UNIVERSITÀ
DI TRENTO
Department of
Industrial Engineering

State and pose estimation of a racing vehicle from onboard camera footage using computer vision

Final dissertation
Master degree in Mechatronics Engineering
University of Trento
Academic year 2022/23

Supervisor: Francesco Biral
Co-supervisor: Edoardo Pagot

Student: Francesco Mazzoni
224840

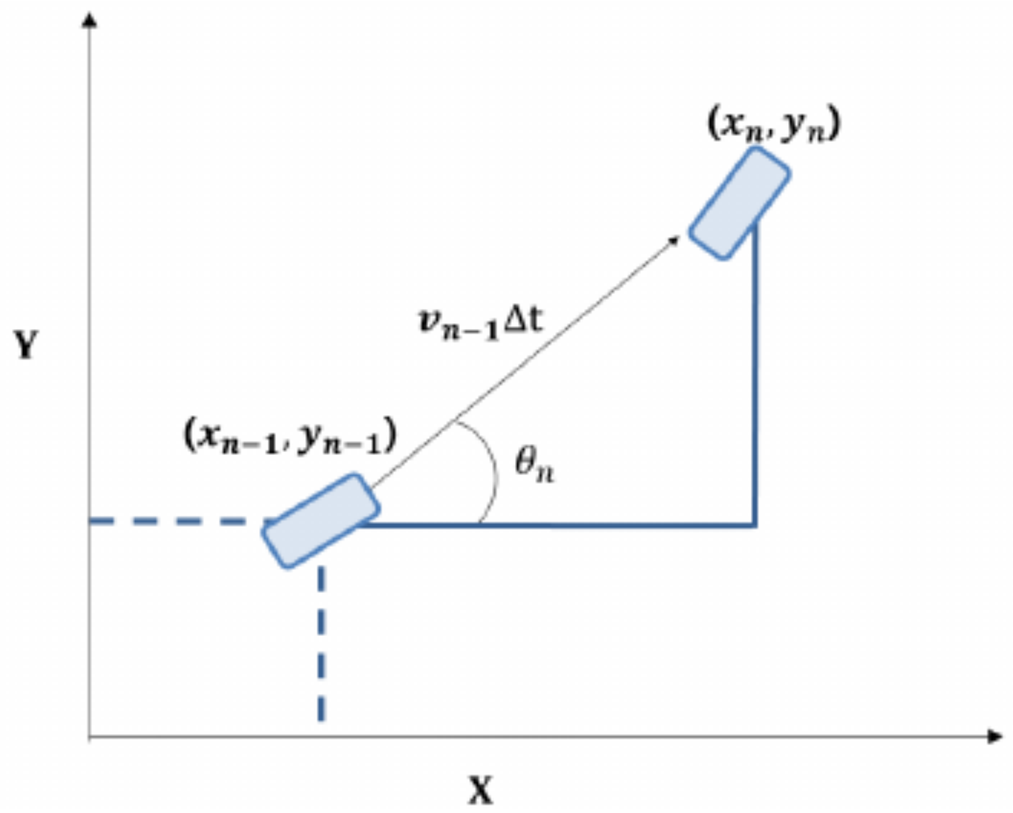
Contents overview

- Introduction to the problem
- First approach: state and pose estimation after camera calibration procedure
  - Camera calibration applied
  - The pattern
  - The procedure
  - Implementation
  - Results
  - Comments on results
- Second approach: constrained minimization problem via lane matching
  - Procedure steps to follow
  - Possible implementation
  - First tentative results

Problem statement

Introduction to the problem

- Car position reconstruction is important both in civil and racing scenarios
- Civil environments: autonomous driving, pedestrian recognition, maneuvers, etc...
- In racing scenarios, trajectory reconstruction can help improving car performances and racing strategies



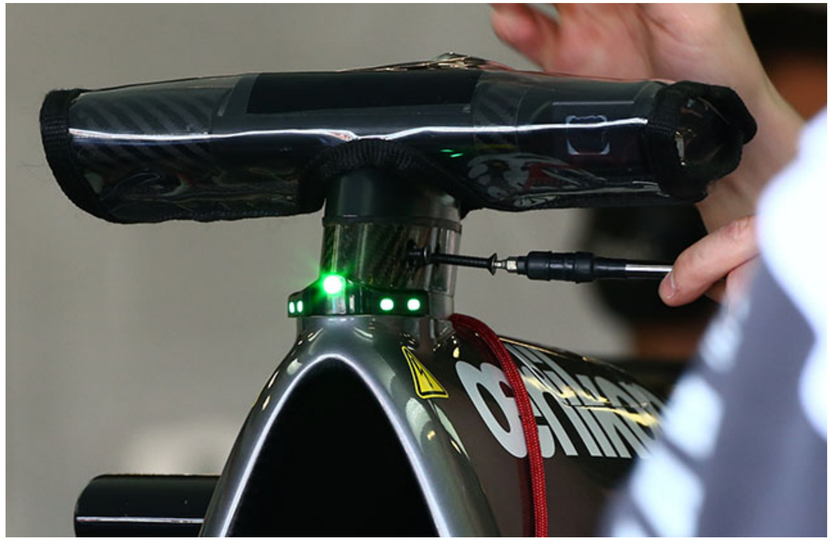
Introduction to the problem (cont'd)

- Vision system play an important role in the automotive sector
- They help reduce cost production
- From racing point of view: some vision sensors are not characterized, a calibration procedure would be interesting to be performed

Introduction to the problem (cont'd)

The work aims at:

- Characterizing a vision system in onboard F1 racing cars
- Reconstruct car position from vision information and camera calibration
- Eventually refine the technique and provide an alternative



First technique: classical camera calibration and position estimation

World points pattern

Image points pattern

Camera calibration

$$w \cdot \begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \begin{bmatrix} x \\ y \\ w \end{bmatrix} = \begin{bmatrix} f_x & 0 & o_x & 0 \\ 0 & f_y & o_y & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} r_{11} & r_{12} & r_{13} & t_1 \\ r_{21} & r_{22} & r_{23} & t_2 \\ r_{31} & r_{32} & r_{33} & t_3 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix} = \mathbf{K} \begin{bmatrix} \mathbf{I} & | & 0 \end{bmatrix} \begin{bmatrix} \mathbf{R} & | & \mathbf{t} \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix}$$

Lens distortion effect:

- Radial:
  - $x_{dist}^{rad} = x_{norm}(1 + k_1 \cdot r^2 + k_2 \cdot r^4)$
  - $y_{dist}^{rad} = y_{norm}(1 + k_1 \cdot r^2 + k_2 \cdot r^4)$
- Tangential:
  - $x_{dist}^{tan} = x_{norm} + [2p_1 \cdot x_{norm}y_{norm} + p_2 \cdot (r^2 + 2x_{norm}^2)]$
  - $y_{dist}^{tan} = y_{norm} + [p_1 \cdot (r^2 + 2y_{norm}^2) + 2p_2 \cdot x_{norm}y_{norm}]$
- $r^2 = x_{norm}^2 + y_{norm}^2$

How is the camera made?

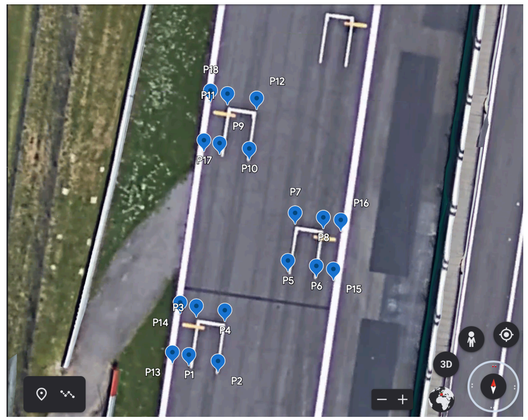
Intrinsic parameters

Extrinsic parameters

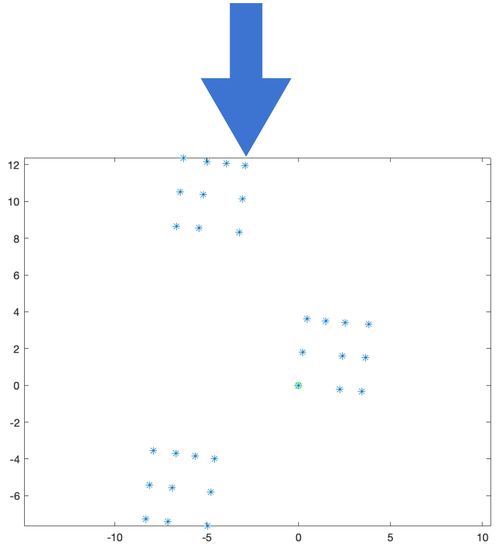
Where is the camera (and the car)?

The pattern

First choice: starting grid position



Extract latitude and longitude from Google Earth



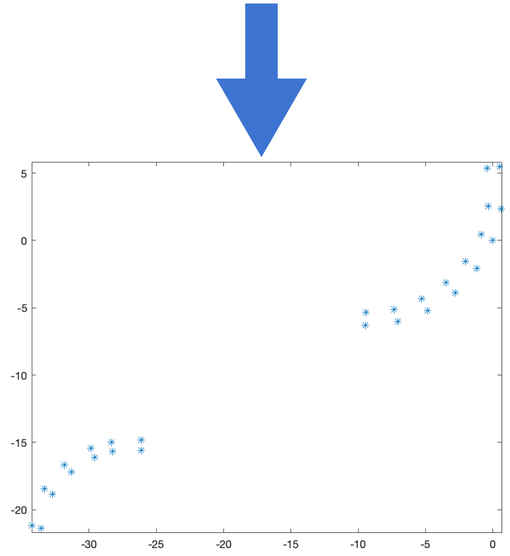
Extract the Cartesian coordinates

Pick the pixel values associated

The pattern

Second choice: chicane curves

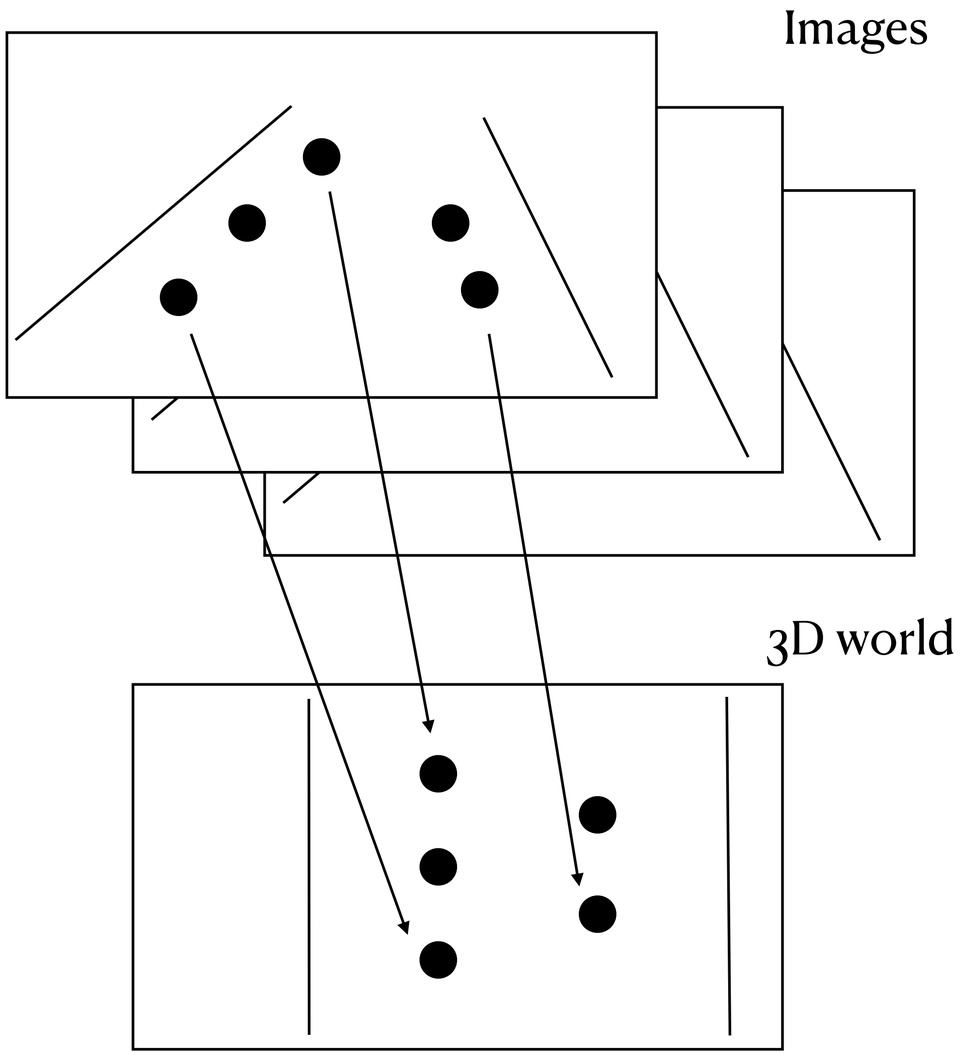
Extract latitude and longitude from Google Earth



Extract the Cartesian coordinates

Pick the pixel values associated

The procedure

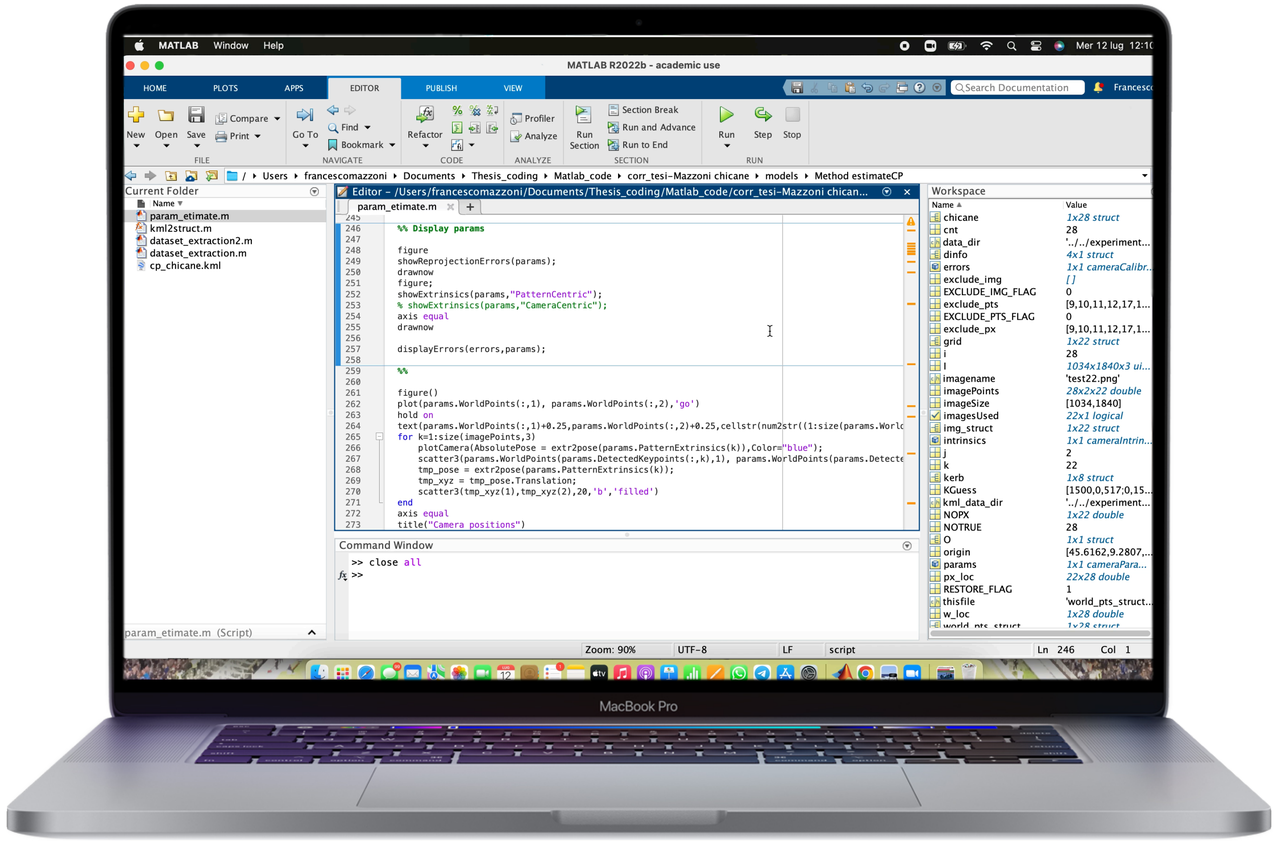


1) Suppose a pinhole camera model and points in real world on a flat surface
2) Find first guess parameters as closed form solution without lens distortion effect
3) Refine the result by minimizing

$$\sum_{i=1}^{n}\sum_{j=1}^{m} ||\mathbf{p}_{ij} - \mathbf{p}_{ij}^{proj}||^2$$

including lens distortion
4) Analyze the intrinsic parameters and the extrinsic apart

Implementation

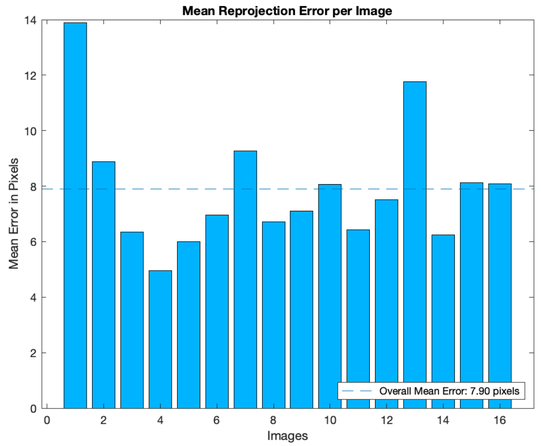


- Use Matlab and its Computer Vision toolbox
- Tune the parameter identification settings by including or not extra parameters and non-idealities
- For starting grid pattern: 16 images and 30 points
- For chicane pattern: 22 images and 28 points
- Image resolution 1034x1840

Results

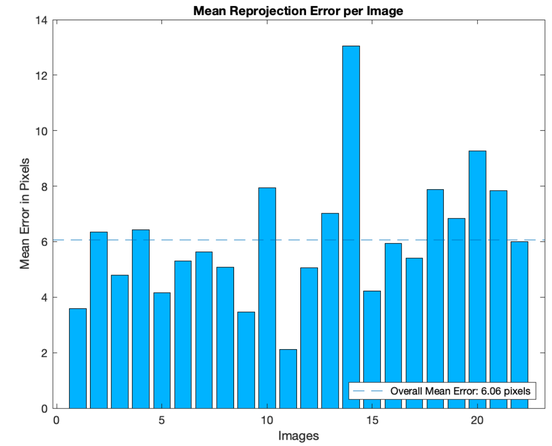
Intrinsic parameters

Starting grids



Intrinsic parameters	Nominal value
f_x [pixels]	875.3722
f_y [pixels]	99.6011
o_x [pixels]	911.3089
o_y [pixels]	364.5416
k_1	0.2099
k_2	-0.1960
k_3	0.0398
p_1	0.1063
p_2	0.0112

Chicane kerbs

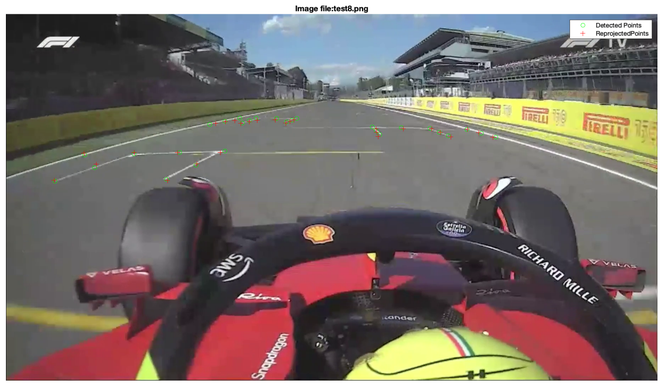


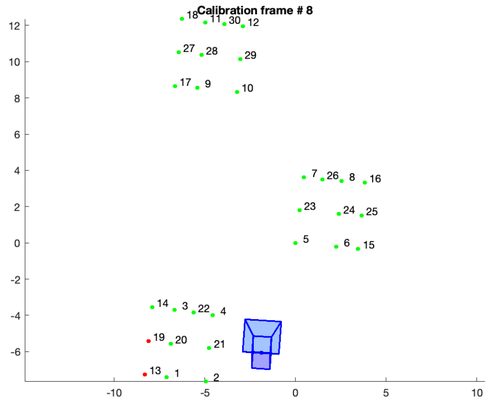
Intrinsic parameters	Nominal value
f_x [pixels]	921.3689
f_y [pixels]	398.9531
o_x [pixels]	545.2952
o_y [pixels]	470.1436
k_1	-0.1152
k_2	0.0005
k_3	0.0010

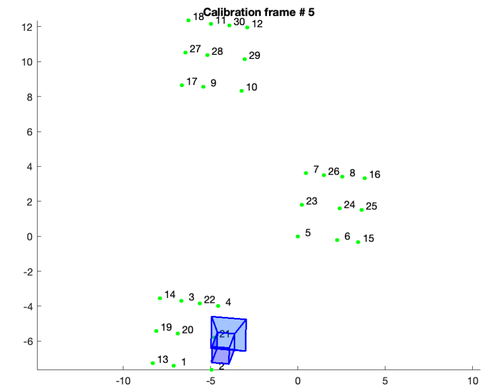
Results

Extrinsic parameters

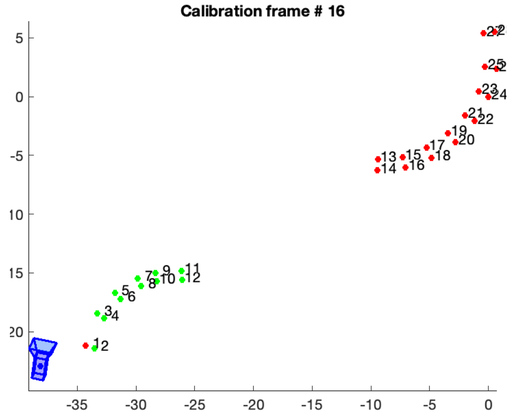
Starting grids

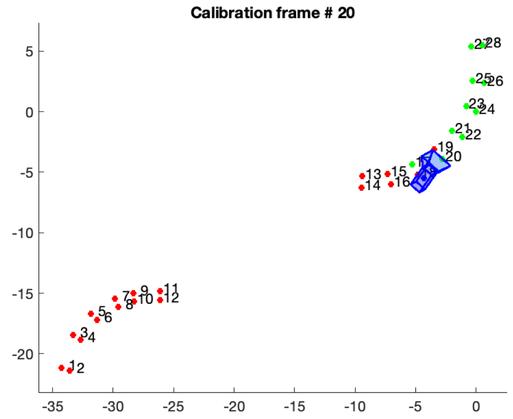






Chicane kerbs





Comments on the results

Observations

- Different results according to the settings used
- Using grids provides less uncertainty on results, but bad positioning
- Using chicanes provides more uncertainties on results, but better positioning

Comments on the results

Issue motivations and possible improvements

- Still few points and images analyzed
- Not all the pixels can be included in the analysis
- Low quality sampling
  - Use lines instead of discrete points
  - Include much more points using interpolation
  - Automatic points detection

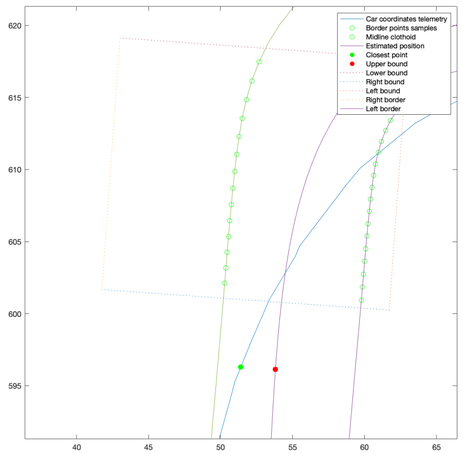
Second technique with curve matching

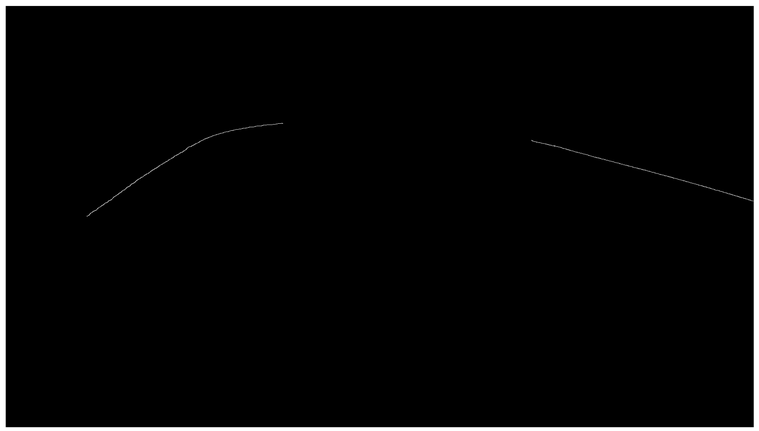
Procedure steps to follow

1. Given right, left and midline points
2. Given N_f frames with instant t_k each
3. Given partial telemetry data (position and time sampling)
4. Estimate camera position $\left(\hat{x}^k_{tlm}, \hat{y}^k_{tlm}\right)$
5. Call the midline clothoid $\left(s_k, \psi_k\right) = M\left(\hat{x}^k_{tlm}, \hat{y}^k_{tlm}\right)$
6. Extract set of pixels representing the lanes
7. Project the set of pixels in camera reference frame
8. Project world points in the neighbourhood of the estimated position in camera reference frame
9. Fit the projected points via clothoid splines and sample every $s = \ell$
10. Repeat from 3-9 for N_f times
11. Minimize the global cost function $\sum_{k=1}^{N_f} \left(\sum_{i=1}^{N_{pts,r}} (\bar{x}^W - \tilde{x}^i)^2 + (\bar{y}^W - \tilde{y}^i)^2 + \sum_{i=1}^{N_{pts,l}} (\bar{x}^W - \tilde{x}^i)^2 + (\bar{y}^W - \tilde{y}^i)^2 \right)$

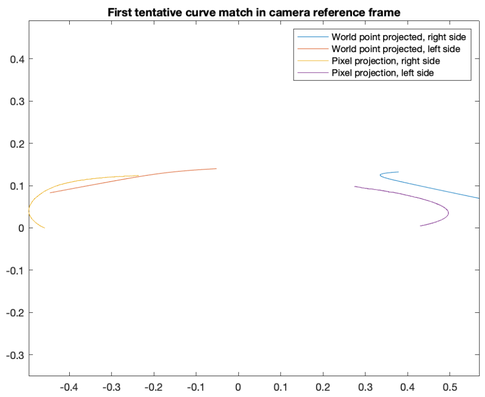
Possible implementation

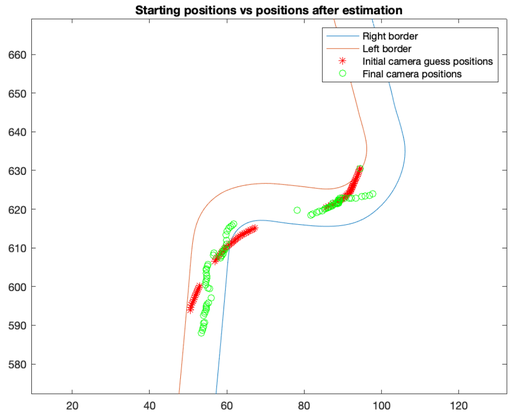
- Use Fast F1 Python library for telemetry data
- Perform a linear interpolation for a frame-by-frame position estimation
- From frames extract features with Canny edge detection algorithm
- Use the midline clothoid provided for yaw angle guess
- Fit the points with clothoids and sample every $s = \ell = 1$
- Build the cost function and solve it via patternsearch and fmincon in Matlab





First tentative results





Possible improvements:

- Refine point-matching selection
- Refine lens undistortion procedure
- Tune the look-ahead distance
- Introduce constraint equations

Thank you for your attention