Using Laser Range Data for 3D SLAM in Outdoor Environments*

David M. Cole and Paul M. Newman
Oxford University Robotics Research Group
Department of Engineering Science
University of Oxford
Parks Road, Oxford, OX1 3PJ, UK
[dmc, pnewman]@robots.ox.ac.uk

***Abstract*— Traditional Simultaneous Localization and Mapping (SLAM) algorithms have been used to great effect in flat, indoor environments such as corridors and offices. We demonstrate that with a few augmentations, existing 2D SLAM technology can be extended to perform full 3D SLAM in less benign, outdoor, undulating environments. In particular, we will use data acquired with a 3D laser range finder. We use a simple segmentation algorithm to separate the data stream into distinct point clouds, each referenced to a vehicle position. The SLAM technique we then adopt inherits much from 2D Delayed State (or scan-matching) SLAM in that the state vector is an ever growing stack of past vehicle positions and inter-scan registrations are used to form measurements between them. The registration algorithm used is a novel combination of previous techniques carefully balancing the need for maximally wide convergence basins, robustness and speed. In addition, we introduce a novel post-registration classification technique to detect matches which have converged to incorrect local minima.**

***Index Terms*— Mobile Robotics, Outdoor 3D SLAM, 3D Laser Data, Delayed State EKF, Point Cloud Segmentation**

Fig. 1. A 3D scanning laser range finder mounted on a research vehicle.

I. Introduction and Previous Work

The SLAM problem has been thoroughly researched theoretically, and has been demonstrated many times on mobile robots in flat, 2D environments. Some successful implementations working in corridors and offices include those in [1] and [2]. Recently, some researchers have looked towards performing SLAM on mobile robots in fully 3D, outdoor environments, including work in [3] and [4].

Perhaps the most successful work to date has been in [5] and [6], which present very compelling results. However, unlike this work they do not use a probabilistic framework which if adopted can offer principled behavior in terms of error distribution when loop closing.

In this paper, we extend some of the methods used in 2D environments, and take advantage of their desirable qualities. We also offer several contributions to enable the 2D to 3D extension, which combine to form a system for SLAM in 3D, outdoor, non-flat terrain.

At present, we use laser data acquired with a custom built 3D laser range finder, along with odometry. As the vehicle moves, we divide this data into 3D point clouds, each referenced to a vehicle pose. This is achieved using a relatively straightforward segmentation algorithm, avoiding entirely the need to periodically stop and take data. As vehicle poses with attached 3D point clouds are formed, odometry provides dead-reckoned transformations between them. These are then used in augmenting a Delayed State Extended Kalman Filter (EKF) with new vehicle poses.

Consecutive point clouds are registered together, or 'scan-matched' (with the odometry derived transformation as an initial estimate) using our modified registration algorithm to provide additional observations between poses. This process can include invoking a classification technique based on a scan match's nearest neighbor statistics to detect any matches that have converged to incorrect local minima. Whilst such a strategy requires supervised training, it proves to be a useful technique for identifying poor registrations which could otherwise progress undetected.

Using existing 2D SLAM techniques has several distinct advantages, predominantly due to the probabilistic nature of the Delayed State EKF. Firstly, maintaining a state vector of poses and corresponding pose uncertainties can be used to detect potential loop closures; if for example a past vehicle

*This work is supported by EPSRC Grant #GR/S62215/01

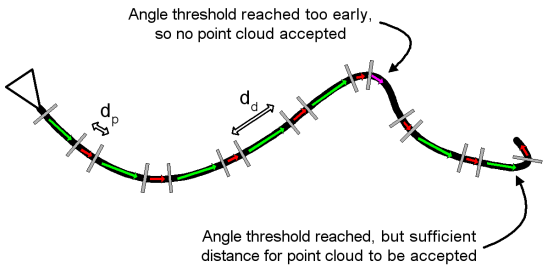

Fig. 2. The black line represents a vehicle trajectory, along which 3D laser data was taken (each scan with a different elevation angle). The grey dashes separate periodic 'potential point cloud' regions of odometry. If a change in angle threshold is not broken along a 'potential point cloud' region, OR is only broken after a point cloud has a certain number of points in it, a point cloud from that region is accepted. The point cloud generated is described relative to the vehicle pose at the beginning of the region.

pose enters the current uncertainty bounds. Secondly, if a loop closure *is* detected, and we are able to find an accurate registration derived transformation between the two poses concerned (an observation), the Delayed State Architecture means that the entire state can be updated with a single observation. The errors will then be redistributed around the loop probabilistically. Additionally, the framework offers a convenient and concise representation.

The rest of this paper is structured as follows: Section II begins by describing our segmentation algorithm, for generating 3D laser point clouds from a moving vehicle. Section III then covers the mathematics behind the Delayed State EKF SLAM algorithm. Section IV describes the registration algorithm used for generating improved pose to pose transformation 'observations', whilst Section V describes a novel technique for checking the integrity of scan matches— especially useful for confirming loop closing transformations. Section VI then shows the current mechanism used for detecting loop closures. Section VII finally describes how we combine these techniques into a functioning 3D SLAM system, and shows the results we have obtained using real data. We end with Section VIII, where we draw some conclusions regarding the system, and suggest areas for improvement and future work.

II. Segmentation

The 3D laser range finder we use consists of a standard 2D SICK scanner, continually oscillating at 0.6 Hz about a horizontal axis, as shown in Fig. 1. This is to ensure data is collected from a whole series of elevations. In [5] and [6], 3D scans were built by executing a 'stop, acquire, move' cycle. Whilst this does produce more accurately referenced point clouds, we feel that such constrained motion is awkward for general robotic applications. In this work we gather data continuously and therefore require a scheme to produce chunks (scans) of 3D data suitable for registration.

The heuristics we have chosen to achieve this produce satisfactory point clouds, each referenced to a vehicle pose, motivated by the fact that gross local odometry error is significantly correlated with changes in orientation. Other strategies, based on observing the inflation of the vehicle pose's covariance matrix have also been examined, but in practice offer little advantage.

Let us assume the vehicle begins at a 'base' vehicle pose. For a distance d_d, all scans taken (with varying elevation and odometry values) are transformed back into the base pose's frame of reference, *unless* a change in orientation threshold is exceeded. If this is the case, the construction of the current point cloud ends. If, as with the magenta arrow in Fig. 2, termination occurs when the point cloud has too few points in it, it is disregarded. Alternatively, if, despite the early termination, the point cloud is sufficiently large, as with the last green arrow in Fig. 2, it is accepted. Regardless of whether all scan planes in a distance d_d are accepted, or whether premature termination occurs, the full potential trajectory length, d_d, is measured. Upon completion, a trajectory distance d_p is observed before a new base pose and point cloud is begun.

III. Delayed State Architecture

The Delayed State or View Based Framework has been used several times in recent work in [7], [8] and [9], evolving from original work in [10], [11] and [12]. In contrast to conventional feature based approaches, the world is represented by a series of past vehicle poses with associated uncertainties. Previous observations, whether 2D laser based as in [7], or visual as in [8] and [9], are then 'attached' or associated to each pose. These can then be referred back to, compared and perhaps registered, to offer potential constraints on the global map of vehicle poses. This is most commonly done immediately after odometry-based state augmentation, but is also essential when revisiting previously traversed areas for 'loop closing'.

Generally speaking, a Delayed State or View Based Framework only refers to the representation used to build a map of the world. Such a representation could be used with a variety of estimators; for instance an Information Filter or an EKF. In this paper we arbitrarily choose to use the EKF, but may consider using other approaches in the future.

A. Delayed State Equations: State Augmentation

For full 3D SLAM, vehicle poses need 6 degrees of freedom. We choose to represent poses with 3 Euclidean coordinates, and 3 Euler angles, conforming to the classical 'Roll, Pitch, Yaw' convention. At time $t = 0$, the 6-vector global state, $\mathbf{x}$, only contains an initial vehicle pose, and the corresponding zero filled 6×6 covariance matrix, $\mathbf{P}$. Our method is similar to that described in [7] which proceeds as follows. Given a noisy control input $\mathbf{u}(k+1)$ at time $k+1$, the vehicle state evolves as shown:

$$\mathbf{x}_{v(n+1)}(k+1|k) = \mathbf{x}_{vn}(k|k) \oplus \mathbf{u}(k+1) \qquad (1)$$

Where $\mathbf{x}_{v(n+1)}(k+1|k)$ is the estimate of $\mathbf{x}_{v(n+1)}$, the $(n+1)th$ vehicle pose at time $k+1$ and the $\oplus$ operator signifies the composition operator, defined in [13]. If we also use the previous pose's covariance matrix, $\mathbf{P}_{vn}(k|k)$, and a control

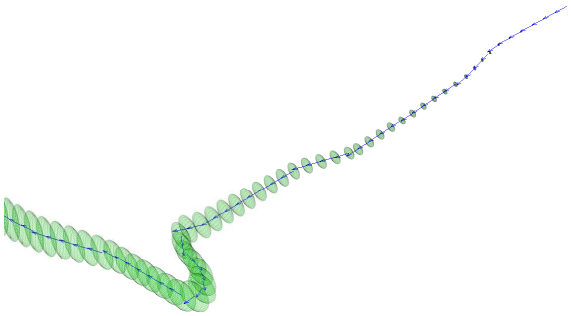

Fig. 3. Here is an example showing the evolution of vehicle poses around a building. The corresponding 'x, y, z' marginal covariance ellipsoids are also shown

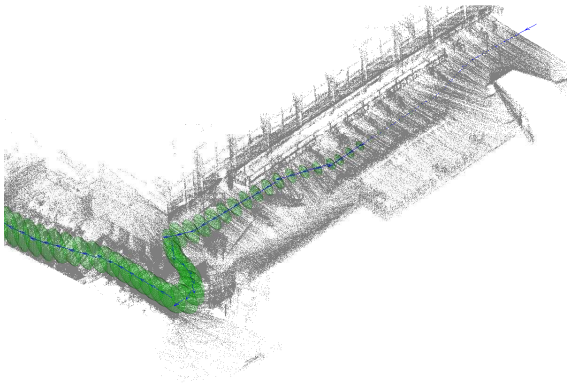

Fig. 4. Here we see the 3D point clouds corresponding to each vehicle pose. The vehicle poses and uncertainty ellipsoids are plotted again for comparison.

input noise covariance matrix, $\mathbf{U}$, the new pose's covariance matrix can be found as follows:

$$\begin{aligned}\mathbf{P}_{v(n+1)}(k+1|k) &= \mathbf{J}_1(\mathbf{x}_{vn},\mathbf{u})\mathbf{P}_{vn}(k|k)\mathbf{J}_1(\mathbf{x}_{vn},\mathbf{u})^T \\ &\quad +\mathbf{J}_2(\mathbf{x}_{vn},\mathbf{u})\mathbf{U}\mathbf{J}_2(\mathbf{x}_{vn},\mathbf{u})^T \end{aligned} \quad (2)$$

Where the k and $k+1$ have been dropped from $\mathbf{x}_{vn}$ and $\mathbf{u}$ for clarity, and $\mathbf{J}_1$ and $\mathbf{J}_2$ are jacobians of the composition operator, $\oplus$, as defined below and described explicitly in [13]:

$$\mathbf{J}_1(\mathbf{x}_1,\mathbf{x}_2) \triangleq \frac{\partial(\mathbf{x}_1 \oplus \mathbf{x}_2)}{\partial \mathbf{x}_1} \quad (3)$$

$$\mathbf{J}_2(\mathbf{x}_1,\mathbf{x}_2) \triangleq \frac{\partial(\mathbf{x}_1 \oplus \mathbf{x}_2)}{\partial \mathbf{x}_2} \quad (4)$$

Upon calculation of the new vehicle pose, $\mathbf{x}_{v(n+1)}(k+1|k)$, and a corresponding covariance matrix, $\mathbf{P}_{v(n+1)}(k+1|k)$, the global state vector, $\mathbf{x}$, and corresponding covariance matrix, $\mathbf{P}$, can be augmented as follows:

$$\begin{aligned}\mathbf{x}(k+1|k) &= \begin{bmatrix}\mathbf{x}(k|k) \\ \mathbf{x}_{vn}(k|k) \oplus \mathbf{u}(k+1)\end{bmatrix} \quad (5)\\ &= \begin{bmatrix}\mathbf{x}_{v0} \\ \vdots \\ \mathbf{x}_{vn} \\ \mathbf{x}_{v(n+1)}\end{bmatrix}(k+1|k) \quad (6)\end{aligned}$$

$$\mathbf{P}(k+1|k) =$$

$$\begin{bmatrix}\mathbf{P}(k|k) & \mathbf{P}(k|k)\mathbf{J}_1(\mathbf{x}_{vn}(k|k),\mathbf{u})^T \\ \mathbf{J}_1(\mathbf{x}_{vn}(k|k),\mathbf{u})\mathbf{P}(k|k)^T & \mathbf{P}_{v(n+1)}(k+1|k)\end{bmatrix} \quad (7)$$

Fig. 3 shows such equations applied to real data from around a building. We can see the evolution of the 6 degree of freedom poses, along with the corresponding 'x, y, z' marginal uncertainty ellipsoids. Fig. 4 also shows the point clouds attached to each of the poses.

B. Delayed State Equations: State Update

Let us assume that an inter-pose registration has been triggered at time k between two scans, $\mathbf{S}_i$ and $\mathbf{S}_j$, the first belonging to pose $\mathbf{x}_v(i)$ from time i, the second belonging to pose $\mathbf{x}_v(j)$ from time j. This will yield an improved transformation estimate between the two poses.

This is most frequently encountered when registering point clouds from consecutive poses, which is performed after all state augmentations. In this instance, the immediately previous pose would be from time i, whilst the current pose would be from time j (ie.$i = k-1$ and $j = k$). However, it *could* be a loop closing transformation. If this is the case, the earliest pose would be from time j, whilst the current pose would be from time i (ie.$i = k$ and $j < k$). In actual fact, an observation between two poses could theoretically be postponed and implemented later - when computation allows (ie. $k \neq i$ *or* j).

Both poses are present in the global state vector, $\mathbf{x}$, and therefore a predicted 'measurement' between the two poses can be found from the observation model as follows:

$$\begin{aligned}\mathbf{T}_{i,j}(k+1|k) &= \mathbf{h}(\mathbf{x}(k+1|k)) \\ &= \ominus(\ominus\mathbf{x}_{vj}(k+1|k) \\ &\quad \oplus\mathbf{x}_{vi}(k+1|k)) \end{aligned} \quad (8)$$

Where the $\ominus$ operator signifies the inverse transformation operator, as defined in [13], and $\mathbf{x}_{vi}(k+1|k)$ and $\mathbf{x}_{vj}(k+1|k)$ refer to the $t = i$ and $t = j$ vehicle poses in $\mathbf{x}(k+1|k)$ respectively. This is then used as the initial estimate for our registration algorithm as follows (as well as the predicted measurement in the update equations below):

$$\mathbf{T}_{i,j}(k+1) = \Psi(\mathbf{T}_{i,j}(k+1|k), \mathbf{S}_i, \mathbf{S}_j) \quad (9)$$

Where Ψ represents our registration algorithm, detailed in Section IV. The state update equations are then identical to the conventional EKF update equations. The innovation ν and corresponding covariance $\mathbf{S}$ can be found as follows:

$$\nu = \mathbf{T}_{i,j}(k+1) - \mathbf{T}_{i,j}(k+1|k) \quad (10)$$

$$\mathbf{S} = \nabla\mathbf{H}_x\mathbf{P}(k+1|k)\nabla\mathbf{H}_x^T + \Delta \tag{11}$$

Where Δ is a registration transformation covariance matrix[1] and ∇H_x is the jacobian of the observation model defined as follows:

$$\nabla\mathbf{H}_x = \frac{\partial\mathbf{h}}{\partial\mathbf{x}} \triangleq \begin{bmatrix} \frac{\partial\mathbf{h}}{\partial\mathbf{x}}_{q=0} \\ \frac{\partial\mathbf{h}}{\partial\mathbf{x}}_{q=1} \\ \vdots \\ \frac{\partial\mathbf{h}}{\partial\mathbf{x}}_{q=n} \end{bmatrix}^T \tag{12}$$

Where:

$$\frac{\partial\mathbf{h}}{\partial\mathbf{x}}_q = \begin{cases} \mathbf{0} & \text{if } q \neq i,j \\ \mathbf{J}_1(\ominus\mathbf{x}_{vi}(k+1|k), \mathbf{x}_{vj}(k+1|k)) \\ \quad \times\mathbf{J}_\ominus(\mathbf{x}_{vi}(k+1|k)) & \text{if } q = i \\ \mathbf{J}_2(\ominus\mathbf{x}_{vi}(k+1|k), \mathbf{x}_{vj}(k+1|k)) & \text{if } q = j \end{cases}$$

Where $\mathbf{J}_\ominus$ is the jacobian of the inverse transformation operator, as defined below and described explicitly in [13]:

$$\mathbf{J}_\ominus(\mathbf{x}_1) \triangleq \frac{\partial(\ominus\mathbf{x}_1)}{\partial\mathbf{x}_1} \tag{13}$$

And finally, the global state and covariance update equations can be used as standard:

$$\mathbf{x}(k+1|k+1) = \mathbf{x}(k+1|k) + \mathbf{W}\nu \tag{14}$$

$$\mathbf{P}(k+1|k+1) = \mathbf{P}(k+1|k) - \mathbf{W}\mathbf{S}\mathbf{W}^T \tag{15}$$

Where the Kalman gain, $\mathbf{W}$, is:

$$\mathbf{W} = \mathbf{P}(k+1|k)\nabla\mathbf{H}^T\mathbf{S}^{-1} \tag{16}$$

Inputs : Model Point Cloud Ω_m, Data Point Cloud Ω_d, Initial Transformation $\mathbf{T_0}$, Initial LM parameter λ_0, Termination Threshold r_T

Output: Final Transformation $\mathbf{T}$

$\lambda = \lambda_0$;

Λ=BuildTree(Ω_m);

$\mathbf{T} = \mathbf{T}_0$;

while $|(||\mathbf{r}'||_2 - ||\mathbf{r}||_2)| < r_T$ **do**

  Ω'_d=$\mathbf{T}$(Ω_d);

  $\mathbf{r}$=GetResiduals(Ω'_d,Λ);

  $\mathbf{J}$=Get_dr_by_dT($\mathbf{T}$,Ω_d,Λ);

  %Use LM to step towards a minimum

  $\Delta\mathbf{T} = (\mathbf{J}^T\mathbf{J} + \lambda\mathbf{I})^{-1}\mathbf{J}^T\mathbf{r}$;

  $\mathbf{T_p} = \mathbf{T} + \Delta\mathbf{T}$;

  Ω'_d=$\mathbf{T_p}$(Ω_d);

  $\mathbf{r}'$=GetResiduals(Ω'_d,Λ);

  if $||\mathbf{r}'||_2 > ||\mathbf{r}||_2$ **then**

    $\lambda = \lambda \times 100$;

  else

    $\lambda = \lambda \div 100$;

    $\mathbf{T} = \mathbf{T_p}$;

  end

end

Algorithm 1: Our registration technique: a unique combination of previous methods. An ICP-like cost function defined between a model and data scan is minimized. However, a Geman-McClure non-linear kernel is included for robustness. Subsequent minimization is then performed using a Levenberg-Marquardt non-linear optimizer. Approximate kd-trees are also used, providing a data structure for efficient nearest-neighbor correspondence searches.

IV. Registration

There has been a large amount of literature published on registration algorithms over the past 15 years, and several SLAM related papers use variants of such algorithms such as [5], [6] and [7]. The problem is generally posed as a search for a transformation T, to transform a data 'scan' or set of points Ω_d (denoted as $\mathbf{T}(\Omega_d)$), onto a stationary model 'scan', termed Ω_m. This is frequently achieved by minimizing a cost function based on the trial transformations and original points. Depending on the type of data being registered, such algorithms fall into two main categories.

In [2] and [7], the registration of 2D laser range scans is performed using a *semi-exhaustive* search (ie. exhaustive over a local region), and whilst expensive, the small, 3-dimensional transformation search space make these techniques viable.

Alternatively, when registering 3D scans, more *directed* searches are favored, given the higher 6 dimensional transformation search space. Often, the cost function is based on the sum of the square of distances between corresponding points. The classical Iterative Closest Point (ICP) algorithm, introduced by Besl and McKay in [14] does exactly this, with the nearest model point used as each data point's correspondence. Horn's closed form minimization [15] is subsequently used to find a minimizing transformation, *given* that set of correspondences, before new correspondences based on that transformation are then found. This process is repeated until convergence. This is the main method used in the work by Surmann, Nüchter and Hertzberg [5] and Surmann, Nüchter, Lingemann and Hertzberg [6]. However, we have found that this algorithm does not always perform well when used in this domain. This is because the two scans being matched often contain *different* objects. The quadratic cost function can be overly biased by such outlying points, resulting in convergence to an undesired minimum.

Consequently, we use an error metric similar to that of the ICP algorithm, but with a Geman-McClure robust kernel (as illustrated in Fig. 5). This effectively transforms the Euclidean distance between corresponding points (x) into a score for the

[1]At present we use an average registration derived transformation covariance. It would be preferable to perturb the transformation around the converged registration minimum, sampling the cost function surface multiple times. A quadric could then be fitted to the surface, to estimate the transformation's covariance.

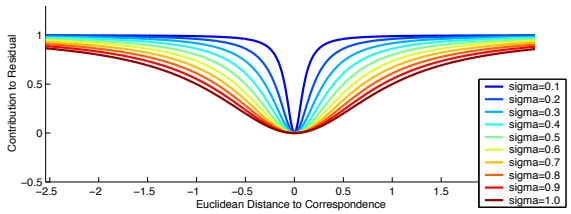

Fig. 5. The general shape of the Geman-McClure robust kernel. Note the variation in profile, for different values of the σ parameter. The optimal choice for σ generally depends on the density of the scans being registered, and the degree of overlap between them. At present we have no analytic method for making such a choice, and it consequently requires some experimentation. However, this is an area of ongoing research.

vector of residuals (denoted $\rho(x)$), and is defined as follows:

$$\rho(x) = \frac{x^2}{\sigma^2 + x^2} \tag{17}$$

Unfortunately this also means that Horn's closed form minimization no longer applies, and we have to use Levenberg-Marquardt non-linear optimization to find a minimum. This is similar to the technique used by Fitzgibbon in [16]. We also accelerate our registration algorithm, using an efficient kd-tree data structure to make correspondence searches fast, as Besl and McKay [14] recommended. In addition, we use approximate kd-tree querying, as suggested originally by Greenspan and Yurick [17], for further saving.

The full algorithm is summarized in algorithm 1, where inputs and outputs have been defined. However, the following intermediate variables are also used: λ, the current Levenberg-Marquardt parameter; Λ, the model point kd-tree; Ω'_d, a set of transformed data points; $\mathbf{r}$, a vector of residuals; $\mathbf{J}$, the Jacobian of the residual vector $\mathbf{r}$, with respect to the current transformation $\mathbf{T}$; $\Delta\mathbf{T}$, a putative transformation change; $\mathbf{T_p}$, a putative transformation and finally, $\mathbf{r}'$, a vector of putative residuals. In addition, note that the Geman-McClure robust kernel is not referred to explicitly as it has been incorporated into the 'GetResiduals' function call.

Several practical issues arise when using this algorithm. The first; choosing a value for λ_0, the initial Levenberg-Marquardt parameter value, is not particularly critical. This is because it only affects the rate of convergence of our algorithm, and does not alter the solution obtained. With experimentation, we have found a value of 10^6 to work relatively well, but in general this depends on the exact nature and initial transformation estimate of the point clouds used.

The second; selecting a value for the residual termination threshold, r_T, is slightly more important. One solution we favor is to use a value commensurate with the machine's precision, although higher values can be used if required, trading off accuracy for increased speed.

V. Registration Integrity Check

Whilst the Levenberg-Marquardt approach offers significant computational advantages over the exhaustive search variety, it can often fall into unwanted local minima. This is especially

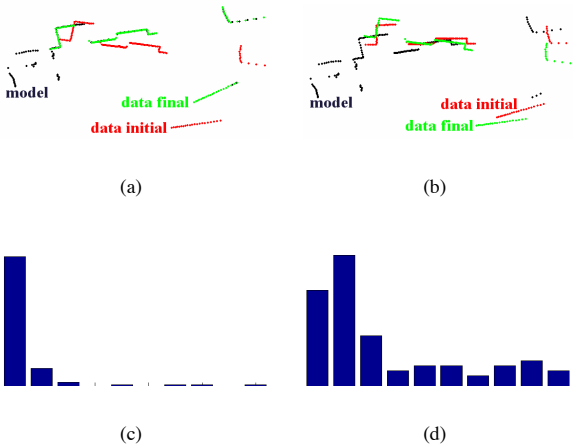

Fig. 6. Example training data, showing a successful and unsuccessful scan match (top). The black points represent the model scans, red represents the initial data scans and green represents the converged data scans. Below are the corresponding nearest-neighbor histograms, beginning with 0-0.5m on the left hand side, with each successive bin of 0.5m width (in this example).

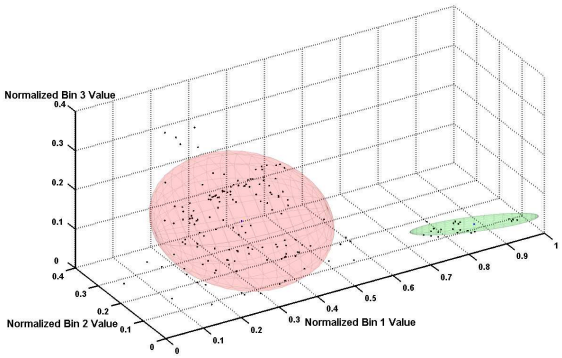

Fig. 7. A plot showing 2D scan-match training data, including the 2D examples from Fig. 6. The three axes represent values from the first three histogram bins. The green ellipsoid shows the successful scan match Gaussian's 1σ bound, whilst the red ellipsoid shows that for the unsuccessful matches.

true when the initial transformation estimate between scans is poor, and additionally when there is low overlap between them. We have developed a method, to recognize, *post-registration*, when this has occurred. Our method uses supervised learning to train a classifier based on scan match's nearest-neighbor statistics. Whilst this requires a set of hand labelled good and bad scan matches, along with the corresponding nearest-neighbor statistics, once trained, it does offer an accurate, fast and robust mechanism for rejecting poor scan-matches.

We begin with a series of point cloud pairs derived from real data. The data set of each point cloud pair is then artificially perturbed relative to the corresponding model set, using a series of transformations. The resulting set of point cloud pairs, each with multiple relative starting positions, are in turn, input to our registration algorithm. Upon termination, a histogram of nearest neighbor distances is calculated for each scan match.

This histogram approximates the distribution of the distances between each point in the data scan, and its nearest neighbor in the model scan. This is illustrated in Fig. 6. The top two sub-figures show model points in black, and the initial position of the data points in red. Once registered, the converged data points are plotted in green. (a) shows a pair of scans which converged to the correct minima, while (b) shows a pair that did not. The corresponding histograms are also plotted in (c) and (d). We can see a high peak in (c), in a low distance bin, as the scan match was successful, and many data points had very close model points. Conversely, (d) shows that when registration had failed, a different histogram resulted. More of the data points lay in more distant, intermediate bins, as they did not have close correspondences to the model scan. It is worth noting that the *furthest* histogram bins, should *not* be used in this learning algorithm. This is because these will be predominantly made up of the points which lack *any* correspondence between scans, ie. the non-overlapping sections.

Each scan-match is then hand-labelled successful or not. Once a whole series of these have been labelled, we fit Gaussians to the distributions of histograms corresponding to 'good' matches, and to 'bad' matches (as each histogram represents a point in high dimensional space). This is illustrated, in 3 dimensions, in Fig. 7. Here, the axes indicate the values of the first 3 histogram bins for training data generated with the simple 2D example looked at earlier. Each point corresponds to a scan-match and a resulting histogram. We can also observe the 1σ covariance ellipsoids for the successful and unsuccessful matches.

Once the classifier is trained, scan match classification is a straightforward task. This is because the parameters for the two Gaussians fitted are known. The histogram of a novel scan match can be found, and used as an input to our Gaussian models which return probabilities of a successful or unsuccessful matching. At present we then take the category with greatest probability as the correct classification.

VI. Detecting Loop Closures

We presently use our estimate of position and its corresponding uncertainty to prompt loop closures. Unfortunately, this strategy is far from ideal, as our maintained Gaussian Probability Density Function (PDF) diverges from the actual one. This is primarily due to compounded linearizations, but is also due to incomplete knowledge of transformation's covariance matrices (if they have been found via registration). This could potentially lead to falsely triggered loop closure detection, or worse still, miss loops that need detecting. This paper focuses on the details of the laser registration and associated SLAM formulation, not on the loop closing problem. We refer the reader to the companion paper [18] for a full description of a robust, multi-modal way to detect and affect loop closure, which is then used in conjunction with the techniques discussed in this paper.

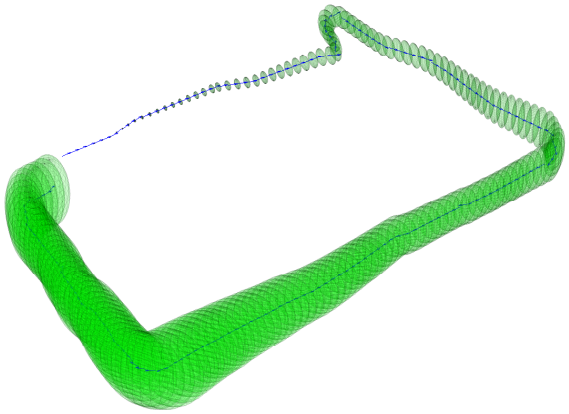

Fig. 8. Here we can see the entire state of vehicle poses, immediately prior to loop closure detection, and the subsequent loop closure itself. The corresponding 'x, y, z' marginal covariance ellipsoids for each pose are also shown. Notice the vertical discrepancy between the first and last pose due to accumulated error, even though the vehicle was in approximately the same location.

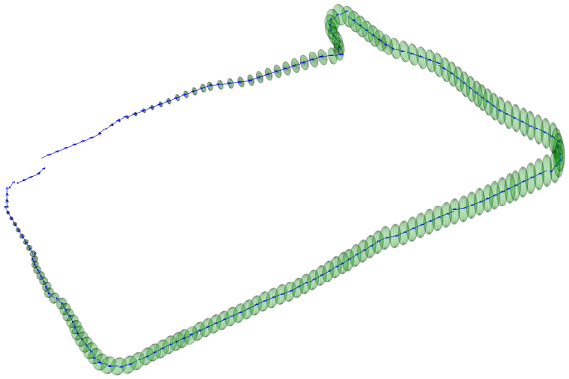

Fig. 9. Here are all of the state's vehicle poses, immediately after loop closure. The corresponding 'x, y, z' marginal covariance ellipsoids for each pose, suitably reduced, are also shown. Notice that the first and last poses do not line up exactly, as the vehicle had a slightly different position on its return to the initial region.

VII. Results

A vehicle was driven around the exterior of a medium sized building on a smooth but non-flat surface, to test our system's performance. Fig. 8 shows the evolution of vehicle poses around the building, along with corresponding 'x, y, z' marginal covariance ellipsoids. In particular, notice the large 2.2 meter vertical discrepancy between the first and last pose, caused by accumulated error.

After the state was augmented with the last pose shown in Fig. 8, the simple loop closure detection stage found that a previous pose, here the very first pose, fell close to it. This prompted a registration between the point clouds belonging to those poses. Due to the large accumulated error in the initial transformation used to seed the registration, a poor scan

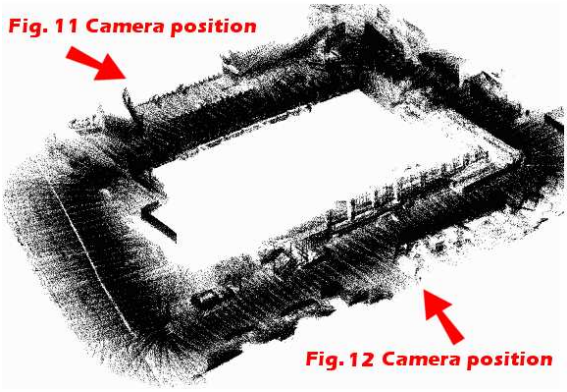

Fig. 10. The entire state's attached point clouds plotted simultaneously. The windows of the building can be made out in the foreground, along with a small fire escape staircase. The strong white line on the left of the figure is laser shadow, caused by the roadside kerb. Camera locations for the subsequent close up shots are also shown.

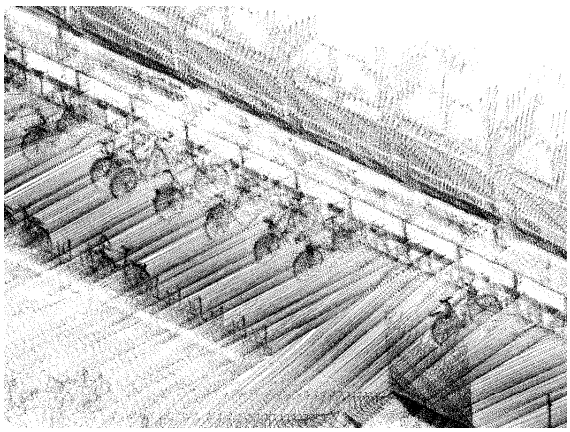

Fig. 11. In this close up of the post loop closure point clouds we can make out the facade of a building, along with its windows and perimeter railing. We can also identify individual bicycles in the adjacent bicycle rack.

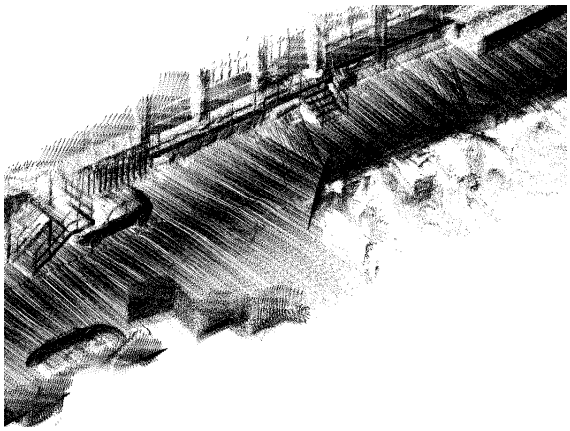

Fig. 12. This close up shows the opposite side of the building. Identifiable features include an emergency staircase, a parked vehicle, and as we move to the right of the image, more of the building facade.

match resulted. A remedy for this is to repetitively perturb the initial transformation until a satisfactory (see Section V) registration is achieved. The registration-derived transformation then served as an observation to update the state, and 'close the loop'. The resulting state's poses, along with *reduced size* 'x, y, z' marginal covariance ellipsoids, are shown in Fig. 9. Finally, the point clouds corresponding to all of these poses are plotted together in Fig. 10, with close up views in Fig. 11 and 12.

VIII. Conclusions and Future Work

We have demonstrated that by extending existing 2D SLAM techniques, and augmenting them with a data segmentation and scan match classification stage, we can perform 3D probabilistic SLAM in outdoor terrain.

We have presented an algorithm for segmenting 3D laser range data from a moving platform into distinct 3D point clouds referenced to vehicle poses. Odometry derived inter-pose transformations can then be used to augment a Delayed State EKF with new six degree of freedom vehicle poses. After each state augmentation, the consecutive point clouds can be registered together, using odometry as an initial estimate, in a fast, robust and reliable manner using a 6 degree of freedom registration algorithm. The improved accuracy transformations can then be used to update the Delayed State EKF. We believe this is the first time 3D laser range data has been used in this formulation, which redistributes errors probabilistically over the vehicle's past trajectory. Additionally we have presented a classification based integrity check for detecting scan matches which have converged to incorrect local minima. This is of particular importance when initial transformation estimates are in error.

One logical extension to the techniques described in this paper would be to incorporate the integrity check's quality metric into the registration process itself. This would widen and deepen the convergence basin leading to superior convergence properties. Additionally, further work is required to detect possible loop closures when traversing very large loops. This becomes critical over large distances, as accumulated linearization error can cause the method described in Section VI to fail and miss loop closing opportunities entirely. One possible remedy for this is discussed in a companion paper [18].

References

[1] D. Hähnel, D. Fox, W. Burgard, and S. Thrun, "An efficient FastSLAM algorithm for generating maps of large-scale cyclic environments from raw laser range measurements," in *Proceedings of the IEEE/RSJ Conference on Intelligent Robots and Systems (IROS)*, vol. 1, Las Vegas Nevada USA, 27-31 October 2003, pp. 206 – 211.

[2] J. Gutmann and K. Konolige, "Incremental mapping of large cyclic environments," in *Proceedings of the International Symposium on Computational Intelligence in Robotics and Automation (CIRA)*, Monterey California USA, 8-9 November 1999, pp. 318 – 325.

[3] A. J. Davison and N. Kita, "3D simultaneous localisation and map-building using active vision for a robot moving on undulating terrain," in *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR)*, vol. 1, Hawaii USA, 11-13 December 2001, pp. 384–391.

[4] J. Weingarten and R. Siegwart, "EKF-based 3D SLAM for structured environment reconstruction," in *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Edmonton Alberta Canada, 2-6 August 2005, pp. 2089 – 2094.

[5] H. Surmann, A. Nuchter, and J. Hertzberg, "An autonomous mobile robot with a 3D laser range finder for 3D exploration and digitalization of indoor environments," *Robotics and Autonomous Systems*, vol. 45, pp. 181–198, 2003.

[6] H. Surmann, A. Nuchter, K. Lingemann, and J. Hertzberg, "6D SLAM - preliminary report on closing the loop in six dimensions," in *Proceedings of the 5th IFAC/EURON Symposium on Intelligent Autonomous Vehicles (IAV)*, Lisbon Portugal, 5-7 July 2004.

[7] P. Newman and K. Ho, "SLAM - loop closing with visually salient features," in *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, 18-22 April 2005, pp. 635 – 642.

[8] R. Eustice, H. Singh, and J. Leonard, "Exactly sparse delayed-state filters," in *Proceedings of the 2005 IEEE International Conference on Robotics and Automation (ICRA)*, Barcelona Spain, April 2005, pp. 2428–2435.

[9] R. Eustice, H. Singh, J. Leonard, M. Walter, and R. Ballard, "Visually navigating the RMS Titanic with SLAM information filters," in *Proceedings of Robotics: Science and Systems*, Cambridge MA USA, June 2005.

[10] F. Lu and E. Milios, "Globally consistent range scan alignment for environment mapping," *Autonomous Robots*, vol. 4, no. 4, pp. 333–349, 1997.

[11] P. F. McLauchlan, "A batch/recursive algorithm for 3D scene reconstruction," in *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR)*, Hilton Head SC USA, 13-15 June 2000, pp. 2738–2743.

[12] J. Leonard, R. Rikoski, P. Newman, and M. Bosse, "Mapping partially observable features from multiple uncertain vantage points," *The International Journal of Robotics Research*, vol. 21, no. 10, pp. 943–975, 2002.

[13] R. Smith, M. Self, and P. Cheeseman, "Estimating uncertain spatial relationships in robotics," *Autonomous robot vehicles*, pp. 167–193, 1990.

[14] P. J. Besl and N. D. McKay, "A method for registration of 3-d shapes," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 14, no. 2, pp. 239–256, 1992.

[15] B. K. P. Horn, "Closed-form solution of absolute orientation using unit quaternions," *J. Opt. Soc. Amer.*, vol. 4, no. 4, pp. 629–642, 1987.

[16] A. W. Fitzgibbon, "Robust registration of 2d and 3d point sets," *Image and Vision Computing*, vol. 21, pp. 1145–1153, 2003.

[17] M. A. Greenspan and M. Yurick, "An approximate K-D tree search for efficient ICP," in *4th International Conference on 3-D Digital Imaging and Modeling (3DIM03)*, Banff Alberta Canada, 6-10 October 2003, pp. 442– 448.

[18] P. M. Newman, D. M. Cole, and K. Ho, "Outdoor SLAM using visual appearance and laser ranging," in *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, Orlando Florida USA, May 15-19 2006.