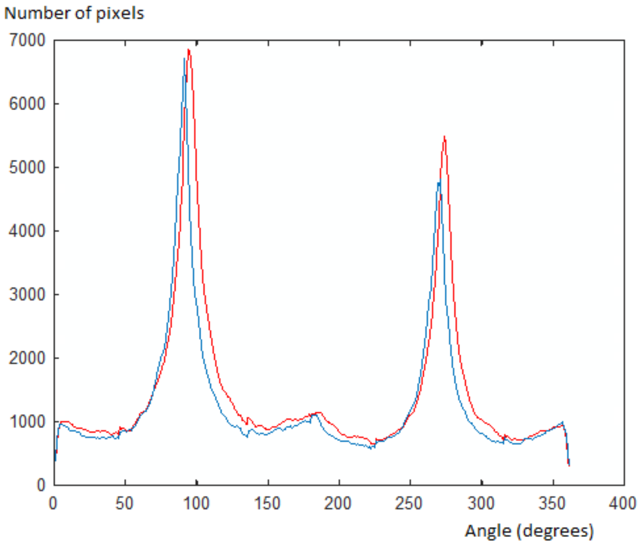
- **The effect on the IPM transformation:**



Roll angle estimation

- **Roll: the angle of rotation around the optical axis (camera Z axis)**
- **Detection of the roll: histogram of oriented gradients on vehicles detected in a central position**
  - **Histograms from single frames are combined for the whole sequence**
- **As the vehicle's features are mostly vertical and horizontal, the histogram peaks around angles multiple of 90º.**
- **The roll angle will cause the shift of the maxima:**





Results

- **Time performance:**
  - Obstacle detection, 100ms / frame on a Samsung Galaxy S8+ phone
  - Calibration performed offline, after a longer sequence is acquired (5 minutes or more) of driving, less than 1 minute of processing
- **Angle estimation performance:**
  - Pitch and yaw angles are estimated correctly, with less than 0.1 degree of error
  - No ground truth (and very difficult to estimate the effect on IPM) for the roll angle, but simulated roll angles (artificial rotation of the image by a set angle) were detected with 0.2 degrees of precision
- **Camera height estimation performance:**
  - The camera height estimation is more sensitive, as sometimes we can have errors of more than 10 centimeters.

Discussion

- **The cause of errors (especially on height) is simply that there are too few vehicles detected, and they lack diversity. Some scenarios:**
  - **A sequence may contain only one vehicle, in front of us, that we follow. If this vehicle is narrow, or wide, and does not fit the 1.75 m average width, the height estimation will fail.**
  - **Most of the detected vehicles are on the side of the road, and they will be detected as larger boxes in the image, including their side view.**
- **The solution for overcoming these problems:**
  - **Collect more data, with diverse obstacles in front of us.**
  - **Use a more complex classifier, which is able to give us more information about the detected vehicles (type of vehicle, orientation, etc).**

Conclusion and future work

- **Automatic estimation of the camera's extrinsic parameters with respect to the host vehicle's reference frame.**
- **The knowledge used for calibration is the result of a CNN-based vehicle detector, which does not require any sort of calibration**
- **The system does not require the presence of lane markings, but can use them with the same estimator**
- **Longer, more diverse sequences mean better results**
- **With small changes, the algorithm can be used for on-line refinement of the calibration results.**

THANK YOU

users.utcluj.ro/~rdanescu

http://users.utcluj.ro/~razvanitu/

cv.utcluj.ro

radu.danescu@cs.utcluj.ro

razvan.itu@cs.utcluj.ro