A 3D Line Alignment Method for Loop Closure and Mutual Localisation in Limited Resourced MAVs

Ilankaikone Senthooran  
Faculty of Information Technology  
Monash University  
Clayton VIC 3800, Australia  
Email: ilankaikone.senthooran@monash.edu

Jan Carlo Barca  
Faculty of Information Technology  
Monash University  
Clayton VIC 3800, Australia  
Email: jan.barca@monash.edu

Hoam Chung  
Department of Mechanical and Aerospace Engineering  
Monash University  
Clayton VIC 3800, Australia  
Email: hoam.chung@monash.edu

Abstract—In this paper we present a new 3D line alignment technique that can be used in limited resourced MAVs for performing loop closures as well as mutual localisation between MAVs. We identify pairs of 3D line matches from RGB-D frames and find the optimal transformation which aligns these two line sets onto each other in order to find the corresponding relative pose. The alignment of the two 3D line sets are achieved by converting each line match into two matching point pairs and then subjecting them to a least squares minimisation process. As we only maintain line features extracted from key frames, our method does not require large memory, high processing power nor a high communication bandwidth between robots. We validate our 3D line alignment technique by performing loop closures and mutual localisation on real-world datasets.

I. INTRODUCTION

Due to their superior manoeuvrability, Micro Aerial Vehicles (MAVs) have the potential for being used for numerous indoor applications such as surveillance, monitoring, collapsed building exploration and aiding in disaster relief operations. However, their lower payload capacity limits the sensors and the processing power that can be carried on-board. Moreover, they are only able to fly short distances due to energy constraints. A multi-robot system may provide a solution to this problem, as they can coordinate to cover a wider area. In addition to this higher efficiency due to the ability to execute tasks in parallel and distributed sensing, Swarm Robotic (SR) systems also provide robustness, scalability and flexibility which are very important for real-world applications.

Due to the leaderless, distributed nature of the control algorithms of SR systems, the failure of one robot does not lead to mission failure. The performance of these systems are not seriously affected by the number of robots because their control algorithms are based on local sensing and local communications. Their distributed sensing enables swarms to detect changes in the task and environment and adapt accordingly. This flexibility allows SR systems to be used for tasks of different size as well as tasks that change over time.

The small size, light weight and low price of RGB-D cameras make them ideal for use in limited-resourced MAVs. Moreover, RGB-D cameras are independent modules directly providing rich 3D information about its environment [1], and do not require any further processing to calculate depth information, such as triangulation in stereo cameras.

A single robot performing pose estimations usually reduce the accumulated estimation error due to drift when they revisit a previously seen scene. However, with multiple robots exploring the same environment, the area covered by a single robot is reduced and this reduces the accumulated estimation error. However, this also makes it less likely for a robot to visit the same scene more than once, thus limiting the usefulness of self-loop closures. The robots can nevertheless, correct one another’s estimations based on mutual localisations. Performing mutual localisation between two robots gives an additional piece of information (constraint) which can be used to correct their estimations when their initial poses are known.

This paper is a continuation of our previous work [2], where we introduced an efficient pose estimation method using RGB-D sensors which enables autonomous flight of a limited-resourced MAV by providing adequately fast and accurate estimates for real-time navigation. Here, we propose a 3D line alignment technique that can complement our earlier point feature based pose estimations by detecting mutual localisation between two MAVs as well as self-loop closures. We assume that the MAVs share a common coordinate system.

Recent developments in line feature extraction and matching methods (Line Segment Detector-LSD [3], MeanStandard deviation Line Descriptor-MSLD [4], Scale-invariant MeanStandard deviation Line Descriptor-SMSLD [5], EDLines [6], Line Band Descriptor-LBD [7]) have made line based localisation popular among researchers. As lines contain more structural information about the environment than point features, they can produce higher quality maps even with a fewer number of features. This enables place recognition, which can be used for correcting estimation errors through mutual localisation of robots observing the same scene.

Most existing work on onboard MAV localisation and mapping with RGB-D sensors [8], [9] maintain RGB-D image pairs on all key frames. Considering resource-limitations on our MAVs we needed a less memory-intensive mapping technique. In our proposed method we do not maintain whole images, but only the line features extracted from key frames. This not only reduces the memory requirement but also the amount of information communicated between MAVs for mutual localisation and collaborative mapping.

Some methods [10], [11] use line features represented as
segments with two endpoints for localisation and mapping. The problem with this line model is that end points are not as reliable as orientation of line segments due to reasons such as occlusions and fragmentation during line extraction. However, modelling 3D lines as infinite lines [12] eliminates problems associated with the ambiguity of the end points.

In [13], vertical line and floor lines are adopted for 3-DoF SLAM on a ground moving robot. They introduce the method using ‘vanishing points’ for reducing the accumulation of heading error as well as for loop closing. Inspired by this, StructSLAM [14] introduces the use of building structure lines as novel features complemented with point features for 6-DoF localisation and mapping. Taking the Manhattan-world assumption into consideration, they identify the lines that can be aligned with one of the three dominant directions (represented as vanishing points) as structural lines. They use a global constraint on camera orientations in order to eliminate accumulated orientation errors and thereby reduce the position drift.

In [15], an indoor place recognition technique has been proposed which extracts lines from each scene and registers the corresponding image to the database using a vocabulary tree approach. When the robot revisits a location, the database returns a match and based on this, the robot corrects its pose estimation. However, this method requires a trained database and considerable amount of memory.

In a recent work [16], multiple RGB-D cameras have been used on ground moving robots to establish initial relative poses of robots with shared field-of-views. Each robot extracts point features from the viewed scene and transmits the feature descriptors and corresponding depth information to a centralised node with high computational power for processing. It is at this central node that point feature matching is performed to establish the initial relative poses of robots. These poses are then transmitted back to the robots and a pose refinement step is performed on-board. As this method of relative pose estimation requires a centralised node with high computational power to establish an initial estimation, it is not suitable for swarm robotic systems made up of limited resourced robots. Centralisation limits the scalability of the system and with the reliance on a single node comes the risks of a single point of failure, thus reducing robustness.

Recently visual relative localisation has been used in a swarm of MAVs. In [17] employs pattern recognition and decentralised rules to maintain a compact formation in a swarm of MAVs. Unlike this method where a geometric pattern fixed on the MAVs is employed for the localisation, we use structural features already available in the environment for localisation between the robots. Moreover, in our scenario, the MAVs do not fly in a formation as they need to cover as large an area as possible within their limited flight time. In our assumptions the robots use mutual localisation in order to improve their pose estimations.

The main contribution of this paper is a new 3D line matching and alignment makes this method applicable to identifying the relative geometric transformation between images frames taken by different robots, given that they contain a common region of the environment. We demonstrate this by applying it for single-robot loop closure as well as multi-robot mutual localisation and mapping.

The rest of this paper is organised as follows: Section II gives a detailed description of the proposed method. In this section we explain our 3D line alignment technique (II-B) and then how this alignment is used in loop closures (II-C) and mutual localisation between two robots (II-D). Experimental results are presented in Section III. Finally, concluding remarks and future work are given in Section IV.

II. METHOD: MUTUAL LOCALISATION USING LINES

The error in pose estimation increases over time due to sensor measurement errors in each relative pose estimation step. This drift in position of a robot can be reduced when it revisits a place. The assertion that a robot has returned to a previously visited location is known as ‘loop closure’. Fig. 1 shows a block diagram of how our 3D line alignment method is combined with our previous point feature based pose estimation routine presented in [2] for detecting loop closures.

The point feature based pose estimations for each image frame are used to construct a ‘pose graph’. The pose graph is...
made up of nodes representing each image frame and edges representing the relative position between nodes. In order to detect loop closures, line features are extracted from certain image frames (key frames) and these attached to the relevant nodes (key nodes) in the graph. Whenever a new key node is added to the graph, line features from this newly added key node are checked for loop closure. In order to limit the number of nodes to check against, a subset of existing key nodes are filtered based on the relative distance from the latest key node. Upon detecting a loop closure, the relative position between the identified nodes is calculated based on the transformation which aligns the two sets of lines in those nodes. The pose graph is then adjusted such that it is consistent with the new constraint (relative pose).

The same line alignment method can be used for computing the relative pose between two robots upon encounter. We assume that the robots are in the same coordinate frame, considering that their initial poses are known. Since the robots are close in proximity, it is very likely that one or more of the ‘latest’ nodes in their pose graphs contain line matches. Therefore, whenever two robots encounter each other, they communicate the most recent portions of their graphs. Each robot then checks the line features from the received graph against their own most recent nodes for matches. In a similar way to detecting loop closures, the relative position between the identified nodes is calculated and used for adjusting the pose graph.

The steps involved in these procedures are detailed in the sub-sections below.

A. Line Detection, Extraction and Matching

The relative pose estimation process starts by detecting line features and extracting descriptors from two RGB images. These images may be from two different cameras or the same camera with a time lapse. We used the Line Band Descriptor (LBD) based method presented in [7] for detecting, extracting and matching line features from images as it is currently the state-of-the-art robust line matching approach. Compared to other methods like the popular Mean–Standard deviation Line Descriptor (MSLD) [4], the LBD approach is faster to generate the matching results and is also robust against various image transformations including rotation, scale changes, illumination changes, occlusion, and moderate view point changes even for low-texture scenes. These advantages can be attributed mainly to the fact that the lines are detected in the scale space and combining the local appearance and geometric constraints together eliminates most of the mismatches.

B. 3D Lines and Alignment

The matched 2D line pairs identified by the LBD technique are first filtered based on length and then projected onto 3D coordinates by using the 2D positions of their end points and the corresponding data from the depth frame. The transformation which aligns these two sets of 3D lines onto one other is related to the relative poses between the two camera frames.

The problem of aligning 3D lines can be transformed into a 3D point alignment problem if two point pairs in common can be identified for each line pair. However, corresponding lines from two images may be different segments of the same physical line. Therefore, the line segments are first converted into infinite lines and then the common points between the line correspondences are identified as illustrated in Fig. 2.

\[
\Delta R = R^T, \quad \Delta t = -R^T t, \quad (1)
\]

where \(\Delta R\) and \(\Delta t\) are the relative rotation and translation of one camera body frame to the other, respectively. The proposed line alignment method is detailed in Algorithm 1.

C. Loop Closure and Pose Graph Optimisation

Line matching and 3D alignment steps both succeed only when loop closure is present. In other cases line matching may not produce any matches or RANSAC may reject false matches and fail to give an alignment. When a loop closure is detected a new constraint (edge) is added between the corresponding nodes in the graph. Then the pose graph is adjusted based on g2o graph optimisation technique [19].

The closest point \(X\) is computed for each set of 3D lines using the generalised Weiszfeld’s algorithm [18]. Due to the correspondences between the two sets of lines and as we assume the environment is static, these computed closest points coincide at the same physical point in space. Therefore, the closest points \(C_i\) (from the computed point \(X\)) on each corresponding line pair \(L_i\) \((i = 1, 2, \ldots)\) also coincide and are taken as matching point pairs. The other matching point pairs \(D_i\) are taken at a fixed distance on each line \(L_i\) from points \(C_i\). Now that the matching lines have been converted to matching points, the two point sets are subjected to least squares minimisation to calculate the best alignment. We use our method using sufficient statistics proposed in [2] for efficiently aligning these 3D point sets and filtering out falsely matched lines using Random Sample Consensus (RANSAC). The relative pose can be determined by computing the inverse of the geometric transformation of the alignment. That is,
Algorithm 1 Proposed 3D Line Alignment

Input: Two sets of corresponding 3D lines $U = \{u_i\}, V = \{v_i\}, i = 1, 2, \ldots n$
Output: Rotation $R$ and translation $T$ (pose $p^*$).

Begin:
1: Set $l$ to 0
2: Set $\text{Succ}$ to false
3: while $l < \text{max} \_ \text{iterations}$ do
4: Randomly select minimal lines set $(U_l, V_l)$ from $(U, V)$
5: Find $L$ closest-point $x_u, x_v$ from $(U_l, V_l)$
6: Find closest points $(C_u, C_v)$ on lines $(U_l, V_l)$ from point $x_u, x_v$
7: Find second points $(D_u, D_v)$ on lines $(U_l, V_l)$ at a fixed distance from $(C_u, C_v)$
8: Calculate relative pose $p_l$ that aligns points $((C_u, D_u), (C_v, D_v))$
9: Calculate corresponding alignment error $e_s$
10: if $e_s < \text{threshold}$ then
11: for each $u_i \in U_l$ and $v_i \in V_l$ do
12: Repeat steps 6, 7 on $u_i, v_i$ and add corresponding points to sets $(C_u, C_v)$ and $(D_u, D_v)$
13: end for
14: Calculate relative pose $p_l$ using a RANSAC based outlier rejection method on $(C_u, D_u), (C_v, D_v)$
15: if step 14 succeeds then
16: Update the best pose $p^*$ with one that has if error of $p_l$ is less than the current
17: Set $\text{Succ}$ to true
18: end if
19: end if
20: increment $l$ by one
21: end while
22: if $\text{Succ}$ is true then
23: alignment successful and return estimated pose $p^*$
24: end if

D. Inter-robot localisation

The proposed method can used to calculate the relative pose between two robots. This is because the employed line descriptors (LBD) are robust against changes in rotation, scale, illumination and view point. In the event two robots come close in proximity they transmit their recent key nodes (lines and pose) to each other. Each robot can then use the same method outlined in the previous sub-sections to estimate the relative pose between themselves (mutual localisation). This estimation in turn can be used to improve their own pose graphs. Even though multiple RGB-D cameras pointed at the same scene can cause interference, a shake ‘n’ sense approach [20] can be used to reduce this. Due to inherent vibrations caused during the flight of an MAV, we expect that the interference between multiple sensors would not arise in our case even without additional components as in [20]. We are yet to validate this through experiments.

III. EXPERIMENTS AND RESULTS

We added the above mentioned loop closing technique to our earlier point feature based pose estimation method implemented in C++ [2]. For the line extraction and matching, we used a modified implementation of LBD descriptor [21] which excludes the original dependencies except OpenCV.

The algorithm was tested on an Intel Core i5 CPU @ 3.20 GHz with 8 GB RAM using the fr3_long_office sequence in the TUM RGB-D benchmark dataset [22] as well as the self recorded dataset used in [2]. We recorded this dataset with a PrimeSenseTM Carmine 1.08, which can provide both colour and depth information with a 640x480 pixels resolution at 30 frames per second. The ground truth was measured using an OptiTrack motion capture system with eight Flex 13 motion capture cameras installed in an indoor environment with much texture and structure.

Fig. 3 shows two nodes from the fr3_long_office sequence where loop closure has been detected. The line matches shown on the images are the output matches from LBD. Even though there are some false matches, these lines are removed during our alignment process. Table I reports the resulting relative pose (position $(x, y, z)$ and rotation quaternion $(q_x, q_y, q_z, w)$) between these image frames (Est) compared against the ground truth (GT) from the dataset.

<table>
<thead>
<tr>
<th></th>
<th>x</th>
<th>y</th>
<th>z</th>
<th>$q_x$</th>
<th>$q_y$</th>
<th>$q_z$</th>
<th>w</th>
</tr>
</thead>
<tbody>
<tr>
<td>GT</td>
<td>-0.293</td>
<td>-0.193</td>
<td>0.960</td>
<td>0.065</td>
<td>-0.022</td>
<td>-0.110</td>
<td>0.992</td>
</tr>
<tr>
<td>Est</td>
<td>-0.260</td>
<td>-0.174</td>
<td>0.995</td>
<td>0.074</td>
<td>-0.036</td>
<td>-0.123</td>
<td>0.989</td>
</tr>
</tbody>
</table>

Fig. 4 shows two nodes for which loop closure has not been detected. In this case, almost all line matches output by LBD are false. But our alignment process succeeds in identifying these as incorrect matches and appropriately does not output a transformation result.

Fig. 5 compares the camera trajectory for the benchmark sequence fr3. It clearly shows how our 3D line alignment based loop closure detection has corrected the earlier point-feature based pose estimations and brought the trajectory closer to the ground truth. Figures 6-7 shows how the camera trajectory for our recorded dataset has been improved by the loop closing. Table II reports the absolute position root-mean-square error.
(RMSE) for the trajectories of both these test sequences, as reported by the tool provided with the benchmark dataset.

**TABLE II: RMSE of estimated trajectories with only point-feature based pose estimations and after adding line-based loop closure on benchmark dataset and our recorded data**

<table>
<thead>
<tr>
<th>Dataset</th>
<th>Only Pose Estimations</th>
<th>With Loop Closure</th>
</tr>
</thead>
<tbody>
<tr>
<td>f3_long_office</td>
<td>0.350</td>
<td>0.196</td>
</tr>
<tr>
<td>Our recording</td>
<td>0.150</td>
<td>0.124</td>
</tr>
</tbody>
</table>

Fig. 5: Trajectory of benchmark dataset f3

(a) View 1
(b) View 2
(c) Ground truth
Fig. 6: Trajectory of recorded data before loop closure

In order to verify mutual localisation of multiple MAVs, we simulated two inter-robot scenarios in our laboratory. First, we moved one RGB-D camera in two different paths that meet at a particular location to allow mutual localisation. Lines features from the most recent pose graph nodes from the meeting location are tested with our proposed method to find the relative pose. Figure 9a shows the estimated trajectory of one camera’s movement before mutual localisation. The corresponding RMSE is 0.134m. The corrected trajectory based on the mutual localisation is shown in Figure 9b and the updated RMSE is 0.109m. In the second experiment, two RGB-D cameras were moved in two different paths and when these met at a common location, mutual localisation process was executed. The result was then added to the pose graph of each camera and g2o graph optimisation technique was applied to get the optimised trajectories. Figure 8 shows the two camera views at the meeting location together with the line matches between these two views. The ground truth and estimated trajectories for the two cameras are shown as path 1 and path 2 in Figure 10. Table III shows that the RMSE is reduced in both paths due to mutual localisation.

**TABLE III: RMSE of estimated trajectories before and after mutual localisation (ML)**

<table>
<thead>
<tr>
<th>Path</th>
<th>Before ML</th>
<th>After ML</th>
</tr>
</thead>
<tbody>
<tr>
<td>Path 1</td>
<td>0.161</td>
<td>0.127</td>
</tr>
<tr>
<td>Path 2</td>
<td>0.148</td>
<td>0.135</td>
</tr>
</tbody>
</table>

Fig. 7: Trajectory of recorded data after loop closure

(a) View 1
(b) View 2

Fig. 8: Views from the two cameras when at the meeting location and the lines matched during mutual localisation

**IV. CONCLUSION**

In this paper we present a 3D line alignment technique that can be used for complementing point-based pose estimations in limited resource MAVs by performing loop closures as well as computing relative pose estimation between MAVs. We apply an extension of the LBD detector as well as our earlier published efficient RANSAC-based outlier detection method for 3D alignment. The validity and accuracy of the proposed
method is evaluated on a public real-world benchmark dataset as well as sequences recorded in our laboratory. The results show that loop closing using our novel line-based method succeeds in considerably reducing the trajectory error from our point-based pose estimations. Because the employed line descriptors (LBD) are robust against changes in rotation, scale, illumination and viewpoint, our proposed method was also used to calculate the relative pose between two robots. We have demonstrated through experiments how this mutual localisation is used to successfully reduce trajectory errors. Collaborative mapping where robots use this method for correcting their maps based on mutual localisation upon encounters will be presented in future work.

REFERENCES


