Laser-only road-vehicle localization with dual 2D push-broom LIDARS and 3D priors

Ian Baldwin
Department of Engineering Science
Oxford University
Oxford, UK
Email: ib@robots.ox.ac.uk

Paul Newman
Department of Engineering Science
Oxford University
Oxford, UK
Email: pnewman@robots.ox.ac.uk

***Abstract*—We demonstrate the viability of using 2D LIDAR data as the sole means for accurate, robust, long-term road-vehicle localization within a prior map in a complex, dynamic real-world setting.**

We utilize a dual-LIDAR system - one oriented horizontally, in order to infer vehicle linear and rotational velocity, and one declined to capture a dense view of the surrounds - that allows us to estimate both velocity and position within a prior map. We show how probabilistically modelling the noisy *local* velocity estimates from the horizontal laser feed, fusing these estimates with data from the declined LIDAR to form a dense 3D swathe and matching this swathe statistically within a map will allow for robust, long-term position estimation.

We accommodate estimation errors induced by passing vehicles, pedestrians, ground-strike etc., by learning a positional-dependent sensor model - that is, a sensor-model that varies spatially - and show that learning such a model for LIDAR data allows us to deal gracefully with the complexities of real-world data. We validate the concept over more than 9 kilometres of driven distance in and around the town of Woodstock, Oxfordshire.

I. Introduction

In this paper we consider long-term navigation using fixed 2D LIDARs, as an extension to the work in [1]. We consider how localization algorithms based on scan-matching - commonly used in indoor environments - are prone to failure when exposed to a challenging real-world outdoor environment. The driving motivation behind this work is to produce a simple, robust system that can be utilized repeatedly over a long period, rather than being forced to repeatedly map the working environment.

Fixed 2D LIDARs have several inherent advantages over 3D sensors (for example the Velodyne), in particular having lower mechanical complexity and a smaller physical footprint. Crucially in our approach, one of the lasers is oriented downwards, intentionally sampling the road surface and surrounds. This rich 3D model, built as a result of motion of the vehicle through the environment, is matched statistically within a prior map in order to generate *Special Euclidean* ($\mathbb{SE}2$) poses (position and orientation).

Given the fluid, complex nature of the world, we assert that one-shot 2D planar maps lack the required descriptiveness to enable accurate, long-term localization. In this work we leverage the accurate relative short-term consistency of scan-matching to provide **local** velocity estimates, which are then integrated with the declined LIDAR data, building a rich map swathe that is then used in an information-based map-matching algorithm that allows for repeatable, global localization in a prior map (note that we are not considering the SLAM problem). Figure 1 shows an example target environment:

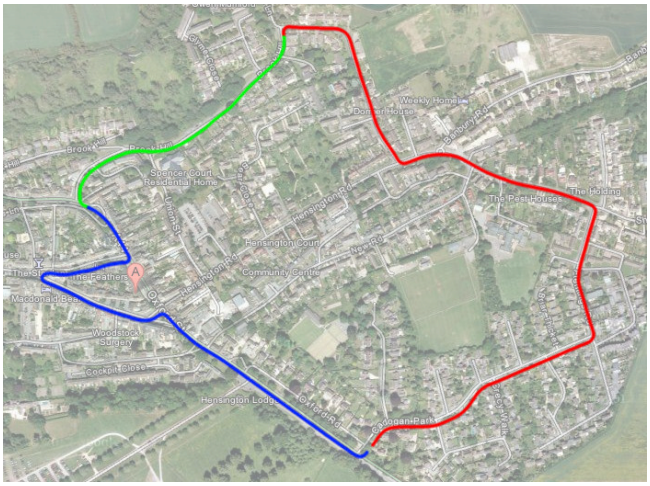


Fig. 1: An overview of the route driven in Woodstock, Oxfordshire. The route passes through a number of very different environments - suburban housing (highlighted in red, eastern section), a region with dense foliage and steep gradients (green, north) and a town-centre with a large number of pedestrians and vehicles (blue, western and southern section). Accurate navigation in these very different environments is a difficult task. (Must be viewed in colour.)

We propose a framework that will make use of the relative consistency of scan-matching, coupled with a dense 3D retrospective swathe in a map-matching algorithm that will

be robust to long-term scene changes. Simultaneously, we learn a positionally-dependent sensor model - that is, a sensor-model that has intrinsic properties that vary spatially - and show that doing so is necessary for successful robustness to short-term scene change. As validation of the concept, We show that this LIDAR-only system can provide accurate pose estimates over 9.5 kilometres of real-world data.

II. Related Work

Laser-based outdoor localization of road vehicles is most often addressed with the use of the Velodyne 3D LIDAR unit [2][3]. The authors in [4] and [5] utilize a Velodyne in addition to GPS and odometry data to generate a precise offline map, which is then matched against at run-time with a particle filter. In [6] the authors develop a system that generates multi-level surface maps, representing the environment with a 2.5D structure - again with a Velodyne. In this work, we relax the requirement for an expensive, complex actuated sensor like the Velodyne.

The authors in [7] make use of a fixed 2D LIDAR - attached to a helicopter platform - and utilize a scan-matching algorithm to generate high-quality 3D maps. Although this work is similar in spirit, the authors make use of GPS and compass data, which is not the case for our framework.

[8] et. al. utilize a robust ICP [9] algorithm to good effect, generating a robust histogram-feature representation to do local submap matching. Although in this work we make use of an ICP-derivative, it is solely in an open-loop capacity in order to infer relative vehicle motion. We do not attempt to use ICP to produce localization estimates directly, given the noise introduced into the LIDAR fan from ground strikes, vehicles and so on. The literature is replete with other 2D-LIDAR scan-matching implementations, from feature-based approaches [10], association by means of a polar representation [11], and a number of others [12][13][14][15]. In this work, we focus on learning a contextual sensor filter for any *arbitrary* scan-matching approach, as opposed to the development of a new scan-matching variant.

A comparable vision-based system from the vision community is Visual Teach and Repeat [16]. In this work, a stereo-camera is used to build a manifold world representation consisting of previously-visited submaps (*teach*), which is then used in subsequent revisits for localization (*repeat*). Due to the limited view of the camera, this approach can suffer from relatively small convergence basin, which is not the case in this approach.

III. Localisation

We assume that we have access to a previous run through the workspace by a survey vehicle - a vehicle equipped with 3D mapping and accurate pose-estimation capabilities - and that it is possible to produce a 3D point-cloud that we term $\mathcal{P}$. We seek to localize ourselves with respect to the trajectory and point-cloud - defined by this prior survey - with a run-time point-cloud $\mathcal{Q}$ - produced by developing the motion of the vehicle over an N-second retrospective window $[t_k, t_{k-N}]$. At run-time, we have access solely to LIDAR data from sensors mounted horizontally, and vertically. Figure 2 shows the layout of the sensor suite on the test vehicle, and Figure 3 depicts sample scans from this sensor layout, in addition to the run-time point-cloud:

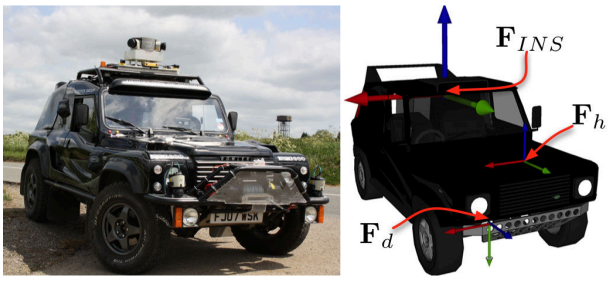


Fig. 2: The reference frames of the on-board sensors, with $\{x, y, z\}$ in red,green, and blue respectively. $\mathbf{F}_h$ corresponds to the horizontal LIDAR, $\mathbf{F}_v$ to the declined LIDAR. Example sweeps from these LIDARs are shown in Figure 3.

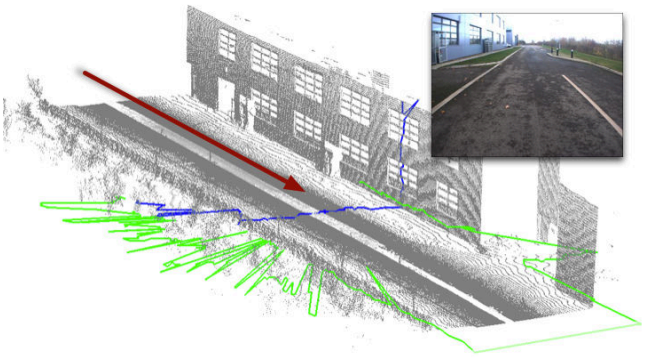


Fig. 3: A perspective view of a typical run-time generated point-cloud ($\mathcal{Q}$), with the vertical and horizontal lasers highlighted in blue and green respectively. The motion of the vehicle - generating the swathe data as it moves through the environment - is indicated by the arrow. Clearly visible in the swathe are the window frames and building edges. The inset image shows the view of the scene from the front bumper of the vehicle. This run-time cloud will be matched within the prior map $\mathcal{P}$ to provide an $\mathbb{SE}2$ pose estimate.

The tracking problem is - given the point-cloud $\mathcal{P}$, and the swathe developed during runtime, $\mathcal{Q}$ - to establish a transformation $\mathcal{T}$ that best aligns the clouds. This procedure is broken down into two components - building the run-time cloud $\mathcal{Q}$ from the observed data (Section III-A), and using this cloud in conjunction with the previous experience $\mathcal{P}$ to generate $\mathbb{SE}2$ poses (Section III-B).

A. Generating $\mathcal{Q}$

In order to generate the swathe $\mathcal{Q}$ (as visualized in Figure 3) at runtime, we need to be able to reconstruct the relative motion of the vehicle over the windowing period. The system state equation is:

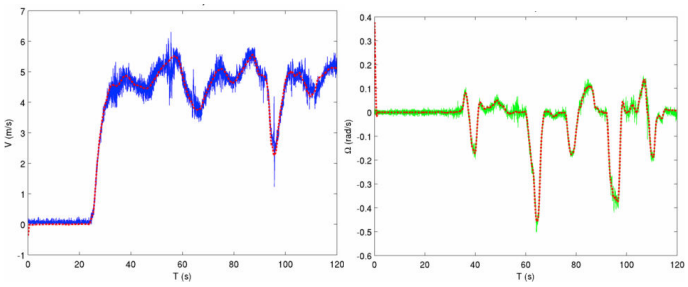
$$\dot{\mathbf{x}}(t) = v(t) \begin{bmatrix} cos(\int_{t_0}^{t} \omega_z(t)\mathrm{dt} \\ sin(\int_{t_0}^{t} \omega_z(t)\mathrm{dt} \end{bmatrix} \quad (1)$$

where $v(t)$ is the velocity in the vehicle frame, and $\omega(t)$ is the vehicle rotational velocity. By integrating the state

equation over the window period, we will be able to generate the relative vehicle motion - however we do not have direct access to either rotational or linear velocity data.

To estimate these velocities, we utilize the Iterative Closest-Surface (ICS) algorithm [17] - a variant of the standard Iterative-Closest Point [9] algorithm - to perform scan-matching between successive laser scans in the horizontal laser. Our focus in this paper is not on improving the performance of ICS, as we are using it in an open-loop capacity to estimate point-to-point velocities only.

Shown in Figure 4a is the linear velocity of the vehicle in metres per second as estimated by ICS (blue) and as measured by the on-board INS [1] (dashed red) for two minutes of driving. Similarly, Figure 4b shows the estimated (green) and actual (red) yaw velocities:



(a) (b)

Fig. 4: Estimates of both the linear (m/s) and rotational (rad/s) velocities of the vehicle over two minutes of driving, with ICS estimates (in blue and green, respectively) and ground-truth (obtained from the on-board INS) in red.

As can be seen from this figure, both linear and rotational velocities are well estimated, but noisy. We seek to model the underlying velocities in a probabilistic regression framework, a natural choice of which is the Gaussian Process (GP) [18]. If we consider the input data X as time, and the output $\mathbf{y}$ as the velocity (linear or rotational), then the GP defined for test input $X_\star$ is defined to be:

$$f_\star \mid X, \mathbf{y}, X_\star = \mathcal{N}(\bar{f}_\star, cov(f_\star)) \tag{2}$$

where $\bar{f}_\star$ and $cov(f_\star)$ are the mean and covariance functions:

$$\begin{aligned} \bar{f}_\star &\triangleq \mathbb{E}[f_\star \mid X, \mathbf{y}, X_\star] \\ &= K(X_\star, X)[K(X, X) + \sigma_n^2 I]^{-1} \end{aligned} \tag{3}$$

$$\begin{aligned} cov(f_\star) = K(X_\star, X_\star) - \\ K(X_\star, X)[K(X, X) + \sigma_n^2]^{-1} K(X, X_\star) \end{aligned} \tag{4}$$

where σ is a hyperparameter for the given kernel function. Figure 5 shows a GP trained - by maximizing the marginal log-likelihood - over a portion of the rotational velocity data in Figure 4.

Although it is computationally expensive to maintain separate processes for both the linear and rotational velocities, the windowing property renders the entire algorithm constant

[1]The INS system used is an Oxford Technologies dual-antenna 3042 unit with OmniStar HP corrections, with quoted sub-decimeter accuracy. However, as shown in [1], this bound is regularly violated.

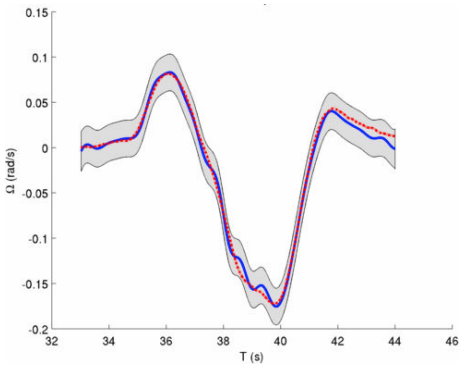


Fig. 5: The mean function of a GP (trained by maximizing the marginal log-likelihood) over a section of the rotational velocity data shown in Figure 4. Shown are the 1σ bounds, and the mean function (solid blue). Comparing the ground-truth data (dashed-red) we see that the GP mean function captures the behaviour of the rotational velocity.

complexity ($\mathcal{O}(1)$). Now that we can estimate the vehicle velocities, we turn our attention to generating the point-cloud $\mathcal{Q}$. We define a laser-scan at time t to be:

$$\mathbf{s}(t) = \{r_1, \ldots, r_{541}, i_1 \ldots i_{541}\} \tag{5}$$

where r_n is the laser range reading (in meters) for beam n of scan $\mathbf{s}(t)$, i_n is the intensity, and $\mathcal{S} = \{\mathbf{s}(1), \ldots, \mathbf{s}(n)\}$ is a set of scans. Given the estimates of the velocities from the GP mean, we can now integrate the state equation (Equation 1) to produce $\mathbb{SE}2$ poses - $\mathbf{x}_s(t)$ - for each scan. We can then project the laser data points $\mathcal{S}(t)$ from $\mathbf{x}_s(t)$, thereby generating the swathe $\mathcal{Q}$. The transformation that best aligns $\mathcal{Q}$ with $\mathcal{P}$ will be the current pose, $\mathbf{x}(t)$. We now turn to the localization procedure itself.

B. Localisation within the prior map

We provide here an overview of the localization procedure, outlined fully in [1]. Once we have developed the swathe over the window period, we need to determine the alignment with the survey point-cloud $\mathcal{P}$. We seek the transformation $\hat{\mathcal{T}}$ that brings the point-clouds $\mathcal{P}$ and $\mathcal{Q}$ into optimal alignment by minimizing an objective function f:

$$\hat{\mathcal{T}} = \underset{\mathcal{T}}{\operatorname{argmin}}\ f(\mathcal{P}, \omega, v, \mathcal{T}) \tag{6}$$

The swathe referenced within the survey, is a function of angular and linear velocity profiles, LIDAR data and the transformation we are estimating. It is factored as:

$$\mathcal{Q} \mapsto g(\omega, v, \mathcal{S}) \cdot \mathcal{T} \tag{7}$$

where Equation 7 generates the swathe $\mathcal{Q}$, and uses the transformation $\mathcal{T}$ to project it into a global frame. Since $\mathcal{P}$ and $\mathcal{Q}$ are distributions of points in space, the Kullback-Leibler divergence is a natural probabilistic method of comparing the similarity of such distributions. We develop the following cost function:

$$f(\mathcal{P}, \mathcal{Q}) = \sum_{i=1}^{N} \mathcal{H}(\mathcal{Q})(i) \log \frac{\mathcal{H}(\mathcal{Q})(i)}{\mathcal{H}(\mathcal{P})(i)} \tag{8}$$

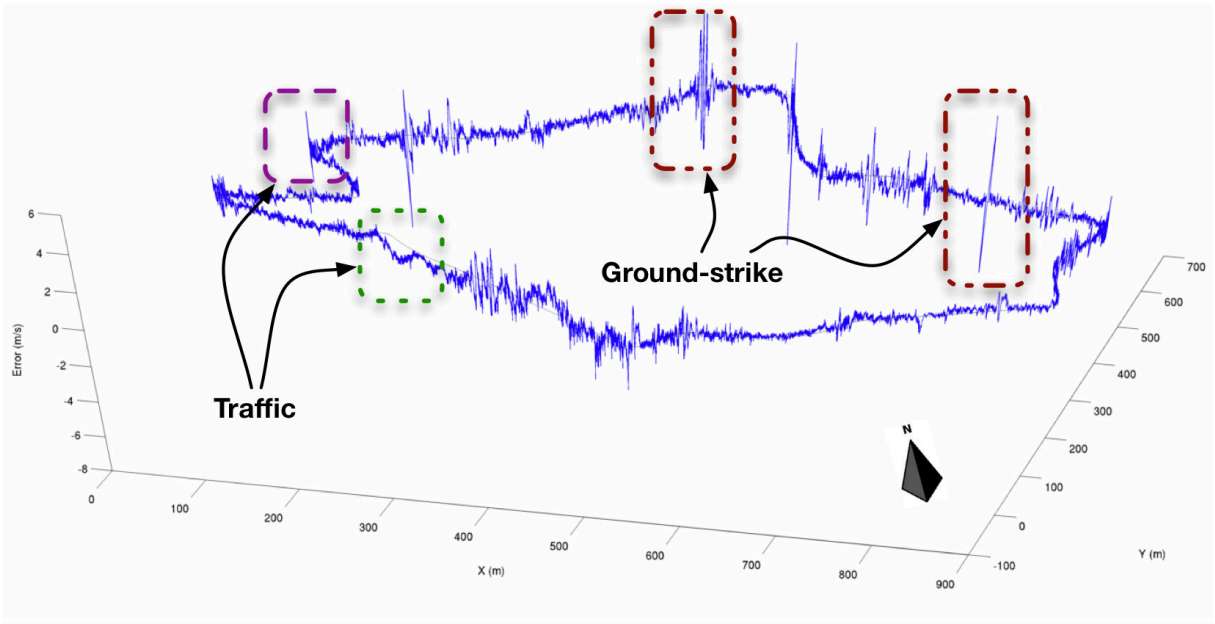


Fig. 6: A plot of the error in linear velocity from the scan-matching algorithm over the course of the Woodstock run, as compared to the commercial-grade INS system on the vehicle. In some locations, the error is in excess of $6m/s$. Typical causes of these errors - detailed in Figures 7a through 7c - are ground-strike induced from pitch and roll (dotted-dashed, top-right, top-centre), oncoming vehicles (dashed, top-left) and vehicles on the road-ahead (dotted, bottom-left). We seek to develop a positional scan-matching filter that can deal with these erroneous velocities.

where $\mathcal{H}(.)$ is used to represent the histogramming operation, and N is the cardinality of the distribution. Note that if we only require a translation and rotation then we can simply project points down into the global XY plane in order to generate the histograms. The entire localization procedure is outlined in Algorithm 1.

Algorithm 1 Localization Procedure

procedure RUNLOCALISATION($\mathcal{P}$)
  $\hat{\mathcal{T}} \leftarrow \mathcal{T}_{init}$
  loop
    $\{\mathcal{S}_h, \mathcal{S}_v\} \leftarrow (s_1^h, \ldots, s_n^h), (s_1^v, \ldots, s_n^v)$
    $\mathbf{V}, \Omega \leftarrow EstimateVelocities(\mathcal{S}_h)$
    $Q \leftarrow BuildSubmap(\mathbf{V}, \Omega, \mathcal{S}_v)$
    $\mathcal{T}_{guess} \leftarrow PredictNextPose(\hat{\mathcal{T}}, \mathbf{V}, \Omega)$
    $\hat{\mathcal{T}} \leftarrow SolveForTransformation(\mathcal{P}, \mathcal{Q}, \mathcal{T}_{guess})$
  end loop
end procedure

The algorithm is seeded with an initial pose guess, $\mathcal{T}$, at system initialisation. It is then run continuously, taking in new horizontal and vertical scan data $(\mathcal{S}_h, \mathcal{S}_v)$. The horizontal scan data is then used to estimate the linear and rotational velocities $\mathbf{V}$ and Ω by running an ICS-based scan-matcher. Once we have an estimate of the velocities, it is possible to build the local submap Q, that is then used in the pose estimation step to solve for the current best pose estimate $\mathcal{T}$.

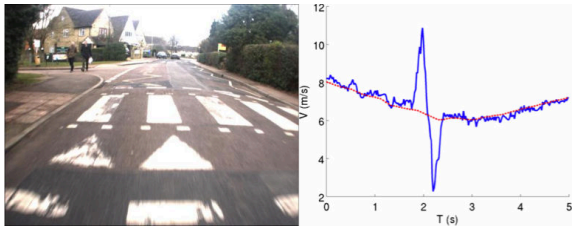
The objective function is minimized by considering the KL-divergence between the XY-projected probability distributions of the 3D points of the swathe vs. the prior point-cloud. This information-theoretic approach allows for a more robust estimator in the face of drastic long-term scene change - the approach is fully detailed in [1].

Unfortunately, the above framework is not robust to highly dynamic environments. Figure 6 shows a velocity error plot over the entire Woodstock run, obtained by comparing the velocity from the INS system against the velocity obtained from the scan-matching algorithm.

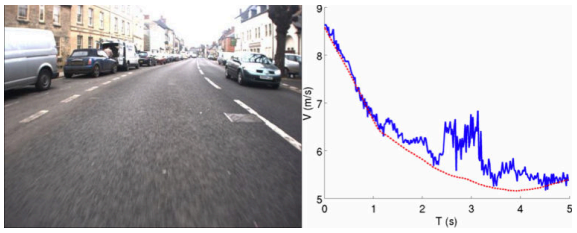
Visible are a number of regions that cause the scan-matching algorithm to perform poorly. Typical reasons for errors are from vehicle pitch/roll (Figure 7a), oncoming vehicles (Figure 7b) and cars in front of the vehicle (Figure 7c). In all these figures, estimated velocity is in (solid) blue, and ground-truth velocity in (dashed) red.

This of course leads us to believe that we require a sensor model that is capable of filtering out areas of the environment that are in motion, relative to the vehicle (traffic, people, and ground-strike as it appears in LIDAR ranging data). As an example of this, Figure 8 shows two scans - separated by one second - from the horizontal LIDAR overlaid into a common frame using the INS data.

As we can see from the figure, the majority of the scene is static in the period between the two LIDAR scans, and this is apparent from the point overlap. However, the relative velocity of the van in front leads to matching difficulties - this manifests itself in the underestimate of the true velocity in the adjacent plot of Figure 7c. To prevent this degenerate behaviour, we seek to learn a *positionally-dependent* sensor-model - that is, a model that will enable us to filter incoming scan-points, given what we know about our position in the world.



(a)



(b)

(c)

Fig. 7: (a) Velocity errors arising from ground-strike after traversing a raised pedestrian crossing, (b) velocity errors arising from the relative velocity of oncoming vehicles, (c) velocity errors arising from the relative velocity of preceding vehicles.

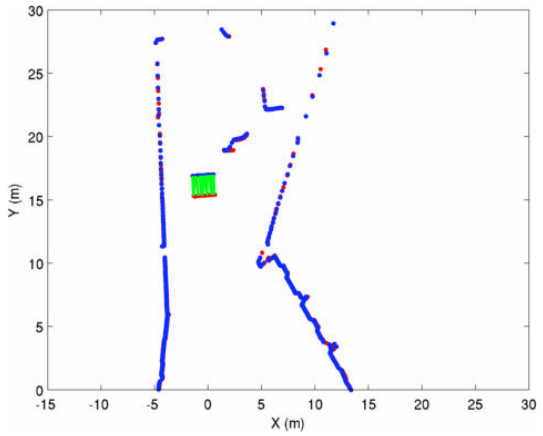


Fig. 8: A LIDAR-view of the scenario depicted in Figure 7c. Here the relative motion between moving objects - the police van, vehicle, and scene background cause points in consecutive scans to shift substantially, introducing matching difficulties. Points from one-second separated scans (red and blue, respectively) are overlaid into a global frame, and correspondences for the van are highlighted in green. As the relative velocity between the van and vehicle is less than the true velocity, velocity estimates are correspondingly suppressed, as is visible in Figure 7c.

IV. Position-Dependent Sensor Models

To correct the aberrant behaviour in Figures 7a, 7b and 7c, we need to learn a probabilistic filter that will allow us to remove points from scans that degrade the performance of the scan-matching algorithm. We seek a way of probabilistically filtering points in scans that would be good match candidates, *given where we are in the world*. We do not *require* a model for every transient obstacle that we encounter - only a way of determining good vs. bad regions of incoming scans.

We therefore introduce the notion of a position-dependent sensor model. Consider a function $\mathbf{f}$ that maps an input value λ to some output space:

$$\mathbf{f}(\lambda) \mapsto \kappa, \quad \lambda \in \mathbb{R}^m, \quad \kappa \in \mathbb{R}^n \tag{9}$$

Given that we are traversing a road network, a natural representation of this mapping is a cubic spline, which will map a floating-point value to a global UTM (x, y) position $(\mathbb{R} \mapsto \mathbb{R}^2)$.

If we consider data from the LIDAR as points in a polar representation (θ, r), we seek to learn a masking probability distribution $p(\mathbf{X} \mid \Theta, \mathbf{r}, \lambda)$, where X_i is a binary variable denoting the **transiency** of laser data observed in a discrete $\{\theta, r\}$ cell in the scan plane. To make the problem tractable, we will learn such a model for discrete locations along the spline parameterization, and therefore need a way of learning the joint distribution $p(\mathbf{X} \mid \Theta, \mathbf{r})$ for each discretization. This measure captures how reliable sensor data from a certain cell will be. Figure 9 depicts this model:

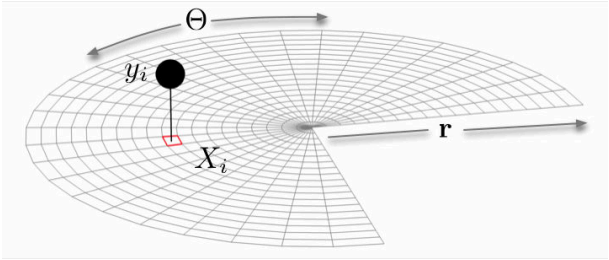
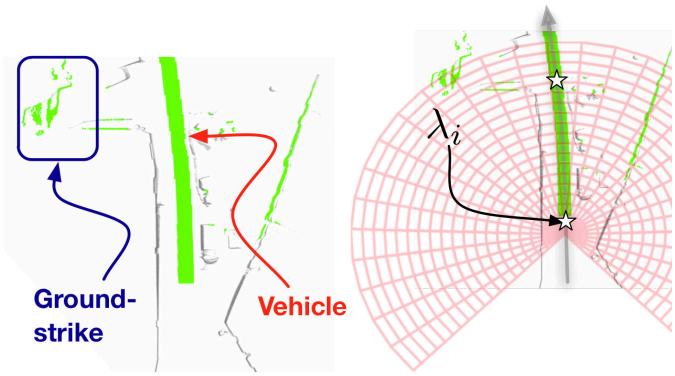


Fig. 9: A graphical representation of the model used to estimate the **transience** of areas in the LIDAR scan plane, given a certain location in the world, λ. The unobserved latent states, $\mathbf{X}$, constitute the underlying transience of a certain location in the beam plane of the LIDAR. The observed value $\mathbf{y}$ are noisy estimates from training data, estimated by observing point misalignment in consecutive laser scans.

This **transiency** measure allows us to determine, probabilistically, how much we can trust LIDAR data from a particular point in the world. We observe - during training - noisy estimates of the transiency of scan cells by overlaying points from consecutive scans using DGPS-corrected INS data. We then generate point correspondences across scans using a nearest-neighbor search and culling points closer than a certain threshold. The remaining points must therefore have moved substantially between consecutive scans. The results of this process can be seen in Figure 10.

We then histogram these observations to produce counts of per-cell transience - $\mathbf{z}$, which are then mapped into a (noisy) observation of cell transience via the following process:

$$y_i = -1 + \frac{2}{1 + e^{-\gamma |z_i - \delta|}} \tag{10}$$



(a) (b)

Fig. 10: The transiency observations for the area shown in Figure 8. By fusing INS and LIDAR data during the initial run, we can isolate areas in the environment that exhibit a high degree of transience. In addition to vehicles, ground strike (left-hand side) is also detected - this is sensible, as ground-strike can be characterized as a fast-moving obstacle. Some static structures are incorrectly observed as transients - however, the number of these observations small. Figure (b) shows a section of the approximating spline (grey) and the model domain for a particular λ value. (Must be viewed in colour.)

where Equation 10 is a generalized logistic function. z_i constitutes an observation of cell i in which points from consecutive laser scans differ by a certain tolerance, and γ and δ are scaling factors. This mapping function produces a value in the range $\{+1, -1\}$, and encodes our belief that the more motion we observe in a cell, the less likely that cell is to be a good source of static scan-match candidates. By modeling the joint distribution of latent and observed transiency as a Markov Random-Field (MRF) we seek to learn the joint distribution given some parameters $p(\mathbf{X}, \mathbf{y} \mid \theta)$:

$$E(\mathbf{X}, \mathbf{y} \mid \theta) = \theta_1 \sum_i X_i - \theta_2 \sum_{i,j} X_i, X_j - \theta_3 \sum_i X_i, y_i \tag{11}$$

$$p(\mathbf{X}, \mathbf{y} \mid \theta) = \frac{1}{\mathcal{Z}} \exp\{-E(\mathbf{X}, \mathbf{y} \mid \theta)\} \tag{12}$$

where $\mathbf{X}$ are the set of binary labels (transient/not transient), $\mathbf{y}$ are the noisy observations and Equations 11 and 12 are the energy and joint probability terms as appear in [19]. $\mathcal{Z}$ is the partition function, ensuring that Equation 12 is a valid probability distribution.

Given a set of parameters for the model, we can then apply Iterated Conditional Modes (ICM) [19] - a greedy optimization procedure - in order to infer the most probable set of labels. Given the assumption that we can factor the joint distribution as:

$$p(\mathbf{y} \mid \mathbf{X}, \theta) = \prod_i p(y_i \mid X_i, \theta) \tag{13}$$

we can, under the Markov assumption, write the distribution for latent variable X_i as:

$$p(X_i \mid y_i, \mathbf{X}_{\setminus\{i\}}) \propto p(y_i \mid X_i)\, p(X_i \mid \mathbf{X}_{\setminus\{\mathbf{i}\}}) \tag{14}$$

where the notation $\mathbf{X}_{\setminus\{i\}}$ denotes all values of $\mathbf{X}$ excluding i. ICM, from a given initial labelling greedily changes individual latent variable states until Equation 14 is maximized. We do not focus here on the parameter estimation problem - show that even with a set of sub-optimal parameters set we can perform robust localization - it is not a limiting factor. For the data gathered over the location in Figure 10 the most probable labelling of $\mathbf{X}$ is shown in Figure 11.

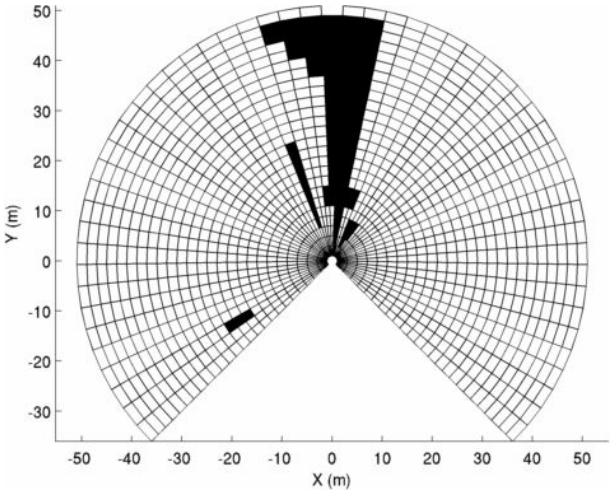


Fig. 11: The resulting locally maximal $\mathbf{X}$ given by applying ICM to the observed data for the location shown in Figure 10 over the model depicted in Figure 9. Note how the road has been learned to be an unreliable place for scan-matching, given the presence of vehicles. Also, the entryway between two buildings has also been classified as transient - this is due to repeated induced roll experienced by the vehicle at that particular point.

Note - crucially - that we have not explicitly encoded a vehicle model, but we have learned that roads are poor places to utilize scan match points. We have **also** learned that ground strike is also undesirable - Figure 11 classifies a region between two buildings as a source of transient LIDAR data. This is due to the repeatedly observed ground-strike arising from vehicle roll in that particular area.

We now use this distribution at run-time to filter out LIDAR points that have a high probability of originating from a transient object - be it a vehicle, or ground-strike from the rolling/pitching motion of the vehicle. We show how this allows us to localize the vehicle within a prior map over a testing real-world route.

V. Results

These results were obtained after running the algorithm over a 9.5km of trajectory around the Woodstock loop (depicted in Figure 1). Figure 12 shows the tracking results over the route length of 2.7 kilometres (which is traversed multiple times to obtain the 9.5km run) As can be seen from this figure, the system has managed to stay localized over the length of the route, despite the large number of transients in the environment as well as the pernicious effect of copious ground at certain places.

Figure 13 contrasts the original ICS-based velocity - shown in Figure 7c - with the correspondingly filtered output using the model in Figure 11.

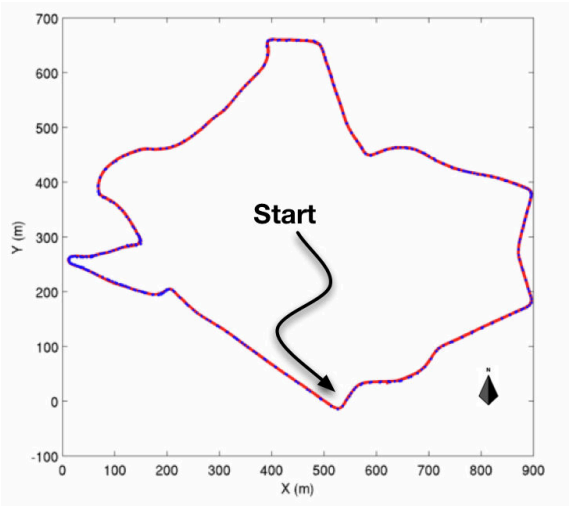


Fig. 12: Tracking results around Woodstock. The estimated trajectory is shown in red, with equally-spaced samples over time from the ground-truth INS data in blue (denser areas correspond to slower velocities - visible on the left-hand side of the image). The system stays fully localized over the length of the route. (Must be viewed in color)

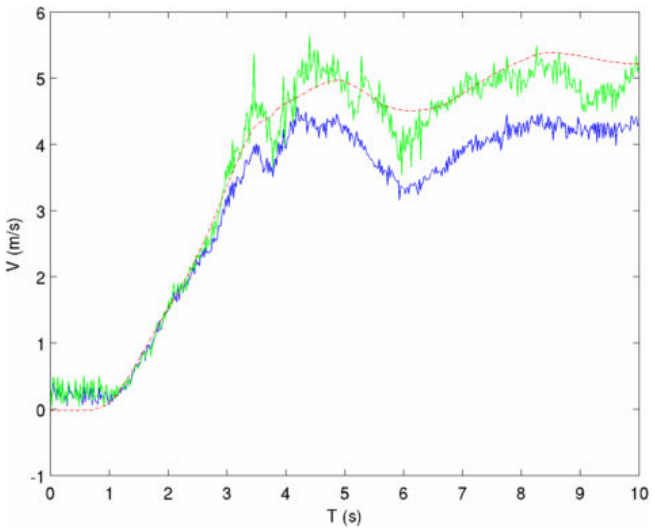


Fig. 13: A comparison of the original (blue) and filtered (green) velocity estimates for the scenario depicted in Figure 7c, using the described procedure. Learning a spatially-varying filter that can isolate regions of the environment that typically generate poor scan-matching candidates leads to better velocity-estimation.

This figure clearly shows that filtering out transient LIDAR data leads to better velocity-estimation, and hence localization. The noise of the filtered signal is somewhat higher than the original - however, this is an artefact of the hard-masking process, and is easily accommodated by using a proportional-weighting approach. This issue is not critical, as the GP regression deals comfortably with the slightly higher noise ratio.

To further illustrate the advantages of this system, we use the same measurement metric as in [1]. We define the

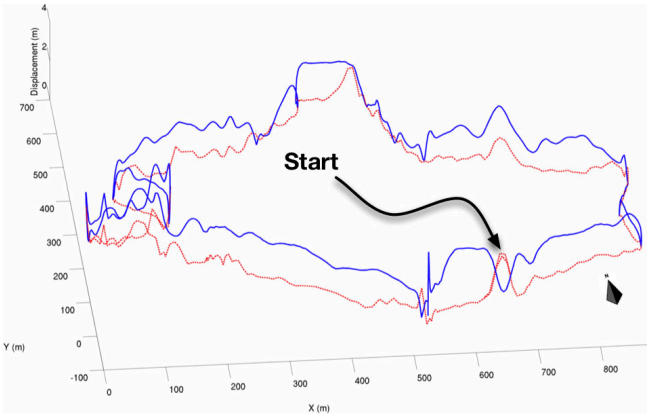


Fig. 14: Relative displacements of estimated trajectories for both the INS system (blue, solid) and the described system (red, dashed) against a reference trajectory over 2.7km. As can be seen from the figure, our system shows a lower displacement to the reference trajectory (apart from an initialization period, shown in the figure).

displacement function for a pose $\mathbf{x}(t)$ in a trajectory as:

$$\delta(\mathbf{x}(t)) = || \mathbf{x}(t) - \hat{\mathbf{x}}_{\mathcal{S}} || \tag{15}$$

where $\mathbf{x}_{\mathcal{S}}$ is defined to be the closest point in the trajectory that defined the initial survey. This metric will capture both the **true** deviation from the initial trajectory, as well as the localization error for both the INS and the described system. The crucial insight here is that, although we don't traverse *exactly* the same route twice, the trajectories are close enough that Equation 15 will allow us to quantitatively compare the performance of our system against a high-performance INS system. Figure 14 shows the comparison of trajectory estimates - relative to the survey trajectory - for both the INS and the described system over one loop (2.7km) around Woodstock.

As can be seen from Figure 14, the relative displacement using our system is consistently lower than the INS - apart from an initialization error - over the course of the route. This confirms our intuition that we will not suffer from the same long-term drift as the INS (as detailed in [1]). Figure 15 shows the averaged relative displacement over the entire 9.5km route for both the INS and the Dual-LIDAR system:

This statistic shows that on average, the Dual-LIDAR system exhibits less displacement to the reference trajectory on multiple loops than the INS.

VI. Conclusions

We have presented a system based solely on 2D LIDAR data that is capable of accurate localization within a prior map over an extended outdoor route. We have shown how using local velocity estimates from a scan-matching algorithm fused with 3D point data carved out through vehicle motion we are able to produce good localization estimates for a 9.5km run in challenging environs. We have shown that learning a spatially-varying sensor model is a prerequisite for allowing us us to estimate velocities in a much more robust fashion, leading to correspondingly better positional estimates. We quantitatively verify the performance of our

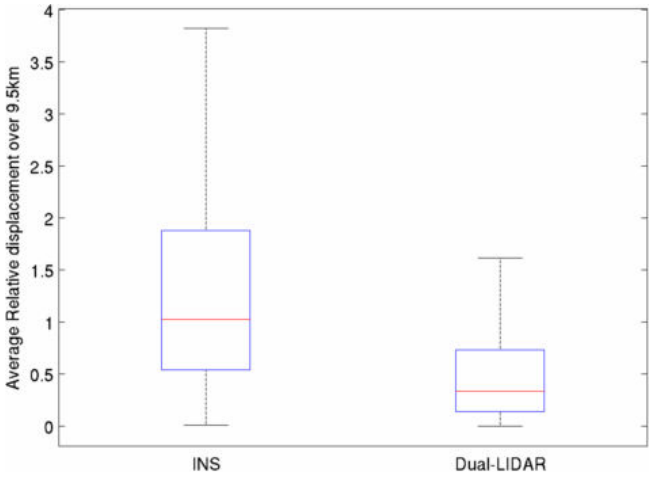


Fig. 15: Averaged displacement to a reference trajectory for both the INS and Dual-LIDAR system over the same 9.5km route. This box-plot shows that - on average - the Dual-LIDAR system exhibits less displacement.

system using a displacement metric against a commercial-grade INS, and show that the resulting pose estimates from our system exhibit a lower displacement to a reference trajectory over the driven route.

VII. Future work

In this work we have focused on learning a *positional* sensor-model - however, we are free to fold in other exogenous cues that inform the algorithm about context. For example, we could make use of one of the many vision-based car-detection frameworks as an extra input into our probabilistic filter (at the cost of adding additional sensors).

We have focused exclusively on the use of LIDAR for both velocity and positional estimates, in order to develop a stand-alone system. However, we could make use of vehicle odometry when available in order to refine our velocity estimates in degenerate areas.

In addition, we seek to learn how the probabilistic filter changes over time - for example, the effect of seasonal changes on LIDAR data and temporal changes in traffic flow - and this forms the basis for future work.

VIII. Acknowledgments

This work was funded by the Office of Naval Research (ONR) Grant N00140810337, and EPSRC Grant EP/I005021.

References

[1] I. Baldwin and P. Newman, "Road vehicle localization with 2d push-broom lidar and 3d priors," in *Robotics and Automation (ICRA), 2012 IEEE International Conference on (to appear)*. IEEE, 2012.

[2] J. Bohren, T. Foote, J. Keller, A. Kushleyev, D. Lee, A. Stewart, P. Vernaza, J. Derenick, J. Spletzer, and B. Satterfield, "Little ben: The ben franklin racing team's entry in the 2007 darpa urban challenge," *Journal of Field Robotics*, vol. 25, no. 9, pp. 598–614, 2008. [Online]. Available: http://dx.doi.org/10.1002/rob.20260

[3] F. Moosmann and T. Fraichard, "Motion estimation from range images in dynamic outdoor scenes," in *Robotics and Automation (ICRA), 2010 IEEE International Conference on*. IEEE, 2010, pp. 142–147.

[4] J. Levinson, M. Montemerlo, and S. Thrun, "Map-based precision vehicle localization in urban environments," in *Proceedings of the Robotics: Science and Systems Conference*. Citeseer, 2007.

[5] J. Levinson and S. Thrun, "Robust vehicle localization in urban environments using probabilistic maps," in *Robotics and Automation (ICRA), 2010 IEEE International Conference on*. IEEE, 2010, pp. 4372–4378.

[6] R. Kummerle, D. Hahnel, D. Dolgov, S. Thrun, and W. Burgard, "Autonomous driving in a multi-level parking structure," in *Robotics and Automation, 2009. ICRA'09. IEEE International Conference on*. IEEE, 2009, pp. 3395–3400.

[7] S. Thrun, M. Diel, and D. Hähnel, "Scan alignment and 3-d surface modeling with a helicopter platform," in *Field and Service Robotics*. Springer, 2006, pp. 287–297.

[8] M. Bosse and R. Zlot, "Map matching and data association for large-scale two-dimensional laser scan-based slam," *The International Journal of Robotics Research*, vol. 27, no. 6, pp. 667–691, 2008.

[9] P. Besl and N. McKay, "A method for registration of 3-d shapes," *IEEE Transactions on pattern analysis and machine intelligence*, pp. 239–256, 1992.

[10] Y. Li and E. Olson, "A general purpose feature extractor for light detection and ranging data," *Sensors*, vol. 10, no. 11, pp. 10 356–10 375, 2010.

[11] A. Diosi and L. Kleeman, "Fast laser scan matching using polar coordinates," *The International Journal of Robotics Research*, vol. 26, no. 10, pp. 1125–1153, 2007.

[12] P. Biber and W. Straßer, "The normal distributions transform: A new approach to laser scan matching," in *Intelligent Robots and Systems, 2003.(IROS 2003). Proceedings. 2003 IEEE/RSJ International Conference on*, vol. 3. Ieee, 2003, pp. 2743–2748.

[13] A. Censi, "An accurate closed-form estimate of ICP's covariance," in *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, Rome, Italy, Apr. 2007, pp. 3167–3172. [Online]. Available: http://purl.org/censi/2006/icpcov

[14] M. Bosse and R. Zlot, "Continuous 3d scan-matching with a spinning 2d laser," in *Robotics and Automation, 2009. ICRA'09. IEEE International Conference on*. IEEE, 2009, pp. 4312–4319.

[15] A. Fitzgibbon, "Robust registration of 2d and 3d point sets," *Image and Vision Computing*, vol. 21, no. 13-14, pp. 1145–1153, 2003.

[16] P. Furgale and T. Barfoot, "Visual teach and repeat for long-range rover autonomy," *Journal of Field Robotics*, vol. 27, no. 5, pp. 534–560, 2010.

[17] S. Rusinkiewicz and M. Levoy, "Efficient variants of the icp algorithm," in *3-D Digital Imaging and Modeling, 2001. Proceedings. Third International Conference on*. IEEE, 2001, pp. 145–152.

[18] C. Rasmussen, "Gaussian processes in machine learning," *Advanced Lectures on Machine Learning*, pp. 63–71, 2004.

[19] C. Bishop, *Pattern recognition and machine learning*. springer New York, 2006, vol. 4.