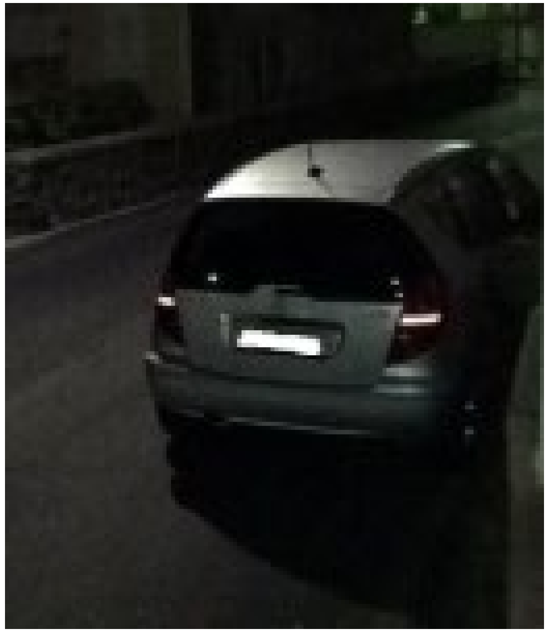
Car motion in the dark



Purpose: develop a system that

- analyzes a video of a moving vehicle taken by a fixed camera

in order to

- determine the occupied space for each frame

knowing

- K MATRIX: the intrinsic calibration parameters of the used camera

and

- MODEL: lenght and width of the car, and distance between car lights

Assumptions

The moving car has a «vertical» symmetry plane.

Two symmetric rear lights are visible:

Suppose that the camera is observing the car from the back,

Between subsequent frames of the video, the car is either translating forwards or steering with constant curvature

The road is locally planar

System operation: offline steps

- Calibrate the camera

- Retrieve the simplified MODEL of the observed car:

 - car width,

 - car lenght,

 - distance between the rear lights, and distance of both from the back of the car (sometimes this could be ≠ 0)

 - height of the rear lights over the ground

System operation online steps basing on two «close» frames

- Extract two symmetric lights in the first frame (or two symmetric features on the lights when the resolution is high enough) that have correspondence in the second frame. Call the «horizontal» segment joining the two symmetric lights THE LIGHT SEGMENT. So the first/second light segment is the one observed in the first/second frame

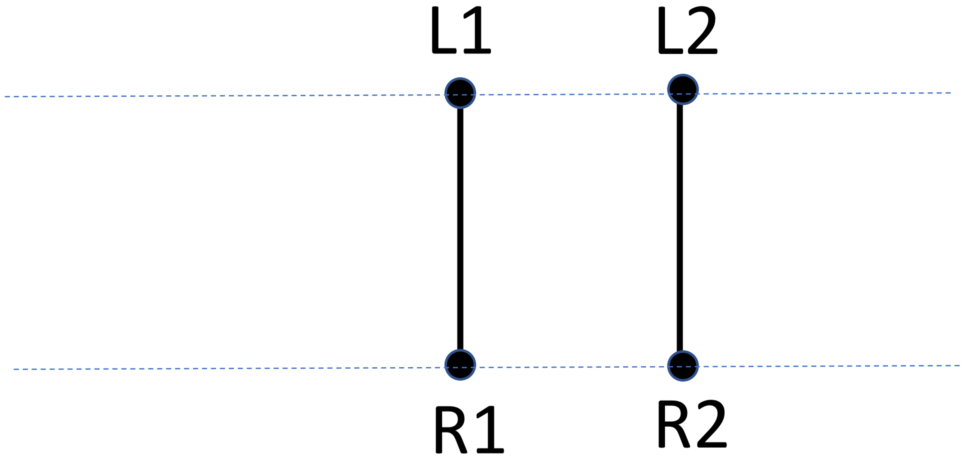
- Apply geometry to find the 3D position of the plane **π** containing the moving light segment, and the position of the lights on **π** (see next slides)

- For each of the two frames, use the simplified car model + the found position of both **π** and the lights in order to determine the space occupied by the vehicle on the road

Geometric facts: within the plane $\boldsymbol{\pi}$

1. **If the car is translating forwards**

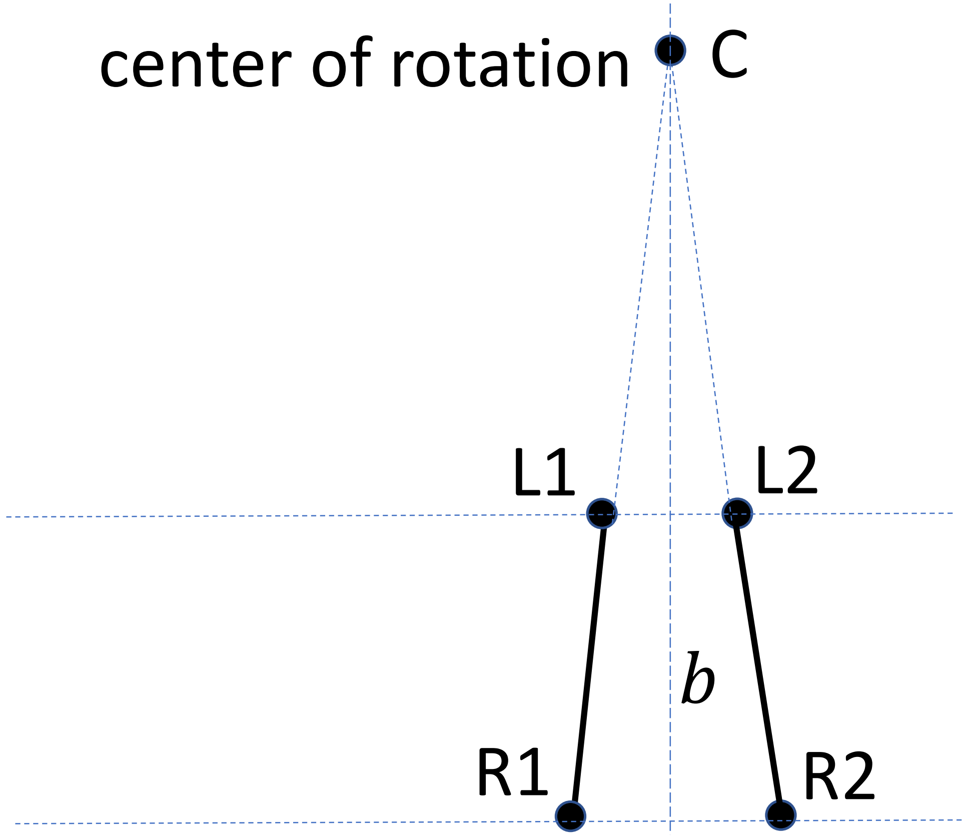
→ the first and the second light segment form a **rectangle**



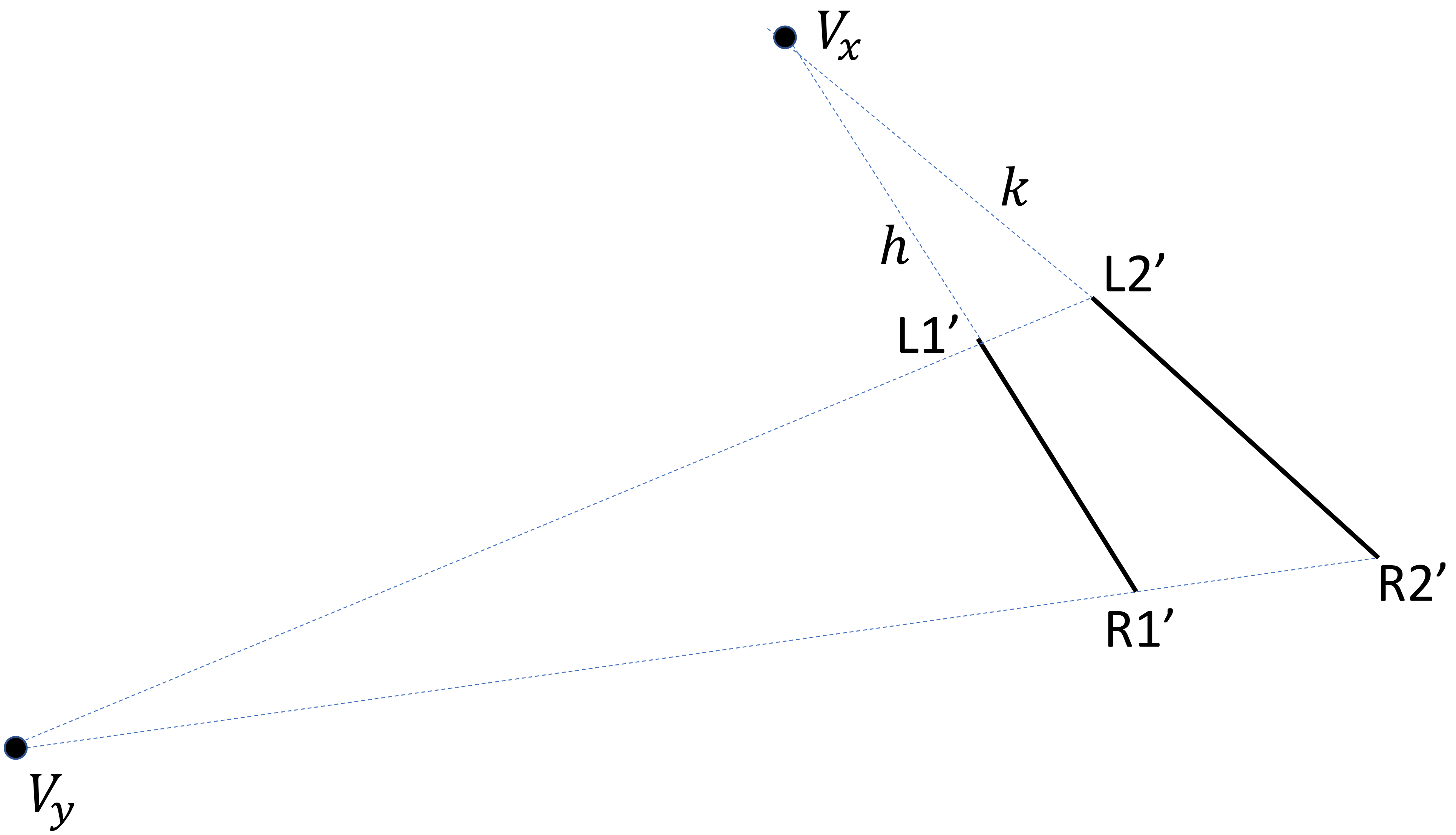
2. If the car is **steering** at constant curvature

→the light segment is rotating about a center of rotation C, that is the intersection between the lines (L1, R1) and (L2, R2) hosting the segments

→ lines (L1, L2) and (R1, R2) are parallel in the real world, and both are perpendicular to the line b bisecting the lines the lines (L1, R1) and (L2, R2)

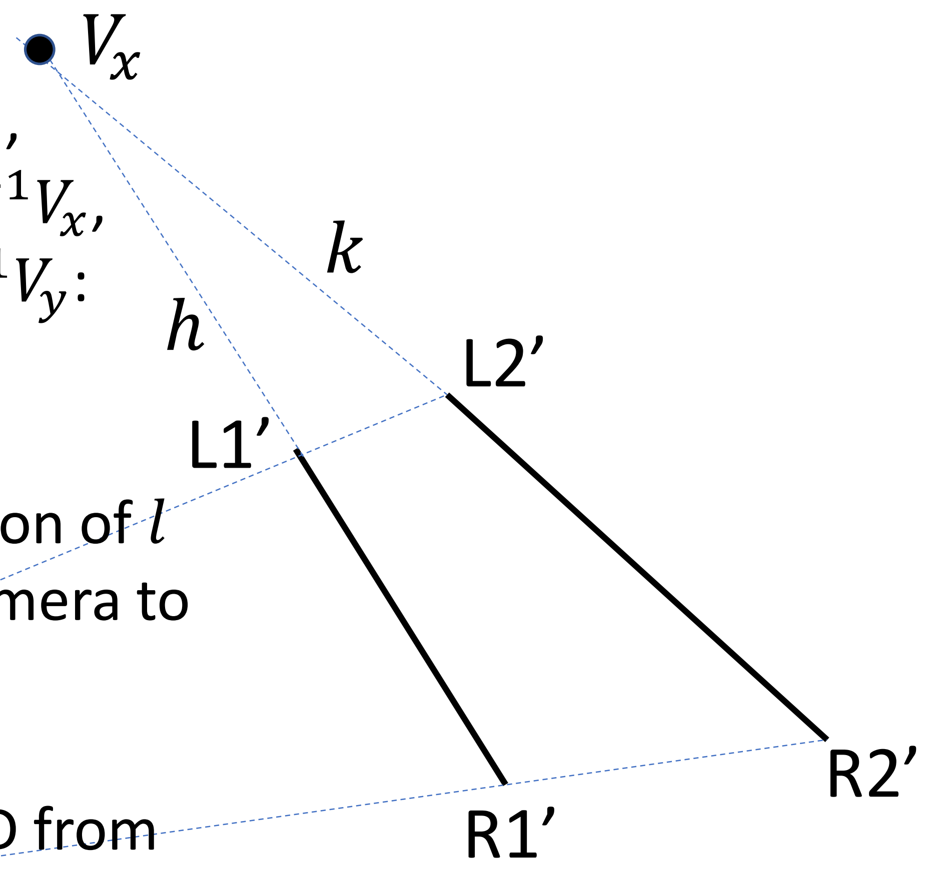


1. car translating forwards: image

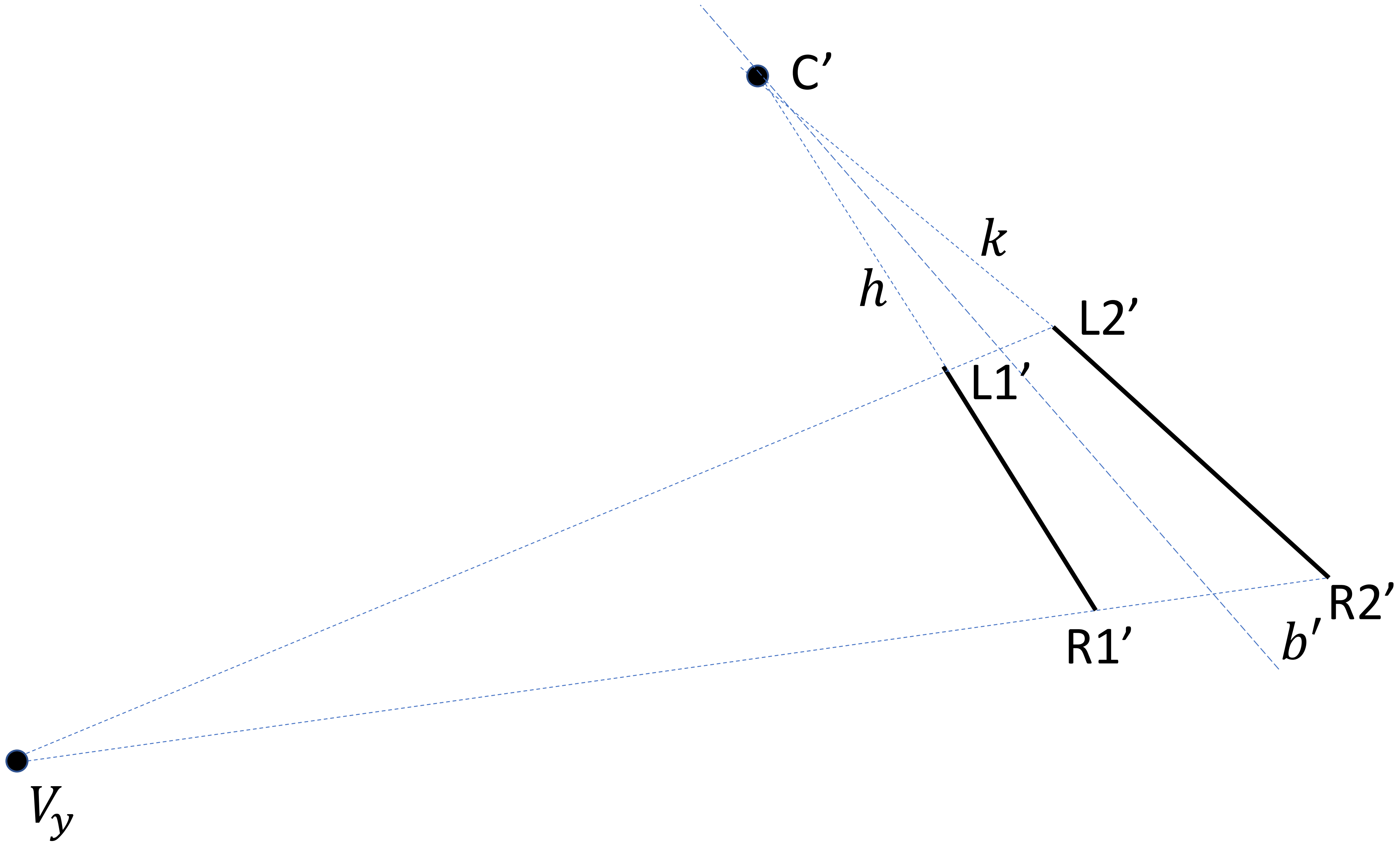


1. car translating forwards: method

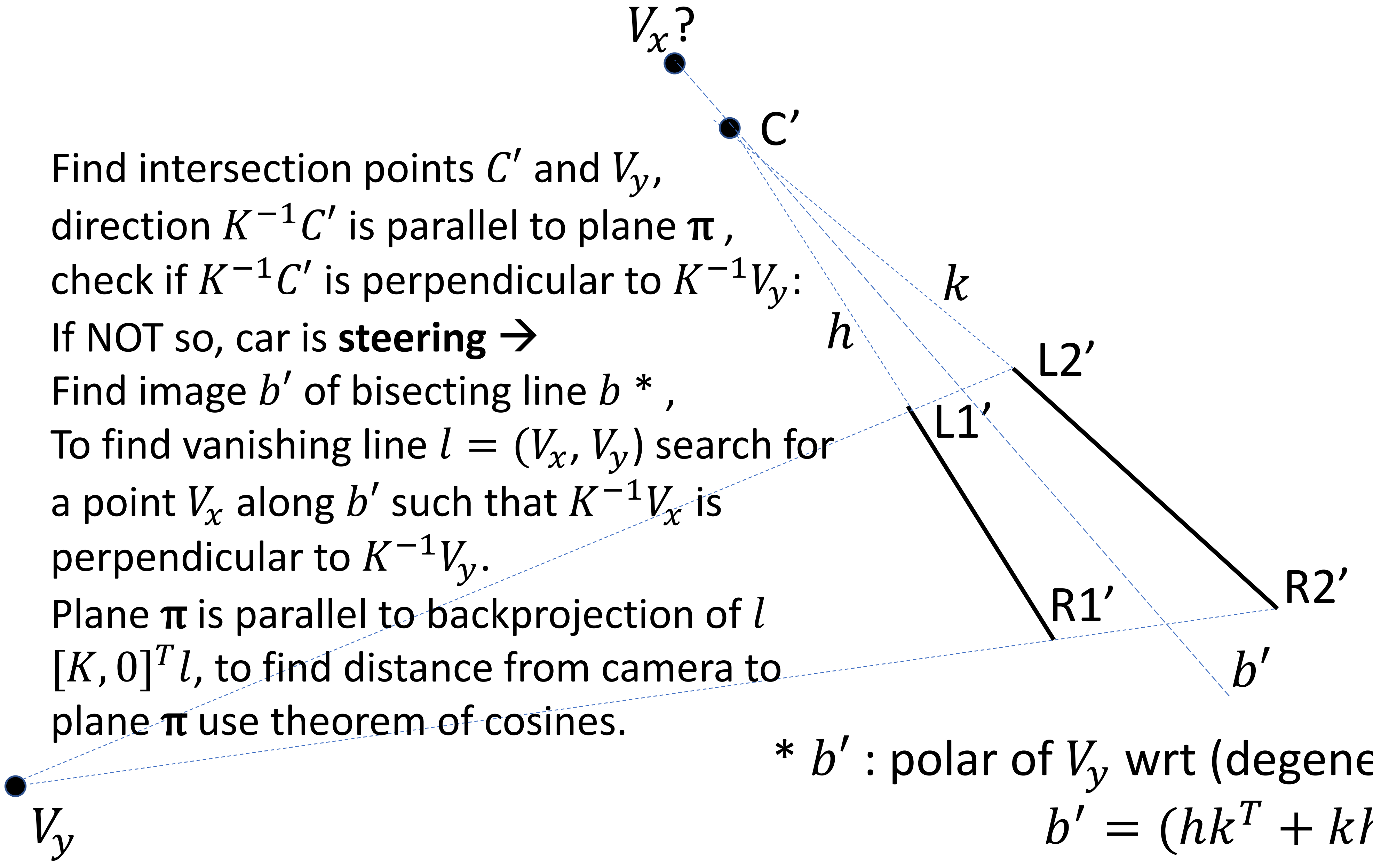
Find intersection points V_x and V_y,
3D direction of light segment: $K^{-1}V_x$,
check if it is perpendicular to $K^{-1}V_y$:
If so, car is **translating** →
Find vanishing line $l = (V_x, V_y)$.
Plane **π** is parallel to backprojection of l
$[K, 0]^T l$, to find distance from camera to
plane **π** use theorem of cosines.
OR
Localize both light segments in 3D from
image knowing direction $K^{-1}V_x$ and size
(use, e.g., theorem of cosines)
OR use time-to-impact



2. car steering with constant curvature: image



2. car steering with constant curvature: method



Find intersection points C' and V_y,
direction $K^{-1}C'$ is parallel to plane **π** ,
check if $K^{-1}C'$ is perpendicular to $K^{-1}V_y$:
If NOT so, car is **steering** →
Find image b' of bisecting line b * ,
To find vanishing line $l = (V_x, V_y)$ search for
a point V_x along b' such that $K^{-1}V_x$ is
perpendicular to $K^{-1}V_y$.
Plane **π** is parallel to backprojection of l
$[K, 0]^T l$, to find distance from camera to
plane **π** use theorem of cosines.

* b' : polar of V_y wrt (degenerate) conic $h \cup k$

$$b' = (hk^T + kh^T)V_y$$

Restrictions:

This method, that uses just the images of the car lights, only works if there is enough PERSPECTIVE.

Otherwise, if viewing rays are practically parallel, the above data are not sufficient, and additional information is needed, symmetric elements on another plane

Suggestions:

set up experiment with enough perspective, i.e., place camera with good inclination and allow sufficient motion between used frames (possibly exploiting intermediate frames to help feature tracking)

first focus on case 1 (car translating straightforward); in this case, also front lights can be used

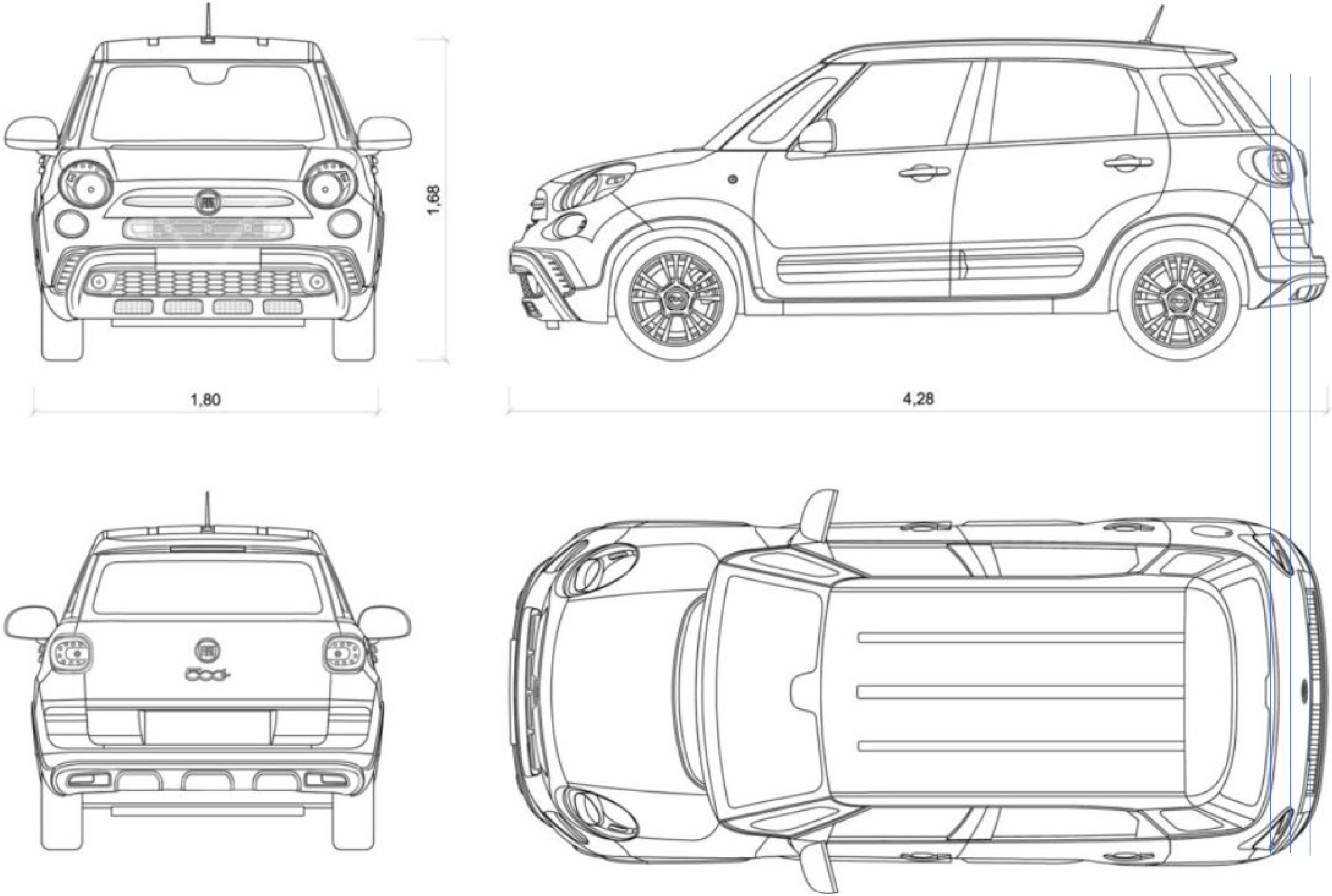
Extract ground truth from independent information (e.g., road plane) not used in the method

Car localization

{'Scion xD Hatchback 2012': '100.00%'}

Use pairs of symmetric features

{'Scion xD Hatchback 2012': '100.00%'}


1,68
1,80
4,28

Proposta operativa: sviluppo e test di 3 tecniche
Ipotesi: camera calibrata → $\mathbf{K}$ nota

1. Localizzazione standard da singola immagine del quadrangolo piano luci posteriori – targa. *potrebbe soffrire la mancanza di prospettiva*

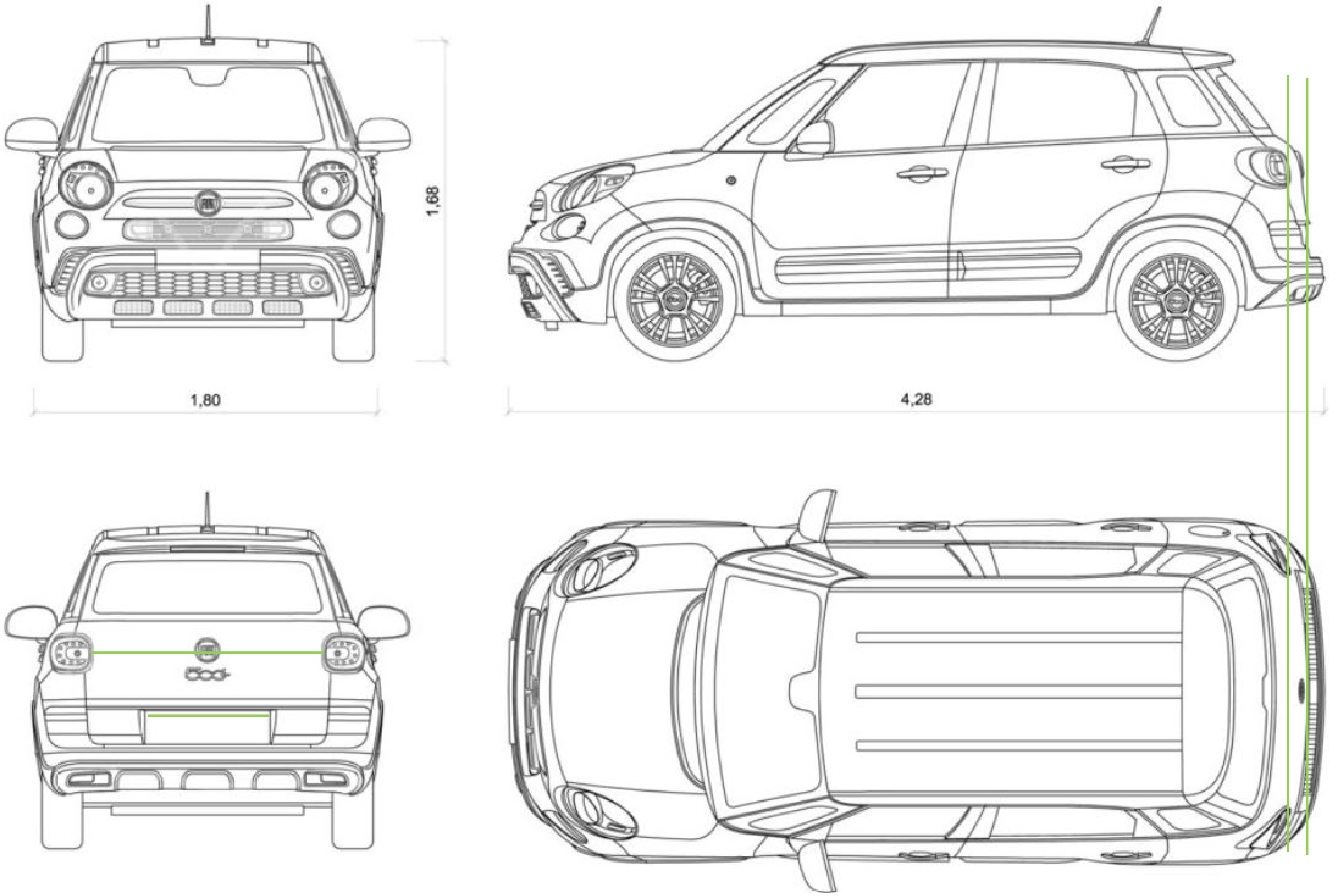
2. Localizzazione «notturna» da coppia di immagini (non consecutive, in modo da aumentare la prospettiva) di un video per il caso **traslatorio**: uso delle luci posteriori

3. Localizzazione con elementi simmetrici non coplanari. Quando a causa della scarsa prospettiva (vanishing point vicini all'∞) la stima dell'inclinazione orizzontale (angolo di rotazione attorno all'asse verticale) è molto incerta, si può usare la differenza con cui appaiono nell'immagine due elementi che nella realtà sono simmetrici

1. Localizzazione standard

Dal modello cad dell'auto determinare le posizioni relative dei due estremi della targa e di due punti simmetrici delle luci posteriori (es quelli più vicini tra loro) e usare metodo di localizzazione da Homography $[\boldsymbol{r}_1 \quad \boldsymbol{r}_2 \quad \mathbf{o}_\pi] = \mathbf{K}^{-1}\mathrm{H}$

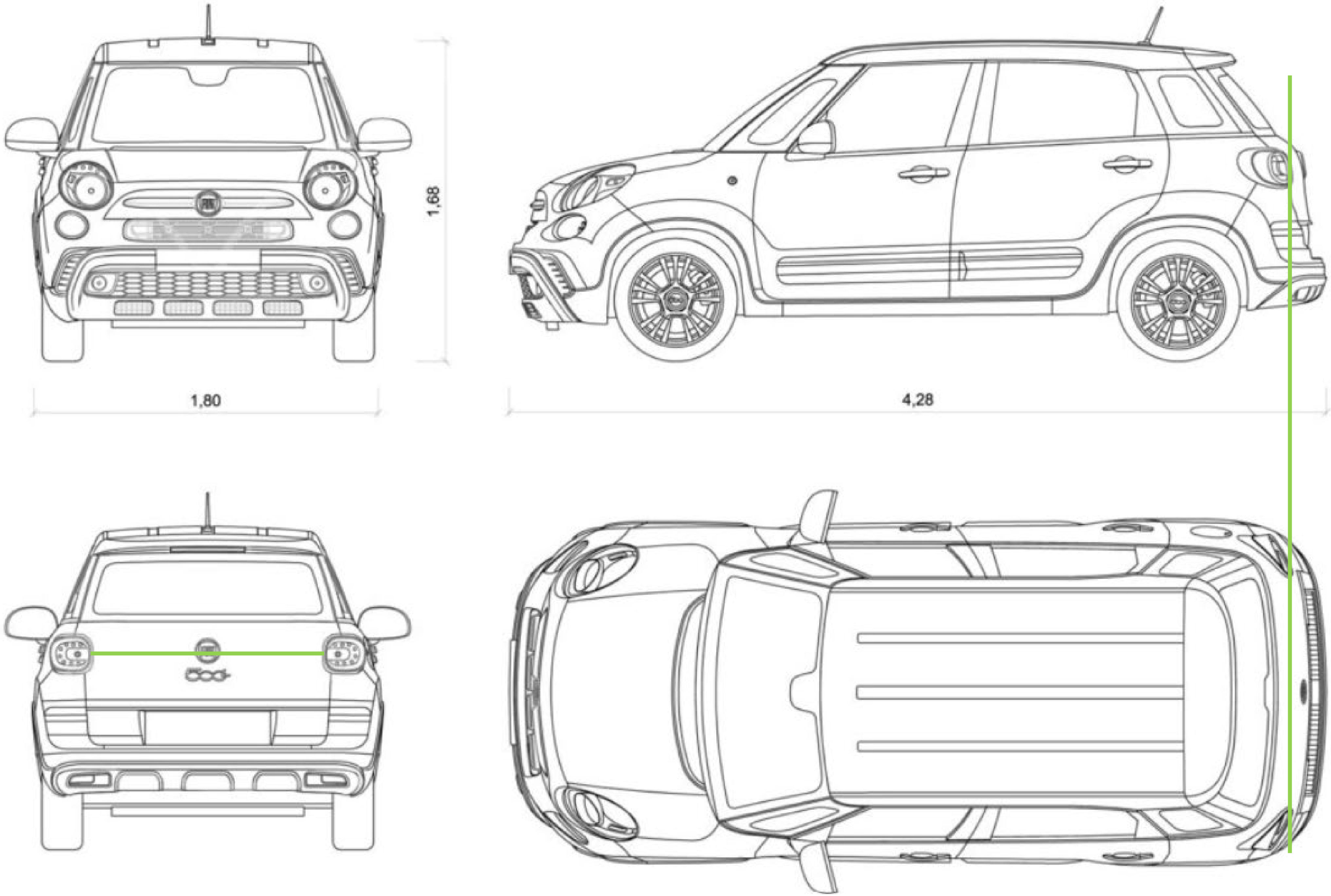
{'Scion xD Hatchback 2012': '100.00%'}


1,68
1,80
4,28

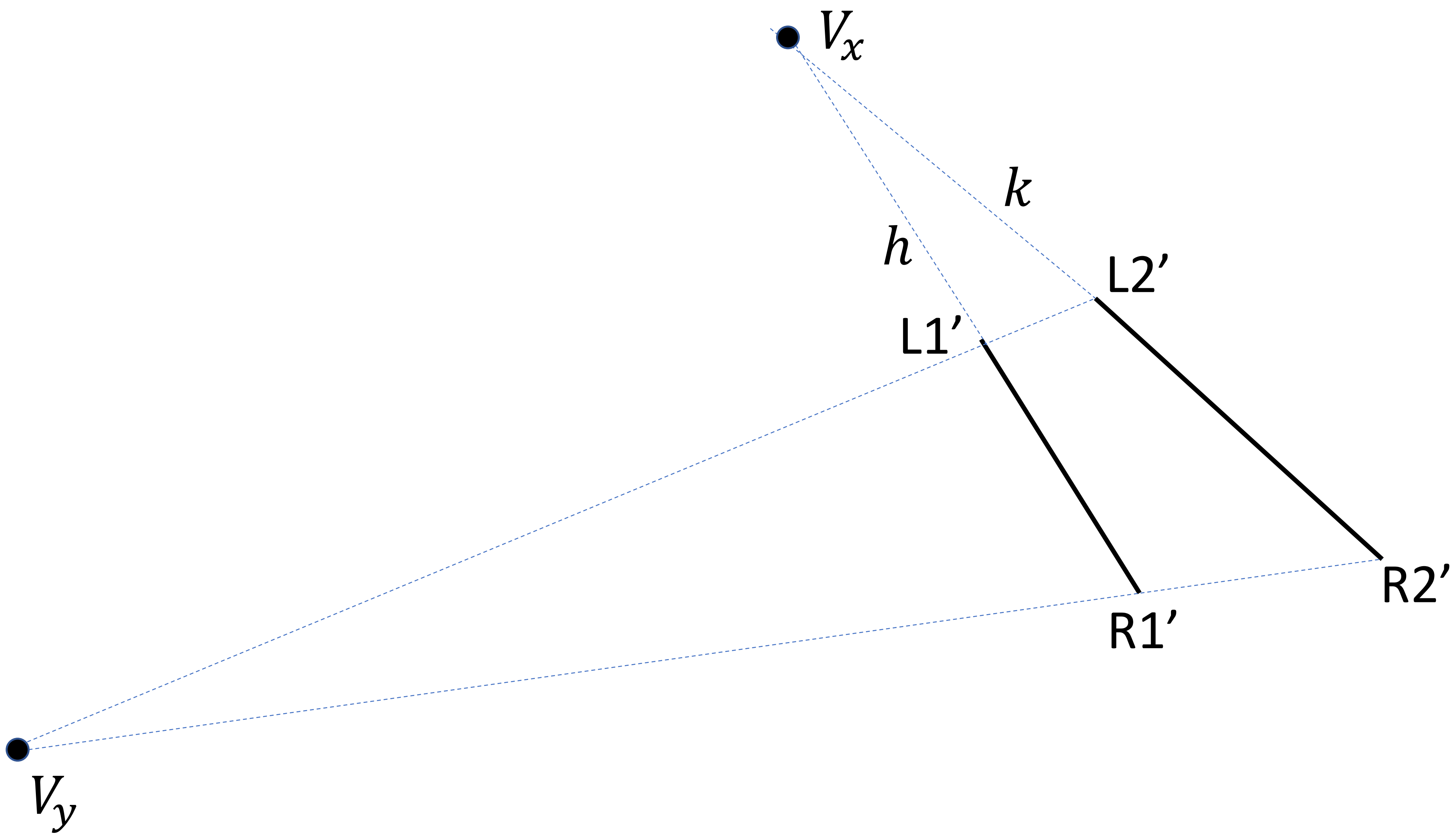
2. Localizzazione «notturna» da coppia di immagini

Dal modello cad dell’auto determinare le posizioni relative di due punti simmetrici delle luci posteriori (es quelli più vicini tra loro) e usare metodo di descritto in nelle tre slide seguenti

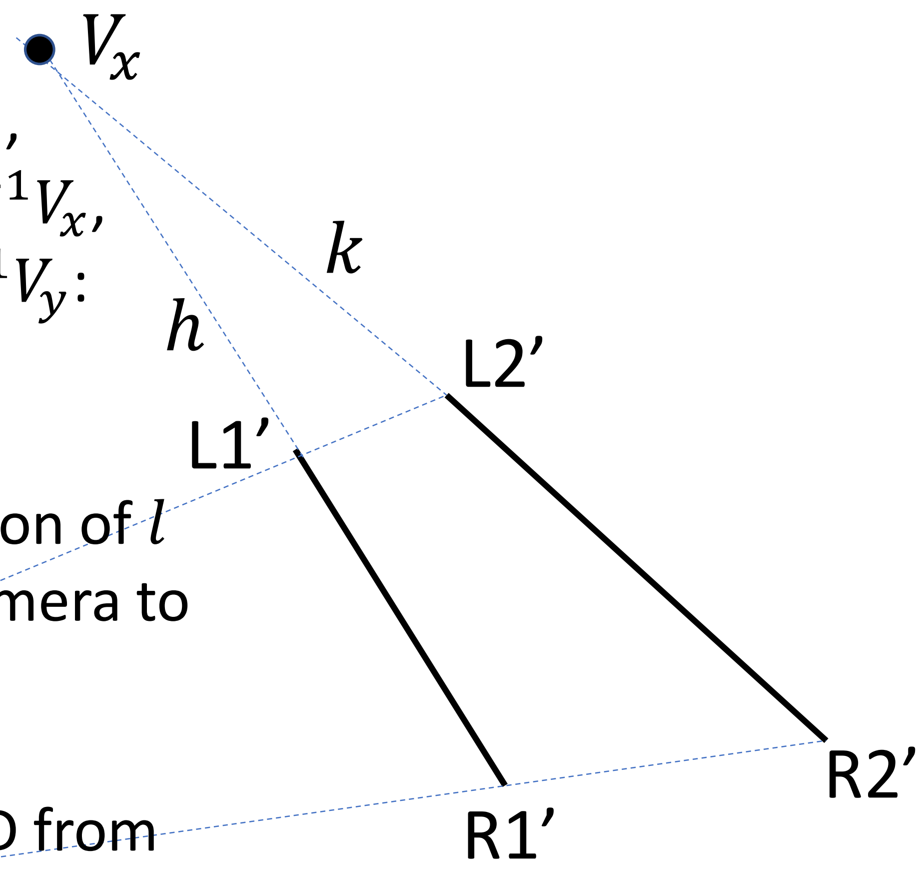
{'Scion xD Hatchback 2012': '100.00%'}


1,68
1,80
4,28

2. car translating forwards: image



2. car translating forwards: method assumes visible perspective effects



Find intersection points V_x and V_y,
3D direction of light segment: $K^{-1}V_x$,
check if it is perpendicular to $K^{-1}V_y$:
If so, car is **translating** →
Find vanishing line $l = (V_x, V_y)$.
Plane **π** is parallel to backprojection of l
$[K, 0]^T l$, to find distance from camera to
plane **π** use theorem of cosines.
OR
Localize both light segments in 3D from
image knowing direction $K^{-1}V_x$ and size
(use, e.g., theorem of cosines)
OR use time-to-impact

3. Localization under poor perspective using out-of-plane symmetric features

Localization under poor perspective using out-of-plane symmetric features

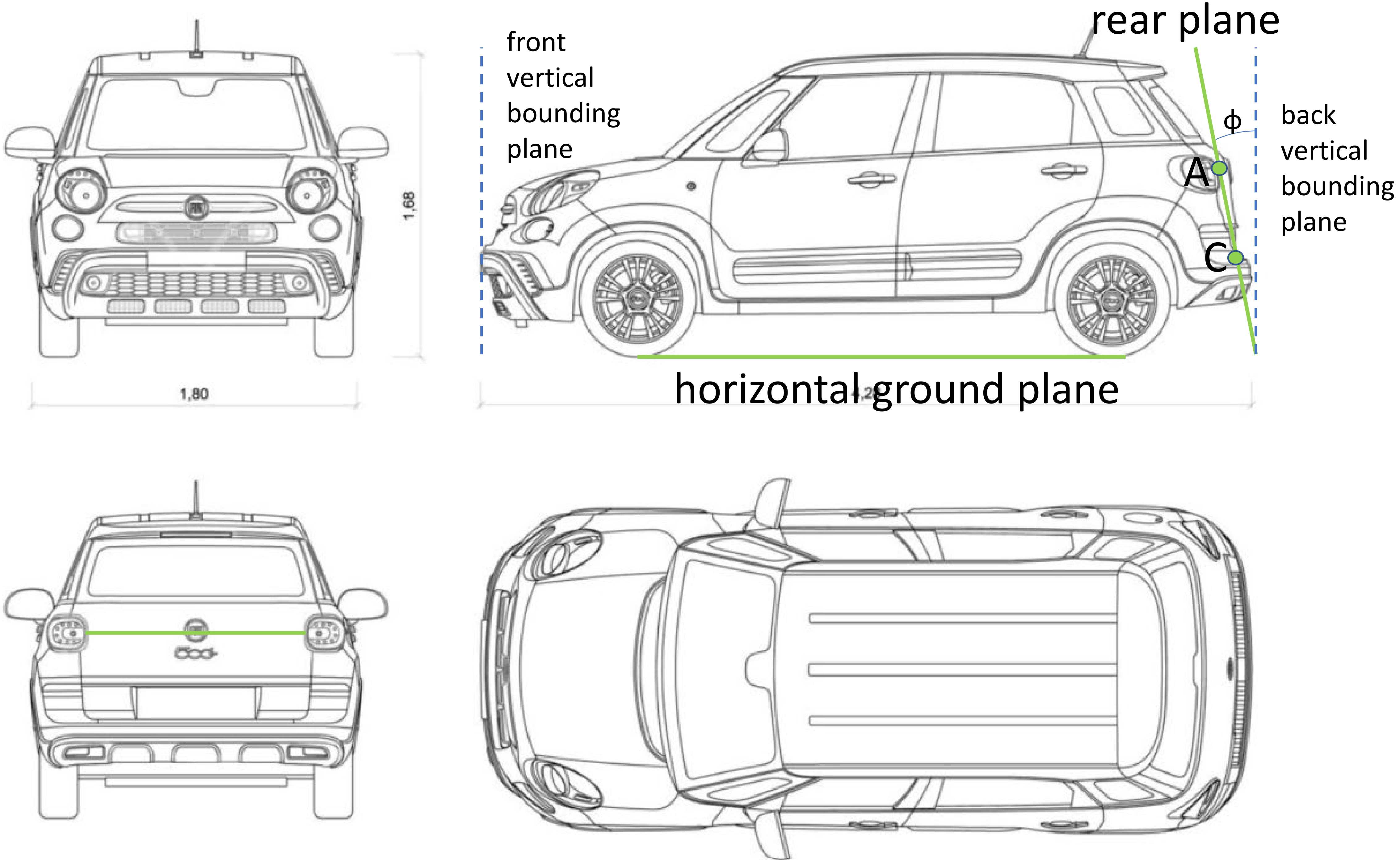
From cad model determine the positions of A, B, C, D, E, F and estimate the pose (position and orientation) of the car wrt the camera

{'Scion xD Hatchback 2012': '100.00%'}

Localization under poor perspective using out-of-plane symmetric features

Segments AB, CD and EF are parallel and symmetric. Points A, B, C, and D are and on the rear plane. Points E, F are out of plane.

{'Scion xD Hatchback 2012': '100.00%'}


1,68
1,80
front vertical bounding plane
rear plane
ϕ
A
C
back vertical bounding plane
horizontal ground plane

Standard localization

Given the coordinates of points A, B, C, D on the rear plane, and their image (taken by a calibrated camera) estimate position of the rear plane relative to the camera: standard homography-based method: $[\boldsymbol{r}_1 \quad \boldsymbol{r}_2 \quad \mathbf{o}_\pi] = \mathbf{K}^{-1}\mathrm{H}$

{'Scion xD Hatchback 2012': '100.00%'}

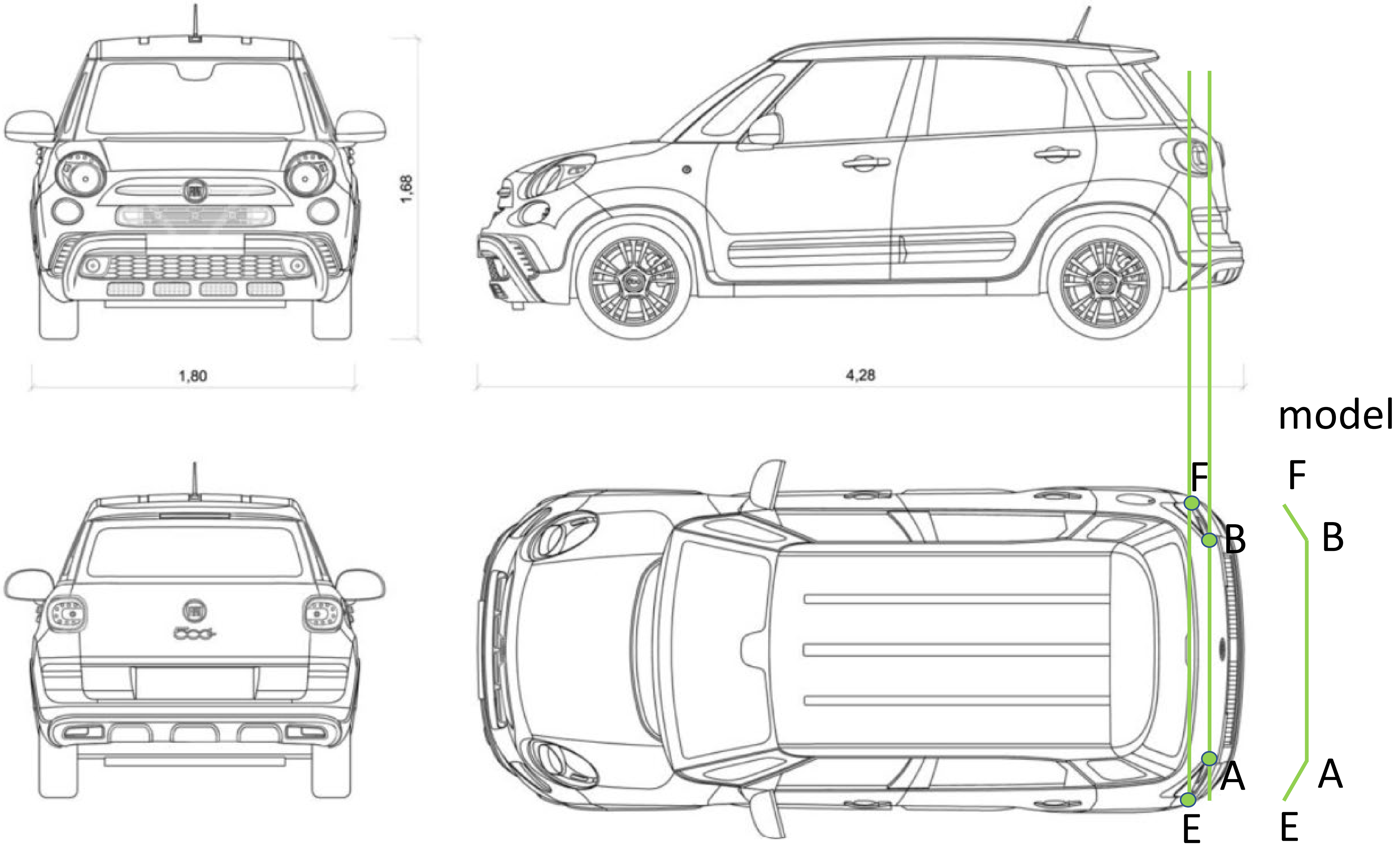
standard homography-based method can **fail** with poor perspective (e.g. when the vanishing point is close to the ∞) → use **iterative method**

1° estimate: vanishing point V_X = line(A'B') ∩ line(C'D') → direction of AB = $\mathbf{K}^{-1}V_X$

From length $\overline{AB}$, direction of AB and images A',B' → 3D position of AB, same for CD

→ position estimation of rear plane → from ϕ, find vertical direction wrt camera

{'Scion xD Hatchback 2012': '100.00%'}


1,68
1,80
4,28
F
B
A
E
model
F
B
A
E

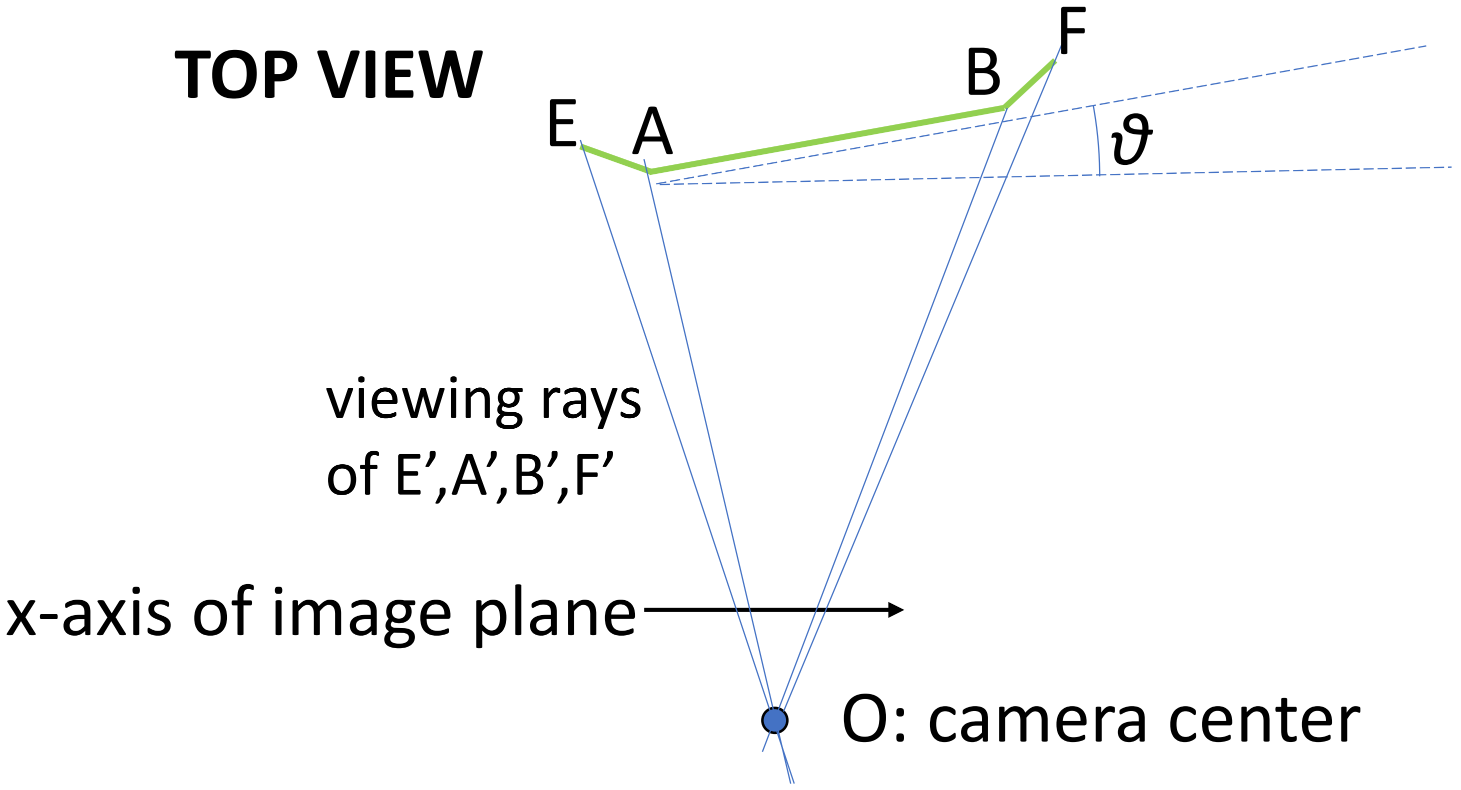
Rotation angle ϑ of the model E,A,B,F within the horizontal plane (rotation about the vertical axis): angle between x-axis of the camera and direction of AB (or CD or EF)

model

image

{'Scion xD Hatchback 2012': '100.00%'}

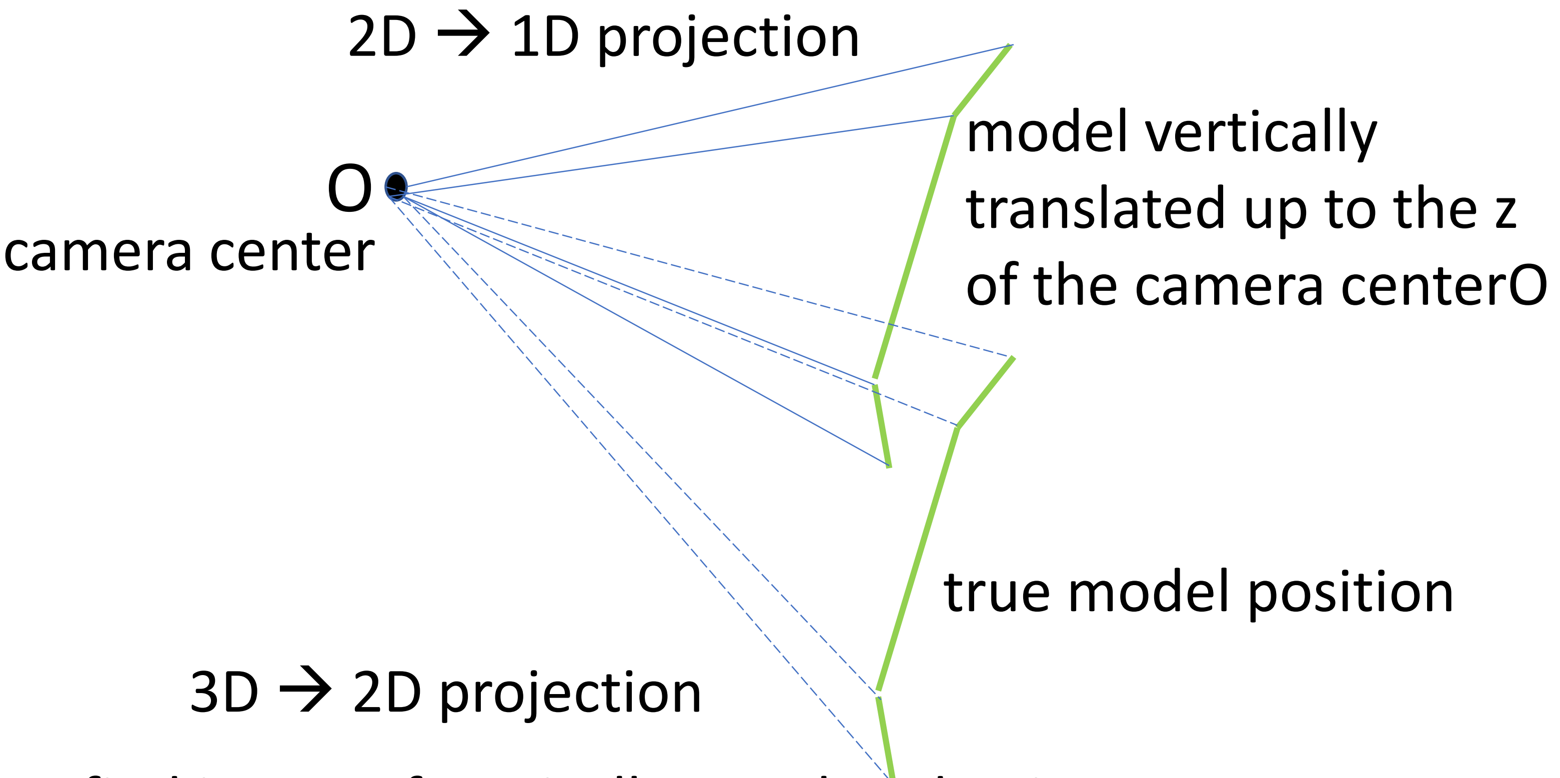
Rotation angle ϑ btw x-axis of camera and direction of AB (or CD or EF)

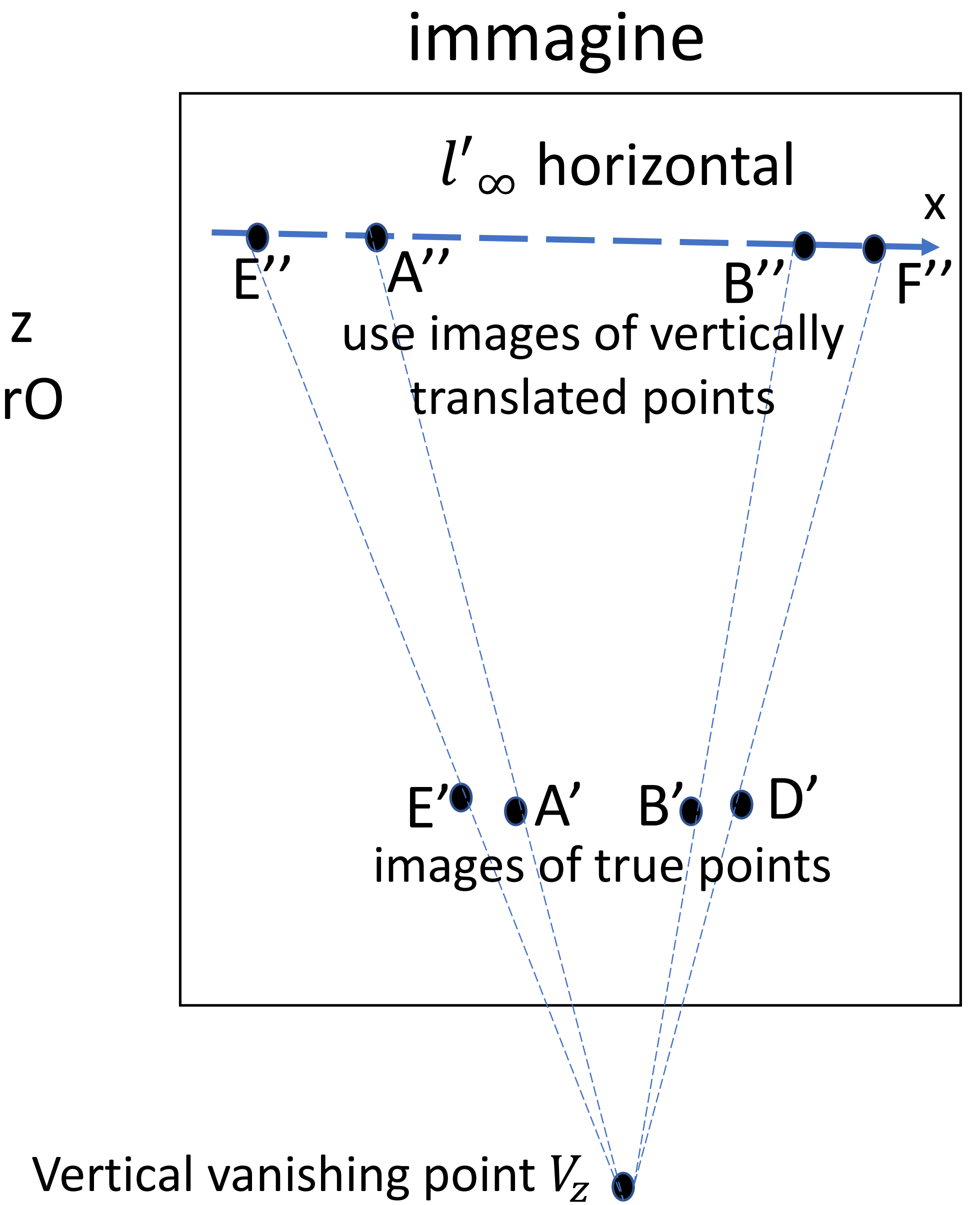


Instead of considering true points E, A, B, F we use vertically translated points, such that their z coords = to z coord of camera center O, and the images of these translated points* → all points are on a horizontal plane through O → Problem becomes 2D → 1D projection: one unknown ϑ.

Use geometry to find angle ϑ.

* Translated model points and their images





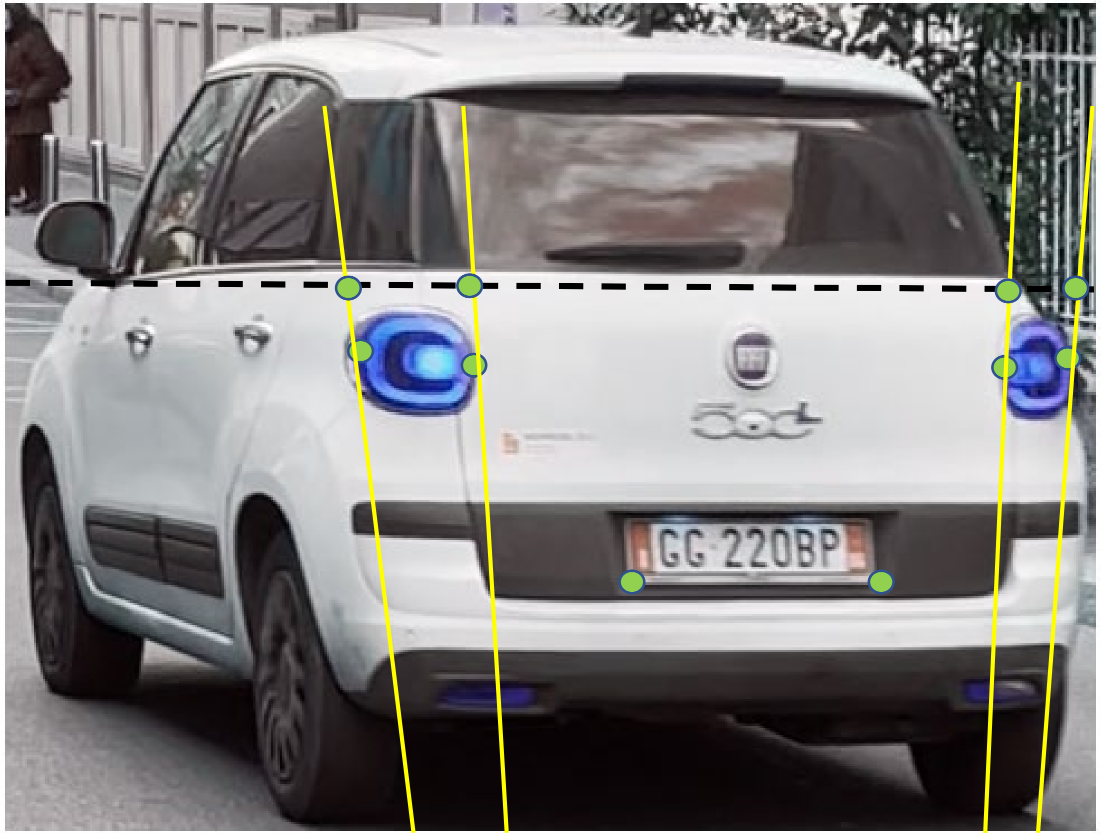
To find image of vertically translated points

- use estimated vertical vanishing point V_z
- derive estimated horizontal vanishing line l'_∞
- use new image points, e.g. A’’ = $l'_\infty \cap$ line(V_z , A’)

* used image points

model

image



$l'_\infty = \omega V_z$

{'Scion xD Hatchback 2012': '100.00%'}

… to vanishing point V_z

Iterative refinement of pose estimate

- From current estimate of vertical rotation angle ϑ
- Update estimated direction of segment AB (and CD)
- From new direction estimates, localize AB and CD in 3D, and update estimate of the rear plane (plane through A, B, C, D) and –from known angle ϕ- update both back vertical bounding plane and horizontal ground plane (relative to camera)
- Use calibration matrix to update estimates of vertical vanishing point V_z and horizontal vanishing line l'_∞
- Use updated V_z and l'_∞ to generate updated estimate of angle ϑ

ITERATE UNTIL CONVERGENCE