HAL open science

Vision based vehicle relocalization in 3D line-feature map using Perspective-n-Line with a known vertical direction

Louis Lecrosnier, Rémi Boutteau, Pascal Vasseur, Xavier Savatier, Friedrich Fraundorfer

To cite this version:

Louis Lecrosnier, Rémi Boutteau, Pascal Vasseur, Xavier Savatier, Friedrich Fraundorfer. Vision based vehicle relocalization in 3D line-feature map using Perspective-n-Line with a known vertical direction. 2019 IEEE Intelligent Transportation Systems Conference - ITSC, Oct 2019, Auckland, New Zealand. pp.1263-1269, 10.1109/ITSC.2019.8916886 . hal-02388633

HAL Id: hal-02388633

https://hal.science/hal-02388633v1

Submitted on 2 Dec 2019

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

Vision based vehicle relocalization in 3D line-feature map using Perspective-n-Line with a known vertical direction

Louis Lecrosnier[1] , Rémi Boutteau[1], Pascal Vasseur [2], Xavier Savatier[1], Friedrich Fraundorfer [3]

***Abstract*— Common approaches for vehicle localization propose to match LiDAR data or 2D features from cameras to a prior 3D LiDAR map. Yet, these methods require both heavy computational power often provided by GPU, and a first rough localization estimate via GNSS to be performed online. Moreover, storing and accessing 3D dense LiDAR maps can be challenging in case of city-wide coverage.**

In this paper, we address the problem of camera global relocalization in a prior 3D line-feature map from a single image, in a GNSS denied context and with no prior pose estimation. We propose a dual contribution.

(1) We introduce a novel pose estimation method from lines, (i.e. Perspective-n-Line or PnL), with a known vertical direction. Our method benefits a Gauss-Newton optimization scheme to compensate the sensor-induced vertical direction errors, and refine the overall pose. Our algorithm requires at least 3 lines to output a pose (P3L) and requires no reformulation to operate with a higher number of lines.

(2) We propose a RANSAC (RANdom SAmple Consensus) 2D-3D line matching and outliers removal algorithm requiring solely one 2D-3D line pair to operate, i.e. RANSAC1. Our method reduces the number of iteration required to match features and can be easily modified to exhaustively test all feature combinations.

We evaluate the robustness of our algorithms with a synthetic data, and on a challenging sub-sequence of the KITTI dataset.

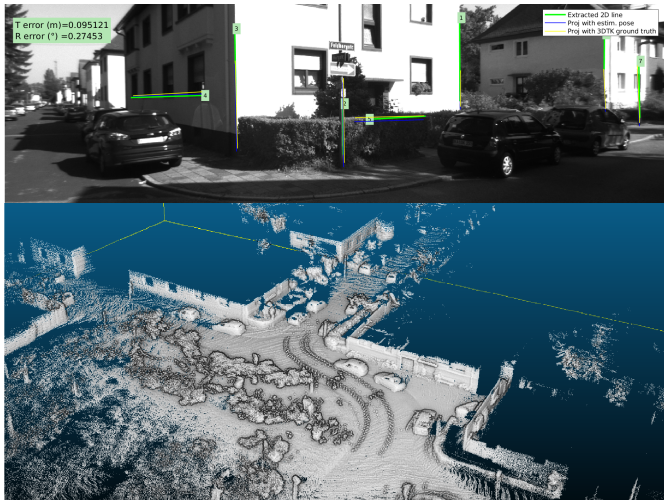

Fig. 1: Image from the vehicle (top), with manually extracted 2D features (green), reprojection of 3D features with the estimated pose (blue) and reprojected 3D features with the ground truth pose (yellow). Reconstructed 3D point cloud of the KITTI sequence used (bottom)

I. INTRODUCTION

Recently, a significant effort has been put into autonomous driving technology to meet safety and efficiency standards. One of the key aspects in autonomous driving is to accurately localize a vehicle on a local scale. Knowing this exact location and orientation, also called pose, is the first step to path planning and navigation. This step is often handled by matching sensor data against a prior map of the environment.

3D maps are commonly acquired with LiDAR (Light Detection and Ranging) sensors. Depending on the model, they output either a sparse or dense depth map of the environment in the form of a point cloud.

A common approach for vehicle localization is to equip the vehicle with a LiDAR sensor, and match the sensor data with the embed environment map. A high-end LiDAR sensor dedicated to autonomous navigation, such as the Velodyne HDL-64e, can span up to a million measurement points per second, and an acquisition of only a few minutes can lead to an uncompressed point cloud of several gigabytes. Because of the size of the point cloud map, matching directly sensor data is a challenging process. Moreover, these kind of sensors often cost several times the value of the vehicle carrying them.

While GNSS (Global Satellite Navigation System) provides reliable information about the global vehicle location, it can suffers inaccuracies of several meters on a local scale when confronted to multi-path effects in urban canyons or with low satellite coverage, making GNSS localization insufficient on its own. To increase localization accuracy, GNSS can be coupled with an inertial measurement unit (or IMU) with vehicle kinematic-aware filters, and to a ground GNSS antenna (e.g. RTK-GPS). While requiring costly hardware, this method can produce a centimetric localization, but drifts over time in case of degraded or lost GNSS signal.

Our work focuses on estimating the pose of a monocular camera in a prior 3D-lines map. When replacing a dense 3D LiDAR map with a line-feature map, we drastically reduce the storage space needed while allowing an increased overall map size. Moreover, by using geometric features such as

*This work is part of the COPTER project, and is co-funded by the European Union and the Région Normandie. Europe is involved in Normandy through the European Regional Development Fund (ERDF)

[1] are with Normandie Univ, UNIROUEN, ESIGELEC, IRSEEM, 76000 Rouen, France

[2] is with Normandie Univ, UNIROUEN, UNIHAVRE, INSA Rouen, LITIS, 76000 Rouen, France

[3]is with Institute for Computer Graphics and Vision, Graz University of Technology, 8010 Graz, Austria and German Aerospace Center (DLR), Remote Sensing Technology Institute, Germany

lines, we rely on time-invariant information. While methods focusing on photometric appearance might struggle over night and day illumination changes, building and street shape are less likely to change over a long period of time.

Because we rely solely on a monocular camera and an affordable IMU sensor for pose estimation, we avoid the substantial cost of a LiDAR sensor. Over the sequence we selected in the KITTI dataset, we show that even a minimalist 3D feature map is sufficient to estimate the vehicle pose online, without the use of a GNSS for a rough pose estimate.

In this paper, our first contribution is a perspective-n-Line algorithm relying on a prior 3D lines map and known vertical direction to accurately estimate the pose a vehicle from a single image. Our algorithm requires only 3 2D-3D line correspondence to operate, and formulates the PnL problem as a set of linear equations that are solved using a linear least squares method, followed by a Gauss-Newton-based pose refinement.

Our second contribution is a RANSAC algorithm requiring a single line correspondence to match 2D and 3D line features, i.e. RANSAC1. We combine these two approaches, and we rely on a prior 3D feature map to leverage the vehicle relocalization in a GNSS-denied environment. We evaluate our algorithms on synthetic data and on a challenging sequence of the KITTI odometry dataset.

II. Related Work

Most of the work regarding visual localization is targeted towards matching photometric features of the environment. This matching process can be either performed by comparing image descriptors, such as SURF [1] or SIFT [2], or by comparing image intensity value directly. However, this approach rapidly become ineffective when the environment appearance varies, e.g. in case of seasonal change, or with night and day lighting conditions. To solve this issue, [3] propose to store multiple images of the same place over time. [4] rely on a similarity matrix to match trajectories.

Aside from photometric feature comparison, other localization methods rely on geometric feature matching from LiDAR map. [5] localize a vehicle in an urban environment by comparing a camera image to a set of synthetic images computed from the intensity values of a LiDAR map. Final result is obtained by maximizing the mutual information, and provides a 3 DoF (degree of Freedom) pose. [6] present a 6 DoF camera pose by combining photometric and geometric data into their appearance prior map, which they use to render and match a view to a live camera image through normalized information distance minimization.

The topic of camera pose estimation from geometric features is itself a well studied subject in computer vision. However, feature matching is often assumed, which is unpractical for outdoor vehicle localization.

When related to point features, pose estimation is referred as Perspective-n-Points (or PnP) [7] [8] [9] [10]. Pose estimation from lines, or PnL, is a less studied subject. PnL methods formulate the pose estimation problem as a system of linear or polynomial equations, that is solved by minimizing an algebraic or geometric error (e.g. reprojection error).

PnL is introduced by [7] and [11] in 1989. [11] state that the problem can only be solved with a minimum of three 2D/3D line correspondences, i.e. P3L, spanning a least 8 solutions, while [7] propose the first closed-form solution to PnL with a polynomial approach.

In a more recent work, [12] propose a PnL formulation relying on a lifting approach to linearize a polynomial system formed with the elements of the rotation matrix. Using 3 line-correspondences, [13] present a PnL method spanning 23 solutions, but more robust to noise, and operating with a minimum of 3 lines. [14] introduce a P4L algorithm operating on 3-line subsets. They recover solutions from the derivatives of a 16th order cost function.

[15] extend the work from [14] with ASPnL (Accurate Subset based PnL), which is outlier-sensitive, but is more accurate with a small dataset. ASPnL includes a Gauss-Newton pose refinement.

[16] [17] present a DLT-based (Direct Linear Transform) PnL algorithm, inspired by the work of [18]. They parameterize lines using the Plücker line representation. Because their method relies on singular value decomposition to estimate the pose, is very efficient and accurate for a dataset with a very high number of lines (hundreds to thousands), but is inaccurate for a small dataset.

Some of the most recent work assume that the camera setup is coupled with an Inertial Measurement Unit (IMU), that provides a vertical direction vector, or *up-vector*. [19] introduced NPnLUpL and NPnLUpC (N-camera PnL Up-vector Linear and Cubic), and rely on modified Plücker coordinates of lines to formulate the PnL problem as a set of linear or cubic equations. Their algorithms are designed to operate in a multi-camera setup with known correspondences between 2D lines across all cameras, but can be used in a monocular setup.

While outliers rejection schemes exist within the PnL literature, feature matching is often assumed. [14] handles unknown feature correspondence using weak pose prior. [10] propose an outliers rejection algorithm for PnP based on algebraic error minimization. This method inspired the work of [15] and [17], and handles efficiently large dataset but is less accurate with small and noisy datasets.

Aside from algebraic error minimization, other outliers rejection methods rely on a statistical approach. [20] proposed the RANSAC (RAndom SAmple Consensus) algorithm in 1991. The key idea is to select a random sample of the dataset to estimate the parameters of a model and validate the parameters on the entire dataset. This process is iterated until a minimal error criterion is met. [15] present RANSAC3 and RANSAC4 algorithms, but the runtime of the two methods for a high number of lines is problematic for online applications.

In this paper, we handle both the line matching and outliers rejection process by introducing a RANSAC1 algorithm relying on a known vertical direction. By reducing the number of lines to be considered by the RANSAC algorithm

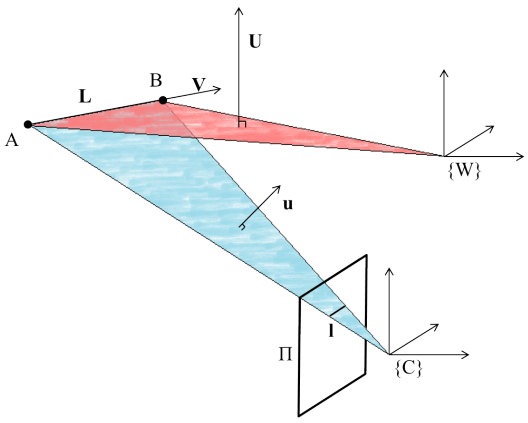

Fig. 2: 3D line and its projection onto the image plane. Points A and B form the 3D line $\mathbf{L}$ in the world frame $\{\mathrm{W}\}$. 2D line $\mathbf{l}$ in the coordinate system $\{\mathrm{C}\}$ lies at the intersection of the projection plane and the image plane Π and is defined by the normal $\mathbf{u}$ of the projection plane.

from lines, we leverage the runtime issue of the latter RANSAC3 and RANSAC4 methods. We extend our work to a pose estimation from lines algorithm, with a known vertical direction. We parameterize lines with Plücker coordinates, in the same manner as [16], but propose a linear formulation, that we solve with a least squares solver. We also propose a subsequent Gauss-Newton based pose refinement scheme.

III. The Perspective-N-Line problem

In this section, we use the pinhole camera model, and we assume that an IMU calibrated with our camera provides the vertical direction (i.e. two of the three rotations are known).

A. 3D Line parameterization

We parameterize the lines in Plücker coordinates, as shown in Fig. 2.

The Plücker representation of a 3D line $\mathbf{L}$ can be expressed from two 3D points in homogeneous coordinates $\mathbf{A} = (a_1, a_2, a_3, a_4)^T$ and $\mathbf{B} = (b_1, b_2, b_3, b_4)^T$. $\mathbf{L}$ is a homogeneous 6-vector $\mathbf{L} = (\mathbf{U}^T, \mathbf{V}^T)^T$ such as:

$$\mathbf{L} = (L_1, L_2, L_3, L_4, L_5, L_6)^T \tag{1}$$

where

$$\begin{aligned} \mathbf{U} &= (a_1, a_2, a_3) \times (b_1, b_2, b_3) \\ \mathbf{V} &= a_4.(b_1, b_2, b_3) - b_4.(a_1, a_2, a_3), \end{aligned} \tag{2}$$

and '$\times$' represents the cross-product. $\mathbf{U}$ is the normal vector to the projection plane Π. In Plücker coordinates, $\mathbf{L}$ must satisfy the bilinear constraint $\mathbf{U}^T.\mathbf{V} = 0$.

$\mathbf{L}^W$ in the world frame $\{W\}$ can be expressed in the camera frame $\{C\}$ with the line motion matrix $\mathbf{M}_{6\times 6}$ as:

$$\mathbf{L}^C = \mathbf{M}.\mathbf{L}^W \tag{3}$$

with

$$\mathbf{M} = \begin{pmatrix} \mathbf{R} & -\mathbf{R}.\mathbf{T}_{[\times]} \\ \mathbf{0}_{3\times 3} & \mathbf{R} \end{pmatrix}. \tag{4}$$

'$[\times]$' is the skew symmetric matrix of a 3-vector. $\mathbf{T} = (t_x, t_y, t_z)^T$ represents a translation vector and $\mathbf{R}$ is a 3×3 rotation matrix.

B. Line projection

The 3D line $\mathbf{l}^W$ is projected onto the image plane Π in the camera frame $\{C\}$ using the projection matrix $\mathbf{P}$ such as :

$$\mathbf{P} = \begin{pmatrix} \mathbf{R} & -\mathbf{R}.\mathbf{T}_{[\times]}. \end{pmatrix} \tag{5}$$

We obtain the projected 2D line $\mathbf{l}^C$ such as

$$\mathbf{l}^C \sim \mathbf{P}.\mathbf{L}^W, \tag{6}$$

where $\mathbf{l}^C = (l_1, l_2, l_3)^T$. We use here the same parameterization as [16], derived from [21]. This framework provides a linear projection, that can easily be used as input for a linear least squares pose solver.

C. Pose estimation with a known vertical direction

Let $\mathbf{l}_i = (l_{i1}, l_{i2}, l_{i3})^T$ be the 2D projection of the 3D line $\mathbf{L}_i = (L_{i1}, L_{i2}, L_{i3}, L_{i4}, L_{i5}, L_{i6})^T$, $\mathbf{l}_i$ and $\mathbf{L}_i$ being the ith element of a n pairs of lines dataset. We have $\mathbf{L}_i = (\mathbf{U}_i^T, \mathbf{V}_i^T)^T$.

Pose estimation through P3L requires solving a 6DoF problem, i.e estimation three rotations ρ, θ, ψ and three translations t_x, t_y, and t_z. With a prior knowledge of the up-vector, two of the three rotation are known, reducing the problem to a 4 DoF problem.

By definition, $\mathbf{L}_i^C$ lies on the projection plane of the 2D line $\mathbf{l}_i$, leading to the constraint

$$(\mathbf{R}_{CW}.\mathbf{V}_i^W)^T.\mathbf{l}_i^C = 0, \tag{7}$$

where R_{CW} is the rotation matrix composed with the remaining rotation to be determined, such as

$$\begin{aligned} \mathbf{R}_{CW} &= \mathbf{R_z}.\mathbf{R_y}.\mathbf{R_x} \\ &= \begin{pmatrix} c_z & -s_z & 0 \\ s_z & c_z & 0 \\ 0 & 0 & 1 \end{pmatrix}.R_y.R_x \end{aligned} \tag{8}$$

where $c_z = \cos(\psi)$ and $s_z = \sin(\psi)$. With a known *up-vector*, only the two unknowns c_z and s_z remain. This means that two line correspondences are required to obtain a unique solution for the complete rotation.

In a noiseless case, we directly obtain $\mathbf{R}_{CW}$. However, real data can lead to a badly scaled rotation matrix $\tilde{\mathbf{R}}$, which does not satisfy the trigonometric constraint. We enforce this constraint with a singular value decomposition of the rotation matrix $\tilde{\mathbf{R}}$, such as

$$\begin{aligned} \mathbf{U}.\Sigma.\mathbf{V}^* &= \tilde{\mathbf{R}} \\ \mathbf{R_{CW}} &= \mathbf{U}.\mathbf{V}^{*T} \end{aligned} \tag{9}$$

In order to estimate the translation vector $\mathbf{T}_{CW}$, we reformulate (6) into

$$\mathbf{l}_{[\times]}^C.\mathbf{P}.\mathbf{L}^W = \mathbf{0}_{3\times 1}. \tag{10}$$

For a single 2D/3D line correspondence, we obtain from (10) a rank deficient system of three equations [18]. We need to stack the equations from at least three 2D/3D line correspondences to obtain a rank 3 linear system and recover

the three remaining unknowns t_x, t_y and t_z. We rearrange this linear system such as

$$\mathbf{M}.\mathbf{T}_{CW} = \mathbf{N}, \tag{11}$$

each column of $\mathbf{M}$ being respectively expressed in term of t_x, t_y and t_z. $\mathbf{T}_{CW}$ is the translation matrix such as $\mathbf{T}_{CW} = (t_x, t_y, t_z)^T$, and $\mathbf{N}$ contains all terms independent from t_x, t_y and t_z. We finally solve (11) using a linear least squares method, i.e.

$$\mathbf{T}_{CW} = (\mathbf{M}^T.\mathbf{M})^{-1}\mathbf{M}^T.\mathbf{N}. \tag{12}$$

We refer to this method as VPnL_LS (Vertical Perspective-n-Line with Least Squares solver).

D. Gauss-Newton optimization for the rotation and the translation

In our PnL formulation, the rotation and translation are sequentially estimated. We propose rotation and translation refinement methods that can also be sequentially applied to the estimated pose.

Our PnL method relies on the known *up-vector* to estimate the complete pose. However, the *up-vector* is prone to sensor error. For this reason, our rotation optimization scheme refines the PnL-estimated rotation angle ψ, as well as the up vector.

Let $\mathbf{A}^W$ and $\mathbf{B}^W$ be two 3D points in the world frame defining the 3D line $\mathbf{L}^W$, and $\mathbf{l}^C$ be its projection in the camera frame. $\mathbf{V}^W$ and $\mathbf{V}^C$ are respectively the 3D line direction in the world and camera frames. For a set of n lines, we express the constraint (7) as a set of n functions $\mathbf{f} = (f_1, ..., f_n)$ depending on the three rotations ρ, θ and ψ defined as in (8), so that

$$f_i(\mathbf{R}_{CW}) = (\mathbf{R}_{CW}.\mathbf{V}_i^W)^T.\mathbf{l}_i = 0. \tag{13}$$

We can then derive this function for each rotation angle ρ, θ and ψ and form a jacobian matrix. We then proceed with a Gauss-Newton optimization scheme.

Any point included in the 3D line $\mathbf{L}^W$ is included in the projection plane of $\mathbf{l}^C$. We express this constraint similarly to (13) in a set of n functions $\mathbf{g} = (g_1, ..., g_n)$ to refine the translation estimation in

$$g_i(\mathbf{T}_{CW}) = (\mathbf{R}_{CW}.\mathbf{A}_i^W + \mathbf{T}_{CW})^T.\mathbf{l}_i = 0. \tag{14}$$

We rely on this geometric constraint to compose a second jacobian matrix, which we use to optimize the translation. When coupled with our PnL algorithm, these two optimizations are referred as VPnL_GN (GN stands for Gauss-Newton solver).

We observed that both our optimizations converge for a very small number of iterations. For this reason, we empirically choose to stop the algorithms after 20 iterations.

E. Line pairing and outliers rejection with RANSAC1

Algorithm 1 presents the line pairing process with outliers rejection. First, a random 2D-3D pair of lines is set as quadratic system of equations such as

$$\begin{cases} \mathbf{l_i}^T.\mathbf{R_{CW}}.\mathbf{V_j} \\ c_z^2 + s_z^2 = 1 \end{cases}. \tag{15}$$

Algorithm 1 RANSAC1 Pairing

procedure PAIRING
  input:
  $\mathbf{l_i}$ = normalized 2D line, $i = 1..n$
  $\mathbf{V_j}$ = normalized direction of the 3D line in Plücker coord., $j = 1..m$
  criterion = loop break threshold
  output:
  $\mathbf{R_{CW1}}$ = Rotation matrix World $\rightarrow$ Camera (first solution)
  $\mathbf{R_{CW2}}$ = Second solution
  $\boldsymbol{\varepsilon}_1$ = error vector such as $\mathbf{l_i}^T.\mathbf{R_{CW1}}.\mathbf{V_j}$, length $m.n$
  $\boldsymbol{\varepsilon}_2$ = second error vector, using $\mathbf{R_{CW2}}$
  ε_1 = first quantile of $\boldsymbol{\varepsilon}_1$
  ε_2 = first quantile of $\boldsymbol{\varepsilon}_2$

  Algorithm :
  for k = 1..m.n **do**
    From all m.n 2D-3D pair, select one at index k
    Estimate $\mathbf{R_{CW1}}$ and $\mathbf{R_{CW2}}$ with a quadratic equation solver
    From $\mathbf{R_{CW1}}$ and $\mathbf{R_{CW2}}$, estimate $\boldsymbol{\varepsilon}_1$ and $\boldsymbol{\varepsilon}_2$ $\forall$ possible pairs
    Split each error vector $\boldsymbol{\varepsilon}_1$ and $\boldsymbol{\varepsilon}_2$ into $\frac{m.n}{6}$ quantiles
    Extract ε_1 and ε_2
    if $(min(\varepsilon_1, \varepsilon_2) <$ criterion **then**
      if $min(\varepsilon_1, \varepsilon_2) = \varepsilon_1$ **then**
        return line pairs for $\boldsymbol{\varepsilon}_1 <$ criterion
      else
        return line pairs for $\boldsymbol{\varepsilon}_2 <$ criterion

Solving this quadratic system provides two rotation matrix $\mathbf{R_{CW1}}$ and $\mathbf{R_{CW2}}$ expressing rotation between the world frame to the camera frame. For each orientation proposition and all possible pairs of lines, we compute an error vector $\boldsymbol{\varepsilon}$ of size m.n.

We split $\boldsymbol{\varepsilon}_1$ and $\boldsymbol{\varepsilon}_2$ into $(m.n)/6$ quantiles, $m.n$ being the number of unique 2D/3D line combinations, and observe the value of the first quantile. If this value is smaller than the given threshold, we assume that the pairs returning the smallest error are highly probable inliers, and stop the algorithm. If not, we iterate until this condition is met, or all line pairs have been examined. In the latter case, we still return the pairs that generated the smallest error over all iterations. Note here that the number of quantiles $(m.n)/6$ is found empirically.

IV. RESULTS

A. Synthetic data

In this section, we evaluate quantitatively the performance and robustness of the different steps of our method.

a) Synthetic dataset setup: For this synthetic dataset, we simulate a 640×480 pixel camera with 655 pixel of focal length and no radial or tangential distortion.

We generate a set of random 2D pairs of points on the image plane, each pair defining a 2D line being at least 70 pixel long. Each line is expressed in metric coordinates using the camera intrinsic parameters, and is then given a depth between 2 and 20 meters. We finally express the previously obtained 3D lines in the world frame $\{W\}$ by applying the transform $\mathbf{Tr}_{CW} = [\mathbf{R}_{CW}(\rho,\theta,\psi)|\mathbf{T}_{CW}]$ with ρ, θ, ψ being random rotations between 0 rad and 2π rad, and $\mathbf{T}_{CW}$ being a random translation between 2 to 20m around the camera optical center.

In order to evaluate the robustness to noise, a normally distributed noise is added to the endpoints of 2D and 3D lines, displacing the points by respectively σ pixels and σ_{3D} mm. This modifies both the position and direction of the lines.

IMU sensors are prone to inaccuracies when measuring the vertical direction, with errors between 0.02°for high-end IMUs, to 0.5°for low-cost sensors. For a fair comparison with methods estimating the complete pose instead of relying on the vertical direction, we add a constant 0.5°noise to the up-vector used as input for the PnL algorithms.

As for metrics, we measure the translation error, ie. the euclidean distance between the estimated location and the ground truth location. We also refer to rotation error as defined in [22], ie. $R_{error} = \arccos((tr(\mathbf{R}\hat{\mathbf{R}}^T)-1)/2)$, with $\mathbf{R}$ the ground truth rotation and $\hat{\mathbf{R}}$ the estimated rotation. We consider the recall rate, ie. the number of correct pairs returned over the number of correct pairs possible for a given image, and the precision rate, ie. the number of correct pairs returned over the number of pairs returned.

We compare our PnL algorithm with optimization, VPnL_GN, and without, VPnL_LS, with the ASPnL algorithm from [15], NPnLUpL and NPnLUpC from [19]. While the two latter are designed for multiview pose estimation, they accept a single view configuration as input. ASPnL estimates the 6 DoF pose, and both our algorithms and NPnLUpL consider a 4 DoF problem.

NPnPUpL, NPnLUpC and ASPnL are known to be very sensitive to outliers, and do not include an outliers rejection scheme. For this reason, we compare our RANSAC1 algorithm to RANSAC3 and RLPnL_Enull from [15]. Because the two latter algorithms do not include feature pairing, we can only evaluate the performance against outliers. To do so, we introduce outliers into the synthetic dataset, from none to 60 %, with a fixed 2D noise of 5 pixels and an up-vector noise or 0.5 °.

b) PnL algorithm evaluation: Fig.3 show a noise robustness evaluation of our methods in comparison with ASPnL, LPnLUpL and LPnLUpC.

With our test setup, ASPnL performs better in the noiseless case, because it does not suffer the *up-vector* error. Otherwise, regarding translation error, our two PnL algorithm outperform all other considered algorithms. In term of rotation error, when confronted to 2D noise up to 20 pixels, our algorithms perform better at first, then similarly to NPnLUpL. In case of 3D noise, our two algorithms lead to similar results for noise up to 200 mm. With a higher

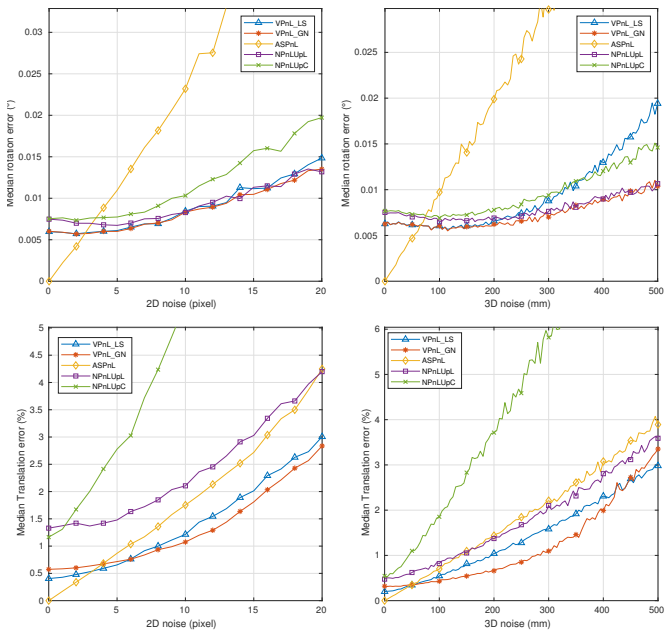

Fig. 3: Rotation and translation errors in case of 2D and 3D noises. VPnL_LS is our PnL formulation with a known vertical direction and a linear least squares solver only. VPnL_GN is the same method with Gauss-Newton optimization on the complete pose. ASPnL refers to [15] method, and NPnLUpL is the PnL linear formulation of [19] operating on a single-camera setup. Left column evaluates robustness against 2D noise, right column evaluates robustness against 3D noise.

3D noise, VPnL_LS is outperformed by both LPnLUpL and LPnLUpC, but VPnL_GN provides similar results than NPnLUpL.

In Fig.4, we show how the number of lines impacts the pose estimation in terms of rotation and translation error. For a medium 2D or 3D noise and a number of line pairs between 4 to 40, both our algorithms outperform the others. Performance increase up to 20 lines, where accuracy increases slower.

Overall, we suggest a hybrid use of VPnL_GN and VPnL_LS.

When confronted to high 3D noise (more then 500mm), rotation refinement provides a more accurate rotation estimate, while our linear VPnL_LS provides a more accurate translation.

In case of moderate 2D or 3D noise, the complete pose refinement VPnL_GN performs better.

Finally, with low 2D or 3D noise, we suggest the use of VPnL_LS.

c) RANSAC1 as outliers removal module: Fig.5 shows the robustness of our RANSAC1 algorithm to outliers in presence of moderate 2D or 3D noise and high noise on the up-vector. When combined with our two PnL methods, we observe a rotation and translation error similar to Fig.3, when RANSAC3 and RLPnL_Enull show resulst unsuitable for vehicle application. Moreover, as seen in I, our RANSAC1

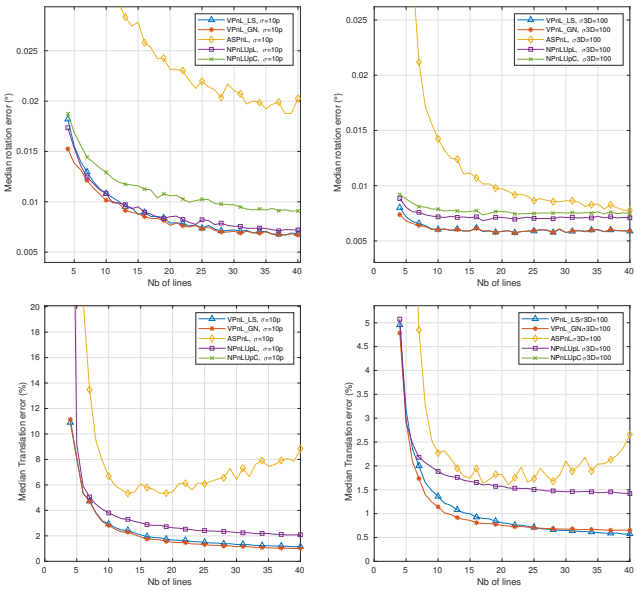

Fig. 4: Impact of the number of lines on the rotation and translation errors for 2D and 3D noise. Comparison between ASPnL, VPnL_LS, VPnL_GN, NPnLUpL and NPnLUpC. Left column evaluates the impact of the number of lines with 2D line endpoints having a 10 pixel noise. Right columns evaluates the impact of the number of lines with a 100 mm 3D noise on all 3D line endpoints.

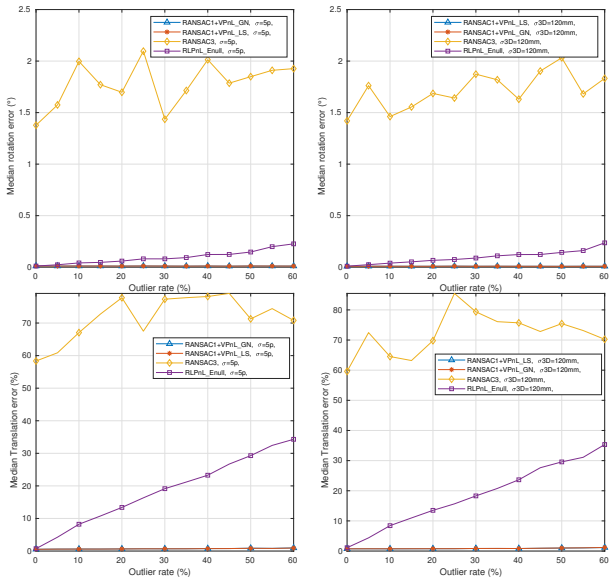

Fig. 5: Evaluation of RANSAC1, RANSAC3 and RLPnL_Enull as outliers removal module in presence of 2D and 3D noise. Constant 0.5° noise on the *up-vector*. 500 tests per outlier step.

algorithm shows runtime an order of magnitude smaller than RANSAC3, and comparable to a non-iterative algorithm such as RLPnL_Enull. Finally, we observed in our test case a 100% precision, i.e. rate of correctly matched lines for our RANSAC1 algorithm.

TABLE I: Runtime results on a simulation dataset

Method	Runtime (ms)
VPnL_LS	1.9
VPnL_GN	3.6
RLPnL_Enull	2.1
RANSAC3	45.8

B. Experiment on real data

In this section, we evaluate simultaneously our PnL method as well as our pairing/outliers rejection algorithms with real data using the KITTI dataset [23]. The KITTI dataset provides images from high resolution color and grayscale stereo camera. The image acquisition is synchronized with a Velodyne HDL-64-E, providing a 3D point cloud. Additionally, the camera pose in the world frame is provided by an OXTS RT3003 inertial and GPS navigation system.

Among the various sequences in the odometry section of KITTI, we selected in the sequence 00 a challenging sub-sequence composed 50 images with two consecutive sharp turns, taking place in an urban environment, starting from image 1223 to image 1276. We recomposed a unique 3D point cloud for the entire sub-sequence using the GPS/IMU pose. The KITTI dataset is known to be prone to GPS/IMU errors for small sub-sequences, especially in challenging consecutive turns.

For this reason, we used the 3DTK toolkit [24], that took as input the GPS/IMU poses and the multiple point clouds of our sequence, and produced a concatenated point cloud with the estimated camera poses over the sequence.

To illustrate the KITTI ground truth inaccuracies on short sequences, we compared the last pose of our sequence in the KITTI dataset to the 3DTK corresponding pose. As seen in 6, we observed a 54cm translation error and a 5.40°error.

We extracted a set of 17 3D lines on this global point cloud, creating this way a map of 3D lines which our vehicle has to localize itself in. We used our RANSAC1 algorithm to match this 3D feature set to 2D lines manually extracted from all images of our sequence. We extracted between 4 to 7 lines per image. Each extracted 2D line has a correspondence in the 3D lines map. We finally localize the camera in our 3D lines map using our pairing and pose estimation algorithms.

Because of the vehicle turns in the sequence, all 3D lines do not have 2D correspondence on each image. This means that for each pose estimation, our pairing algorithm has to deal with 58% up to 76% outliers.

To illustrate the GPS/IMU pose errors, Fig.6 compares the KITTI poses ground truth and the 3DTK poses, as well as the results of our PnL algorithm. We observe that the consecutive PnL-estimated poses are consistent with the 3DTK poses. It is to be noted that all the PnL-estimated poses are independently estimated, ie. no filter was applied over the trajectory.

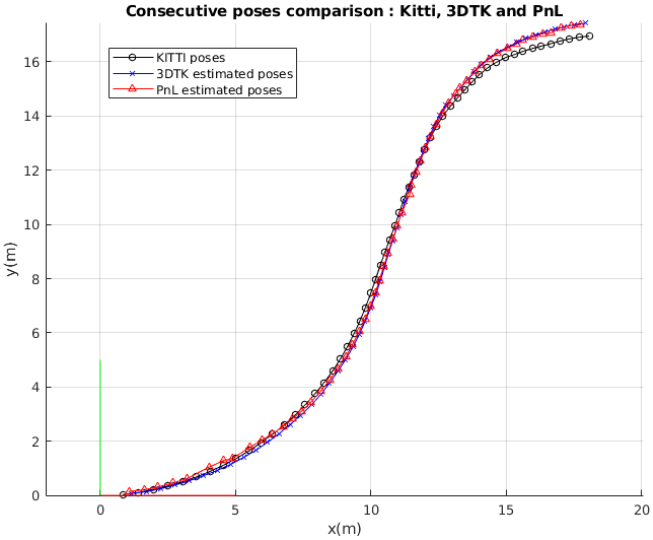

Fig. 6: Comparison between PnL-estimated trajectory and ground truth

TABLE II: Numerical results over 54 poses

Result	Value
Mean rotation error (°)	0.56
Mean translation error (m)	0.161
Mean recall rate (%)	100
Mean precision rate (%)	100
Mean runtime (ms)	13.5

Results on table II show that our PnL method coupled with our pairing algorithm achieves a mean translation error of 16.1cm, with a mean rotation error of 0.51°, for a 100% precision rate and 100% recall rate. Over the entire sequence, we measure a mean runtime for the combined line matching, pose estimation and pose refinement of 13.5 ms.

V. Conclusion

In this paper, we presented and evaluated a line-based pose estimation algorithm and a pairing/outliers removal algorithm relying on a known vertical direction. Using these two combined algorithm, we are able to localize a car in 3D lines map of an urban environment with only a very small number of 2D and 3D feature correspondences, and from a single image. Over a challenging sequence of the KITTI dataset, where the GPS/IMU provided ground truth poses with inaccuracies up to 54cm and 5°, we located successfully our vehicle within 16cm and with 0.5°error.

With a prior acquisition of a 3D feature map, our algorithms can be used to rapidly relocate a vehicle where a GNSS/IMU localization operates in degraded mode because of signal loss.

References

[1] H. Bay, T. Tuytelaars, and L. V. Gool, "Surf: Speeded up robust features," *Computer vision–ECCV 2006*, pp. 404–417, 2006.

[2] D. G. Lowe, "Distinctive image features from scale-invariant keypoints," *International journal of computer vision*, vol. 60, no. 2, pp. 91–110, 2004.

[3] W. Churchill and P. Newman, "Experience-based navigation for long-term localisation," *The International Journal of Robotics Research*, vol. 32, no. 14, pp. 1645–1661, 2013.

[4] T. Naseer, M. Ruhnke, C. Stachniss, L. Spinello, and W. Burgard, "Robust visual slam across seasons," in *2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2015, pp. 2529–2535.

[5] R. W. Wolcott and R. M. Eustice, "Visual localization within lidar maps for automated urban driving," in *Intelligent Robots and Systems (IROS 2014), 2014 IEEE/RSJ International Conference on*. IEEE, 2014, pp. 176–183.

[6] G. Pascoe, W. Maddern, and P. Newman, "Direct visual localisation and calibration for road vehicles in changing city environments," in *Proceedings of the IEEE International Conference on Computer Vision Workshops*, 2015, pp. 9–16.

[7] M. Dhome, M. Richetin, J.-T. Lapreste, and G. Rives, "Determination of the attitude of 3d objects from a single perspective view," *IEEE transactions on pattern analysis and machine intelligence*, vol. 11, no. 12, pp. 1265–1278, 1989.

[8] V. Lepetit, F. Moreno-Noguer, and P. Fua, "Epnp: An accurate o (n) solution to the pnp problem," *International journal of computer vision*, vol. 81, no. 2, pp. 155–166, 2009.

[9] J. A. Hesch and S. I. Roumeliotis, "A direct least-squares (dls) method for pnp," in *Computer Vision (ICCV), 2011 IEEE International Conference on*. IEEE, 2011, pp. 383–390.

[10] L. Ferraz, X. Binefa, and F. Moreno-Noguer, "Very fast solution to the pnp problem with algebraic outlier rejection," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2014, pp. 501–508.

[11] H. H. Chen, "Pose determination from line-to-plane correspondences: existence condition and closed-form solutions," in *Computer Vision, 1990. Proceedings, Third International Conference on*. IEEE, 1990, pp. 374–378.

[12] A. Ansar and K. Daniilidis, "Linear pose estimation from points or lines," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 25, no. 5, pp. 578–589, 2003.

[13] F. M. Mirzaei and S. I. Roumeliotis, "Globally optimal pose estimation from line correspondences," in *Robotics and Automation (ICRA), 2011 IEEE International Conference on*. IEEE, 2011, pp. 5581–5588.

[14] X. Zhang, Z. Zhang, Y. Li, X. Zhu, Q. Yu, and J. Ou, "Robust camera pose estimation from unknown or known line correspondences," *Applied optics*, vol. 51, no. 7, pp. 936–948, 2012.

[15] C. Xu, L. Zhang, L. Cheng, and R. Koch, "Pose estimation from line correspondences: A complete analysis and a series of solutions," *IEEE transactions on pattern analysis and machine intelligence*, vol. 39, no. 6, pp. 1209–1222, 2017.

[16] B. Přibyl, P. Zemčík, and M. Čadík, "Camera pose estimation from lines using plucker coordinates," *arXiv preprint arXiv:1608.02824*, 2016.

[17] ——, "Absolute pose estimation from line correspondences using direct linear transformation," *Computer Vision and Image Understanding*, vol. 161, pp. 130–144, 2017.

[18] R. Hartley and A. Zisserman, *Multiple view geometry in computer vision*. Cambridge university press, 2003.

[19] N. Horanyi and Z. Kato, "Multiview absolute pose using 3d–2d perspective line correspondences and vertical direction," in *2017 IEEE International Conference on Computer Vision Workshop (ICCVW)*. IEEE, 2017, pp. 2472–2480.

[20] M. A. Fischler and R. C. Bolles, "Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography," *Communications of the ACM*, vol. 24, no. 6, pp. 381–395, 1981.

[21] A. Bartoli and P. Sturm, "The 3d line motion matrix and alignment of line reconstructions," in *Computer Vision and Pattern Recognition, 2001. CVPR 2001. Proceedings of the 2001 IEEE Computer Society Conference on*, vol. 1. IEEE, 2001, pp. I–I.

[22] F. Huang, R. Klette, and Y.-H. Xie, "Sensor pose estimation from multi-center cylindrical panoramas," in *Pacific-Rim Symposium on Image and Video Technology*. Springer, 2009, pp. 48–59.

[23] A. Geiger, P. Lenz, and R. Urtasun, "Are we ready for autonomous driving? the kitti vision benchmark suite," in *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2012.

[24] 2019. [Online]. Available: https://sourceforge.net/projects/slam6d/