JVLDLoc: a Joint Optimization of Visual-LiDAR Constraints and Direction Priors for Localization in Driving Scenario

Longrui Dong and Gang Zeng

Key Lab. of Machine Perception (MoE), School of Intelligence Science and Technology, Beijing 100871, China
{lrdong, zeng}@pku.edu.cn

Abstract. The ability for a moving agent to localize itself in environment is the basic demand for emerging applications, such as autonomous driving, etc. Many existing methods based on multiple sensors still suffer from drift. We propose a scheme that fuses map prior and vanishing points from images, which can establish an energy term that is only constrained on rotation, called the direction projection error. Then we embed these direction priors into a visual-LiDAR SLAM system that integrates camera and LiDAR measurements in a tightly-coupled way at backend. Specifically, our method generates visual reprojection error and point to Implicit Moving Least Square (IMLS) surface of scan constraints, and solves them jointly along with direction projection error at global optimization. Experiments on KITTI, KITTI-360 and Oxford Radar Robotcar show that we achieve lower localization error or Absolute Pose Error (APE) than prior map, which validates our method is effective.

Keywords: Visual-LiDAR SLAM · Sensor Fusion · Structure from Motion · Vanishing Point

1 Introduction

In the fields of computer vision and robotics, Simultaneous localization and mapping (SLAM) is an active research topic, it also plays a vital role in many real world applications. For example, by knowing their precise location, vehicles are able to navigate safely. To fully perceive surrounding scenes, agents are usually equipped with several sensors: camera, LiDAR, GPS/IMU, etc. As we all know, relatively cheaper cameras have been widely set on various kinds of platforms, which makes visual odometry (VO)/visual SLAM (V-SLAM) the main force in research[4,15]. Due to the characteristics of camera and algorithms’ reliance on low-level visual features, V-SLAM tends to fail in challenge conditions: abrupt motion, textureless region, too much occlusion. Light Detection And Ranging sensor (LiDAR), which can cover 360° FoV information, has an advantage over camera that the noise associated with each distance measurement is independent
Fig. 1: JVLDLoc: a tight-coupling of visual reprojection error, point to Implicit Moving Least Square(IMLS) surface error and direction projection error at global joint optimization.

of the distance and the lighting conditions[6]. But LiDAR does not offer texture and is often sparse due to the physical spacing between laser fibers.

In order to take advantage of those two types of sensors, fusion of visual and LiDAR data during localization is quite reasonable, which is called visual-LiDAR odometry/SLAM. Some recent work[8,10,19] have explored the way of fusing those two sensors. However, they nearly all fuse visual-LiDAR at a level of loosely coupling. [17,25] use pose graph to fuse different sensors. Although they are very efficient, rich geometric constraints from data are thrown away in optimization.

Unlike [25] which just takes odometry as relative pose factor between adjacent keyframes, we use input trajectory to construct direction priors. They contain scene structure information from images’ vanishing points. Since the direction is independent of distance, for the vehicle, as long as there is no road turn, the direction is always in view. As a result, we can establish longer range correspondences among much more keyframes than traditional visual point features[16]. Ideally, the motion of a vehicle on the road plane can be approximated as a two-dimensional movement, in which the rotating part is almost only a twist around an axis perpendicular to the ground. A more accurate rotation estimate can reduce the overall drift of the trajectory[20]. Motivated by them, we extract at most one vanishing point each image, whose corresponding direction is forward. Actually this direction provides only two degrees of freedom constraints to the vehicle’s orientation[1]. The main contributions are as follows:

- We come up with a scheme that fuses map prior and image-detected vanishing points, which can establish an energy term that only constrains rotation. That constraint, called the direction projection error, finally helps to mitigate the drift.

- We propose JVLDLoc for localization in driving scenario. Based on graph optimization, it integrates visual reprojection error, point-to-IMLS surface error and direction projection error into one energy function, and directly uses geometric constraints to optimize pose.
Experiments on KITTI, KITTI-360 and Oxford Radar Robotcar show that we achieve lower localization error or APE than prior map, which validates the effectiveness of our method.

2 Related Work

2.1 Visual-LiDAR SLAM

First we briefly review some of the major visual or LiDAR SLAM in literature. ORB-SLAM[4,16] completely uses ORB features, realizing a full V-SLAM system including loop detection, relocalization, and map fusion. IMLS-SLAM[6] estimates pose by minimizing the distance from the current point to the surface represented by the implicit moving least square method (IMLS). Then we categorize visual-LiDAR SLAM into two major genres according to whether or not to use both multi-modal residuals in the optimization stage.

Loosely Coupled Method Methods of this type usually extract depth from LiDAR to enhance visual odometry. DEMO[24] integrates the depth information of point clouds into visual SLAM for the first time. LIMO-PL[10] utilizes line features with depths from LiDAR. [23] uses 3D lines from prior LiDAR map to associate with 2D line features from images. DVL-SLAM[19] requires no feature points when using sparse depth from point cloud. However, incorrect pixel-depth pair might damage accuracy.

Tightly Coupled Method ViLiVO[22] adds the image feature reprojection error and the point cloud matching error to Bundle adjustment (BA). Similar methods appear in [26] but they further introduce observation error between image feature and point cloud. TVL-SLAM[5] also incorporates visual and lidar measurements in backend optimization. There are also filter based methods for tightly fusion different data[14]. LIO-SAM[17] and LVI-SAM[18] achieve sensor-fusion atop the pose graph instead of including any geometric constraints.

2.2 Using Vanishing Points as Direction Constraint

The common directions of parallel 3-D lines are known as vanishing points (VPs). Because the VPs offer direction information independent of the present robot proposition[3], Many academics have researched VPs to help improving orientation. [15] apply the orthogonal constraint between multiple VPs, known as the Manhattan world constraint, to calculate the full rotation. [13] no longer need Manhattan world assumption but they will extract all valid VPs from an image while our method indeed only use one VP. [21] also combine VP error in a pose estimator, but it have to generate VPs of three directions before refinement.

3 Notation

\( \mathcal{I}_n, \mathcal{S}_n \) stands for an image and a point cloud at time \( t_n \). A frame \( \mathcal{F}_n \) is the data structure that includes ORB features of \( \mathcal{I}_n \) and corresponding processed scan \( \mathcal{S}_n \), while \( \mathcal{K}_k \) refers to a keyframe. Every 3D point \( \mathbf{X} = (X, Y, Z, 1)^T \in \mathbb{R}^4 \) in a
Fig. 2: The Overall diagram of the proposed JVLDLoc. Our approach inputs image and point cloud stream and estimates 6DoF poses.

Homogeneous coordinate from visual map, can be observed in the image plane as a pixel coordinate \( u = (u, v, 1)^T \in \mathbb{R}^3 \) via a projection function:

\[
\pi(X) = \frac{1}{Z} \mathbf{K} [1|0] X = \frac{1}{Z} \begin{bmatrix}
    f_x & 0 & c_x & 0 \\
    0 & f_y & c_y & 0 \\
    0 & 0 & 1 & 0
\end{bmatrix}
\begin{bmatrix}
    X \\
    Y \\
    Z \\
    1
\end{bmatrix}
= \begin{bmatrix}
    f_x \frac{X}{Z} + c_x \\
    f_y \frac{Y}{Z} + c_y \\
    1
\end{bmatrix}
= \begin{bmatrix}
    u \\
    v \\
    1
\end{bmatrix}
\]

(1)

where \( \mathbf{K} \in \mathbb{R}^{3 \times 3} \) is camera intrinsic matrix, \( f_x, f_y, c_x \) and \( c_y \) are intrinsic parameters. We use \( T_{wn} \in SE(3) \) to express the transformation from the \( n_{th} \) frame’s camera coordinate to the world coordinate, \( T_{nw} \) is the inverse of \( T_{wn} \). \( R_{wn} \in SO(3) \) and \( t_{wn} \in \mathbb{R}^3 \) are rotation and translation part of \( T_{wn} \).

4 Method

The dataflow of the JVLDLoc is in Fig. 2. The whole system consists of four modules: tracking (Section 4.1), local mapping (Section 4.2), visual based loop closing (see supplemental material for detail) and global mapping (Section 4.3).

4.1 Tracking

We process visual and LiDAR data and compute their features and decide whether to generate a new keyframe. Note that we need to transform point cloud to camera coordinate by given extrinsic: \( T_{CL} \). If there is no initial pose from other odometry, we can still handle it by launching a LiDAR odometry. It is modified from the approach in IMLS-SLAM [6]. The main changes compared to [6] can be found in supplemental material.

Visual Feature Association Unlike [16], we no longer launch motion-only BA. Instead, we just use odometry poses to construct 2D feature-3D point associations within the visual map. Finally, we incorporate processed image keypoints and point cloud into a frame \( F_n \). Tracking module will treat current \( F_n \) as a new keyframe \( K_n \) if 4 conditions in [16] are satisfied OR \( F_n \) has valid supporting direction constraints (described in Section 4.3).
4.2 Local Mapping

A new keyframe $K_n$ will trigger a visual-LiDAR graph optimization. The corresponding joint energy function consists of visual reprojection error and point to IMLS surface error.

Our backend maintains a covisibility graph where keyframes that observe same 3D point are connected. Denote keyframe launching local mapping as $K_n$, sort all the keyframes connected to $K_n$ at covisibility graph (abbreviated as $K_{cov}^n$) including itself by their timestamps: $K_{local} = \{...K_i, K_n\}$. $P_{local}$ refers to all 3D visual map points seen by $K_{local}$. Defining $X_i = \{(x_j, X_j)|j = 0, 1, ...\}$ as the set of correspondences between feature points at keyframe $K_i$ and 3D visual map points from $P_{local}$. Then do the following steps:

1) For all keyframes in $K_{local}$, according to visual correspondences from $X_k$, we can get visual reprojection error term like [16]: $e_V(i, j) = \|x_j^i - \pi_i(R_{iw}X_j^i + t_{iw})\|_2^2$.

2) For current keyframe $K_n$ which has a global frame index $f_n$, we find the first keyframe whose frame index met $f_n - f_i \leq N$, where $N$ is the size of local scan map of each keyframe. Denote this keyframe as $K_{start}$. Now list all corresponding scan from $K_{start}$ to $K_{n-1} : \{S_{start}, ..., S_{n-1}\}$. We use each keyframe’s current pose to transform these scans to world frame and accumulate to be a whole point cloud model named $S_{model}$ (See Fig. 3 (a)).
3) Sort every point in $S_n$ by their computed nine feature values. Then transform $S_n$ to world coordinate to find correspondences in its $S_{model}$. We use the same scan sampling strategy as Section V in [6] to get sampled point cloud $\tilde{S}_n$. we compute for each point the distance to the IMLS surface in $S_{model}$:

$$I^n(p_x) = \frac{\sum_{p_i \in S_{model}} W_i(p_x)((p_x - p_i) \cdot \hat{n}_c)}{\sum_{p_i \in S_{model}} W_i(p_x)}$$

(2)

where $p_x \in \tilde{S}_n$ but in world coordinate, $p_i$ is the neighbor of $p_x$ from model scan and $\hat{n}_c$ is the normal of $p_i$ that is nearest neighbor of $p_x$. And $p_c$ is from $S_c$ of $K_c$. $W_i(p_x) = e^{-\|p_x - p_i\|^2/h^2}$ is the same weight function as [6]. Next we can obtain the projection of $p_x$ at IMLS surface, $p_y$: $p_y = p_x - I^n(p_x)\hat{n}_c$.

4) Now we have found a association between $K_c$ and $S_c$. We express global direction prior $\hat{D}_w \in \mathbb{R}^4$ in a homogeneous coordinate as follows [1]: $\hat{D}_w = [\hat{d}_w, 0]^T$.

4.3 Global Mapping using Direction Priors

After local mapping of the final keyframe, we will launch the global mapping.

Association of Direction Priors and VPs We detect the vanishing point $v$ in each image and get corresponding direction via $d = R_{wc}K^{-1}v/\|R_{wc}K^{-1}v\|$[1], where $d \in \mathbb{R}^3$. Detailed steps and bi-level mean-shift clustering on directions are in supplemental material. After clustering, there is at most one VP per image. Each VP corresponds to a cluster’s center which is direction prior $d_w$.

Global Joint Optimization One can regard global optimization as a special case of local mapping: ALL keyframes in the map are included in $K_{global}$. We traverse all keyframes in $K_{global}$ to construct Eq. (3). The global optimization graph is shown in Fig. 3 (b). We express global direction prior $\hat{D}_w \in \mathbb{R}^4$ in a homogeneous coordinate as follows [1]: $\hat{D}_w = [d_w, 0]^T$. 
Fig. 4: Construction of direction projection error. The left figure shows the process of vanishing point detection. A ray parallel to the direction prior passing through the camera optical center intersects the image plane at $\tilde{v}$ as Eq. (5).

The direction prior $\tilde{D}_n$ at world coordinate is transformed by current estimated rotation to camera’s coordinate: $\tilde{D}_n$. This direction can be written as a 3D line through a homogeneous point $A$:

$$Z(\lambda) = A + \lambda \tilde{D}_n,$$

where $\lambda \in (0, \infty)$.

The observation on image of that direction is then equal to the projection of a point at infinity on that 3D line as follows:

$$\tilde{v} = \lim_{\lambda \to \infty} \pi(Z(\lambda)) = \lim_{\lambda \to \infty} \pi((A + \lambda \tilde{D}_n)) = \pi(\tilde{D}_n) \quad (5)$$

The direction projection error is just distance between $\tilde{v}$ and actual detected VP $v$ (See Fig. 4):

$$e_D(n) = \|v - \tilde{v}\|^2 = \|v - \pi(\tilde{D}_n)\|^2 = \|v - \pi([R_n w |I] \tilde{D}_w)\|^2 \quad (6)$$

Finally, we take out all keyframes $K_{global}$ from the global map, the global joint energy function can be written as:

$$E_{global} = \sum_{n \in K_{global}} \{\alpha \sum_{j \in X_n} \rho(e_V(n, j)) + \beta \sum_{j \in S_n} e_L(n, j) + \gamma e_D(n)\} \quad (7)$$

where $\alpha, \beta, \gamma$ are weights factors to balance these three types of residuals. In practice, we set $\alpha$ to be 1 and set the other two according to the corresponding number of residuals. The poses are optimized by:

$$\{R_n, t_n | n \in K_{global}\} = \arg \min_{R_n, t_n} E_{global} \quad (8)$$

Limited to the length of the paper, please refer to our supplemental material for the jacobian of direction projection error.

5 Experiments

This method is implemented in C++ and Python. All the non-linear optimization mentioned above are Levenberg-Marquardt method in $g^2o$ library [11]. Our experiments involve the following 3 driving datasets which contain camera and LiDAR sensors:
Fig. 5: Mapping result on KITTI 02. (a) is the full view output from JVLDLoc. (b), (c), (d) are the enlargement of the red box region in (a) output from different methods respectively. LiDAR odo means LiDAR Odometry with \( N = 1 \).

Table 1: RMSE(m) of APE for KITTI 00-10. For each sequence, we report RMSE of input pose, JVLDLoc without direction priors and full JVLDLoc respectively.

| KITTI Odometry | 00 | 01 | 02 | 03 | 04 | 05 | 06 | 07 | 08 | 09 | 10 | AVG |
|----------------|----|----|----|----|----|----|----|----|----|----|----|-----|
| ORB-SLAM2 [16]| 1.354 | 10.16 | 6.899 | 0.693 | 0.178 | 0.783 | 0.714 | 0.536 | 0.542 | 1.643 | 1.156 | 2.515 |
| w.o. direction priors | 1.367 | 3.289 | 4.669 | 0.614 | 0.169 | 0.705 | 0.485 | 0.473 | 3.354 | 2.433 | 0.990 | 1.686 |
| w. direction priors | 1.205 | 1.999 | 3.544 | 0.530 | 0.151 | 0.593 | 0.300 | 0.441 | 2.862 | 1.498 | 0.818 | 1.267 |
| LiDAR odometry \( N=1 \) | 6.849 | 35.03 | 12.74 | 0.839 | 0.265 | 3.379 | 2.109 | 0.759 | 8.123 | 3.917 | 1.084 | 6.828 |
| w.o. direction priors | 2.027 | 7.932 | 4.414 | 0.764 | 0.186 | 2.286 | 0.374 | 0.945 | 4.602 | 3.347 | 1.065 | 2.268 |
| w. direction priors | 1.616 | 3.097 | 3.015 | 0.585 | 0.175 | 0.802 | 0.302 | 0.521 | 3.131 | 1.828 | 0.933 | 1.455 |

KITTI Odometry Dataset [7] may be the most popular dataset for SLAM field, which covers urban city, rural road, highways, roads with vegetation, etc. We compare the algorithm with other state-of-the-art method on 00-10 because they are widely used in other papers.

KITTI-360 Dataset [12] is still under development, the ground truth does not fully cover each frame of raw data, because they discard some frames collected at very low moving speed. For the convenience of evaluation, we clip two sequences from raw data as test data.

Oxford Radar RobotCar Dataset [2] is another fresh dataset. The car is driven to travel 32 times in same scene. We select a sequence recorded on datetime 2019-01-10-14-36-48.

5.1 Improvements over Prior Map

To prove that JVLDLoc will reduce global drift, we visualize the output point cloud map with color in Fig. 5. Qualitively, our method greatly improves the global consistency of the map. For more mapping visualization, please turn to supplemental material. We use RMSE of APE[9] to quantify the drift.

Table 1 includes results test on KITTI odometry when fed with different prior map. We have set four baselines as input odometry including ORB-SLAM2[16], our LiDAR odometry under different \( N \) which is the number of scans kept as local map. The other two groups of result is shown in supplemental material. Under different quality prior maps, the average accuracy of these 11 sequences can be significantly improved. Especially when input LiDAR odometry with \( N = 1 \), the mean APE is reduced by about 79%. Table 2 is for the other two datasets, which
Fig. 6: A visualization of direction priors of KITTI 05 when downsampled under different intervals $V_{interval}$: 2, 15, 300 (from left to right). The number of total direction projection error is reduced from 117 to 12.

Table 2: APE for KITTI-360 and Oxford Radar RobotCar. $MIN$ and $MAX$ are the minimum and maximum APE over the entire trajectory.

| KITTI-360 | 12 | 20 |
|-----------|----|----|
| ORB-SLAM2 [16] | 0.538 | 1.904 | 3.289 |
| w.o. direction priors | 0.196 | 1.367 | 3.934 |
| w. direction priors | 0.057 | 0.891 | 2.828 |

| Oxford | 1 |
|--------|---|
| Lidar odometry N=10 | 2.592 | 26.25 | 53.43 |
| w.o. direction priors | 2.158 | 11.09 | 19.01 |
| w. direction priors | 0.437 | 7.982 | 27.08 |

Also prove our method’s effect. That is because JVLDLoc directly uses different kinds of geometric constraints to optimize pose.

5.2 Effects of Direction Priors

To demonstrate the effects of direction priors itself, we further compare accuracy with and without using the proposed direction priors. APE are still shown in Table 1, Table 2. It can be seen the direction priors are able to further improve localization due to smaller average RMSE on KITTI. For other two datasets, with help of direction priors, maximum of APE strongly decrease on KITTI-360 from the level when there is none direction constraint; The minimum and RMSE have again significantly reduced by 80% and 28% on Oxford Radar RobotCar.

In a word, although the direction priors are not very accurate as they are based on input poses, they still contribute to better localization. We believe the reason is that direction priors impose longer term geometric constraints as the direction is an infinite landmark. Fig. 7 presents intuitive trajectory comparisons. Apparently, we can get less drift after long distance under direction priors.

5.3 Comparison to Other Methods on KITTI Odometry Dataset

Besides APE, Table 3 also calculates relative transition error and rotation error for 100, 200,... and 800 meter distances with KITTI’s official evaluation code.

Compared to other visual-LiDAR methods, JVLDLoc achieve best relative translation error and APE, which indicates the superiority of our method over others using the same modality of data. LIMO-PL[10] and DVL-SLAM[19] are loosely-coupled methods, which cause quite large translation error; TVL-SLAM[5] is a SOTA visual-LiDAR SLAM on KITTI, which also incorporates visual and lidar measurements in backend optimization. TVL-SLAM VL calib refers to their online camera-lidar extrinsics calibration version. Our JVLDLoc can outperform
Fig. 7: Trajectory of JVLDLoc(red), JVLDLoc without direction priors(green), LiDAR odometry(blue) on Oxford. The right image is the enlargement of the dotted box on the left image.

Table 3: Comparison to other methods on KITTI 00-10. $t_{rel}$ and $r_{rel}$ are relative translational error (%) and rotational error ($^\circ/100$m). Due to page limitations, results on 05-10 are in supplemental material. (NA : Not Available).

| Method               | 00 | 01 | 02 | 03 | 04 | AVG on 00-10 |
|----------------------|----|----|----|----|----|--------------|
|                      | $t_{rel}$ | $r_{rel}$ | $t_{rel}$ | $r_{rel}$ | $t_{rel}$ | $r_{rel}$ | $t_{rel}$ | $r_{rel}$ | $t_{rel}$ | $r_{rel}$ | $t_{rel}$ | $r_{rel}$ | $t_{rel}$ | $r_{rel}$ | $t_{rel}$ | $r_{rel}$ | $t_{rel}$ | $r_{rel}$ | $t_{rel}$ | $r_{rel}$ |
| IMLS-SLAM [6]        | 0.60 0.18 3.90 | 0.82 0.10 2.41 | 0.53 0.14 7.16 | 0.68 0.22 6.65 | 0.33 0.12 1.84 | 0.52 0.14 1.96 |
| LIMO-PL [10]         | 0.99 NA NA | 0.87 NA NA | 1.38 NA NA | 0.65 NA NA | 0.42 NA NA | 0.94 NA NA |
| DV-L-SLAM [19]       | 0.93 NA 1.47 NA | 1.11 NA 0.92 NA | 0.67 NA 0.98 NA | 0.98 NA NA | 0.98 NA NA |
| TVL-SLAM-No calib [5] | 0.69 NA 0.84 6.50 | 0.74 NA 2.16 0.71 NA 0.75 | 0.49 NA 0.18 0.62 | 0.62 NA 1.50 |
| TVL-SLAM-VL calib [5] | 0.57 NA 0.88 4.40 | 0.67 NA 1.87 0.71 NA 0.74 | 0.45 NA 0.22 0.57 NA 1.19 |
| LIO-SAM [17]         | 3.33 3.00 956 | 2.94 0.60 19.8 | 2.68 87.6 212 | NA NA NA | 0.92 0.47 0.36 | 1.08 0.40 4.46 |
| JVLDloc input IMLS   | 0.63 0.15 1.31 | 0.58 0.06 1.77 | 0.65 0.11 2.62 | 0.30 0.14 0.50 | 0.29 0.10 0.15 | 0.54 0.12 1.13 |

TVL-SLAM VL calib though using raw extrinsics. LIO-SAM[17] is representative method of Lidar-inertial SLAM. We run its open source code with given parameters $^1$. Due to the unknown intrinsics of the IMU, LIO-SAM failed on 00, 02, 08, but we can get better accuracy on three metrics for all left sequences.

5.4 Ablation Study

Different Number of Direction Priors Here we attempt to relax the strength of direction priors by reducing the number of keyframes involved in direction priors. For each direction cluster, downsample at a distance $V_{interval}$ from the distribution of existing VPs. Fig. 6 is a visualization of different downsamprled direction priors.Fig. 8 shows the comparison of APE under different $V_{interval}$.

From those figures, we can concluded that the APE do not suffer drastic expansion due to less direction constraints. Even though there is only about 10 direction constraints retained, we still get quite better accuracy than no direction

$^1$ https://github.com/TixiaoShan/LIO-SAM#other-notes
Fig. 8: left: Comparison of APE on KITTI 01. The x-coordinate is frame index. right: Box plots of APE on KITTI 07. full refers to no downsampling, WOdirection refers to no direction priors. Other lines respectively correspond to different $V_{interval}$. Take ds2.txt as an example, it is the APE when $V_{interval} = 2$.

priors at all. Because the left direction priors can still play some roles. So we argue that as long as there is a few direction constraints over a long distance, the drift will be reduced at a certain level. For each KITTI sequence’s relative rotation error and APE RMSE under different $V_{interval}$, see supplementary material.

We also explore the influence of different scan map size $N$ of each keyframe. The plot and analysis of that result are in supplemental material (page limit).

6 Conclusion

This paper present a strategy that combines map prior and VPs from images to create direction projection error, which is an energy term solely bound on rotation. Then, we implement these direction priors into a visual-LiDAR localization system that tightly couples camera and LiDAR measurements. Experiments on three driving datasets prove that our method achieve lower localization drift. Even though our method is effective and outperforms many related work, There are still two flaws. One thing is time efficiency. Our current implementation can not run in real-time. It is a trade-off between performance and time cost. The second thing is that our method relies on an assumption that agent locally moves on an approximate plane. So compared to UAV or handheld devices, the driving scenes are more likely to meet the above assumptions.

Acknowledgements. This work is supported by the National Key Research and Development Program of China (2017YFB1002601), National Natural Science Foundation of China (61632003, 61375022, 61403005), Beijing Advanced Innovation Center for Intelligent Robots and Systems (2018IRS11), and PEK-SenseTime Joint Laboratory of Machine Vision.
References

1. Andrew, A.M.: Multiple view geometry in computer vision. Kybernetes (2001)
2. Barnes, D., Gadd, M., Murcutt, P., Newman, P., Posner, I.: The oxford radar robotcar dataset: A radar extension to the oxford robotcar dataset. In: Proc IEEE Int. Conf. Rob. Autom. (ICRA) (2020). https://doi.org/10.1109/ICRA40945.2020.9196884
3. Bosse, M., Rikoski, R., Leonard, J., Teller, S.: Vanishing points and three-dimensional lines from omni-directional video. The Visual Computer 19(6), 417–430 (2003)
4. Campos, C., Elvira, R., Rodríguez, J.J.G., Montiel, J.M., Tardós, J.D.: Orb-slam3: An accurate open-source library for visual, visual-inertial, and multimap slam. IEEE Trans. Robot. (2021)
5. Chou, C.C., Chou, C.F.: Efficient and accurate tightly-coupled visual-lidar slam. IEEE Trans. Intell. Transp. Syst. (2021)
6. Deschaud, J.E.: Imls-slam: scan-to-model matching based on 3d data. In: Proc IEEE Int. Conf. Rob. Autom. (ICRA). pp. 2480–2485. IEEE (2018)
7. Geiger, A., Lenz, P., Stiller, C., Urtasun, R.: Vision meets robotics: The kitti dataset. The International Journal of Robotics Research 32(11), 1231–1237 (2013)
8. Gräter, J., Wilczynski, A., Lauer, M.: Limo: Lidar-monocular visual odometry. In: Proc IEEE Int. Conf. Intell. Rob. Syst. (IROS). pp. 7872–7879 (2018)
9. Grupp, M.: evo: Python package for the evaluation of odometry and slam. https://github.com/MichaelGrupp/evo (2017)
10. Huang, S.S., Ma, Z., Mu, T.J., Fu, H., Hu, S.: Lidar-monocular visual odometry using point and line features. In: Proc IEEE Int. Conf. Rob. Autom. (ICRA). pp. 1091–1097 (2020)
11. Kümmerle, R., Grisetti, G., Strasdat, H., Konolige, K., Burgard, W.: g 2 o: A general framework for graph optimization. In: Proc IEEE Int. Conf. Rob. Autom. (ICRA). pp. 3607–3613. IEEE (2011)
12. Liao, Y., Xie, J., Geiger, A.: KITTI-360: A novel dataset and benchmarks for urban scene understanding in 2d and 3d. arXiv.org 2109.13410 (2021)
13. Lim, H., Jeon, J., Myung, H.: Uv-slam: Unconstrained line-based slam using vanishing points for structural mapping. IEEE Robot. Autom. Lett. (2022)
14. Lin, J., Zheng, C., Xu, W., Zhang, F.: R2live: A robust, real-time, lidar-inertial-visual tightly-coupled state estimator and mapping. arXiv preprint arXiv:2102.12400 (2021)
15. Liu, J., Meng, Z.: Visual slam with drift-free rotation estimation in manhattan world. IEEE Robot. Autom. Lett. 5(4), 6512–6519 (2020)
16. Mur-Artal, R., Tardós, J.D.: Orb-slam2: An open-source slam system for monocular, stereo, and rgb-d cameras. IEEE Trans. Robot. 33(5), 1255–1262 (2017)
17. Shan, T., Engel, B., Meyers, D., Wang, W., Ratti, C., Rus, D.: Lio-sam: Tightly-coupled lidar inertial odometry via smoothing and mapping. In: Proc IEEE Int. Conf. Intell. Rob. Syst. (IROS). pp. 5135–5142. IEEE (2020)
18. Shan, T., Engel, B., Ratti, C., Rus, D.: Lvi-sam: Tightly-coupled lidar-visual-inertial odometry via smoothing and mapping. In: Proc IEEE Int. Conf. Rob. Autom. (ICRA). pp. 5692–5698. IEEE (2021)
19. Shin, Y.S., Park, Y.S., Kim, A.: Dvl-slam: sparse depth enhanced direct visual-lidar slam. Auton. Robots 44, 115–130 (2020)
20. Sturm, J., Engelhard, N., Endres, F., Burgard, W., Cremers, D.: A benchmark for the evaluation of rgb-d slam systems. In: Proc IEEE Int. Conf. Intell. Rob. Syst. (IROS). pp. 573–580. IEEE (2012)
21. Wang, P., Fang, Z., Zhao, S., Chen, Y., Zhou, M., An, S.: Vanishing point aided lidar-visual-inertial estimator. In: Proc IEEE Int. Conf. Rob. Autom. (ICRA). pp. 13120–13126. IEEE (2021)
22. Xiang, Z.Z., Yu, J., Li, J., Su, J.: Vilivo: Virtual lidar-visual odometry for an autonomous vehicle with a multi-camera system. In: Proc IEEE Int. Conf. Intell. Rob. Syst. (IROS). pp. 2486–2492 (2019)
23. Yu, H., Zhen, W., Yang, W., Zhang, J., Scherer, S.: Monocular camera localization in prior lidar maps with 2d-3d line correspondences. In: Proc IEEE Int. Conf. Intell. Rob. Syst. (IROS). pp. 4588–4594. IEEE (2020)
24. Zhang, J., Kaess, M., Singh, S.: A real-time method for depth enhanced visual odometry. Auton. Robots 41(1), 31–43 (2017)
25. Zhao, S., Zhang, H., Wang, P., Nogueira, L., Scherer, S.: Super odometry: Imu-centric lidar-visual-inertial estimator for challenging environments. In: Proc IEEE Int. Conf. Intell. Rob. Syst. (IROS). pp. 8729–8736. IEEE (2021)
26. Zhen, W., Hu, Y., Yu, H., Scherer, S.: Lidar-enhanced structure-from-motion. In: Proc IEEE Int. Conf. Rob. Autom. (ICRA). pp. 6773–6779 (2020)
Supplemental Material to
JVLDLoc: a Joint Optimization of
Visual-LiDAR Constraints and Direction Priors
for Localization in Driving Scenario

Longrui Dong and Gang Zeng

Key Lab. of Machine Perception (MoE),
School of Intelligence Science and Technology,
Beijing 100871, China
{lrdong,zeng}@pku.edu.cn

1 Supplement to Method
1.1 LiDAR Odometry in Tracking
Our JVLDLoc contains a LiDAR odometry in tracking in case an input without
prior poses is encountered. It is modified from the approach in IMLS-SLAM[4],
which runs scan-to-model matching with Implicit Moving Least Square(IMLS)
surface representation. For each input point cloud frame, \( N \) last scans are kept
as local map. The technique details can be found in [4] except the following main
differences:

– For each input scan, we discard points of which distance to local sensor’s
origin is less than 5m.
– Before small object removal, We apply RadiusOutlierRemoval to pointcloud
and extract ground plane by SACSegmentation[9] instead of voxel growing
from original paper.

After the above LiDAR odometry, we send initial poses to do visual feature
association and direction clustering.

1.2 Scan to Map Association in Local Mapping
In Section 4.2 of our submitted paper, we only construct IMLS constraint be-
tween current keyframe \( K_n \) and its point cloud map at Step 4). To build enough
dense 3D data associations in current \( K_{local} \), we take out all previous saved cor-
respondences (see gray dotted line in Fig.3 (a) of submitted paper) that \( p^j_x \),
\( p^j_y \) and \( n^j_c \) are within \( K_{local} \):

\[
e_{L_{old}}(m,j) = \|(T_{wc} T_{wm} p^j_x - p^j_y) \cdot n^j_c \|^2
\]  \hspace{1cm} (1)

where \((p^j_x, p^j_y, n^j_c) \in L\). Apparently above equation is nearly the same as
Eq.(3) in main paper except that \( m < n \). In this way, we can give the poses of
the keyframes in \( K_{local} \) more constraints from the point cloud data.
Note that the pipeline constructing point to IMLS surface residual (including Eq.(2), Eq.(3) in main paper) are all borrowed from [4] and modified for integration into our local mapping module.

1.3 Loop Closing
To weaken cumulative localization error, we also detect loop candidate for each keyframe[8]. Specific loop candidates detection procedures are the same as [8], which is based on visual bags of words place recognition. Once we estimates the correction transformation $\hat{T}_{nw}$ of current accepted keyframe $K_n$, this correction is distributed over all keyframes from $K_l$ to $K_n$. The pseudocode of distributing $SE(3)$ error, inspired by [13], is in Algorithm 1. To simplify the description, we use $T_i$ to represent $\hat{T}_{iw}$ in the following algorithm flow, where $i \in [i, n)$.

Algorithm 1 Distribute $SE(3)$ error over keyframes

| Input: $T = \{T_l, T_{l+1}, ..., T_n\}$, $\hat{T}_n$ |
| Output: $T_{corrected} = \{\hat{T}_l, \hat{T}_{l+1}, ..., \hat{T}_n\}$ $\triangleright$ ($\hat{\cdot}$) means corrected pose, $T_i$ refers to pose of keyframe $K_i$ |
| 1: $\Delta T_n \leftarrow \hat{T}_n T_n^{-1}$ $\triangleright$ the correction transformation for $K_n$ |
| 2: $\Delta R_n, \Delta t_n \leftarrow \text{Rot}\_\text{part}(\Delta T_n), \text{Tran}\_\text{part}(\Delta T_n)$ |
| 3: $\Delta \phi_n \leftarrow \log(\Delta R_n)$ $\triangleright$ Logarithmic map of $SO(3)$ Lie group |
| 4: $i \leftarrow (l + 1)$ |
| 5: while $i < n$ do |
| 6: $\Delta t_n^i \leftarrow \frac{i-1}{i-1} \Delta t_n$ |
| 7: $\Delta \phi_n^i \leftarrow \frac{i-1}{i-1} \Delta \phi_n$ |
| 8: $\Delta R_n^i \leftarrow \text{Exp}(\phi_n^i)$ $\triangleright$ Exponential map of $so(3)$ Lie algebra |
| 9: $\Delta T_n^i \leftarrow (\Delta R_n^i, \Delta t_n^i)$ |
| 10: $\hat{T}_i \leftarrow \Delta T_n^i T_i$ |
| 11: end while |

1.4 Association of Direction Priors and VPs
The direction of a line, not its position, determines the vanishing point[1]. So only the rotation part of provided prior poses are utilized to generate global direction at world coordinate. Detailed steps are as follows:

1) Detect at most 3 vanishing points from each (left) image.
2) Denoting obtained image plane coordinates of vanishing points as $v$. Then we use prior orientation $R_{wc}$ from other odometry to obtain corresponding direction at world coordinates: $d = R_{wc} K^{-1} v / \| R_{wc} K^{-1} v \|$, where unit direction $d \in \mathbb{R}^3$.
3) Run bi-level mean-shift clustering (see Algorithm 2) to all 3D vanishing directions from last step.

It is worth mentioning that these direction priors are mainly distributed in the approximate straight path of the trajectory.
Algorithm 2: Bi-level Mean-shift clustering for 3D vanishing directions

Input: All raw frame index-direction pairs: PD = \{(f_i, d_i)\}, 2D translations from prior poses \(t = \{t_i\}\), where \(i \in |PD|\)

Output: Supporting directions \(\hat{d}_w\) represented by Cluster centers and corresponding frame index

1: \(i \leftarrow 0\)
2: \(\text{while } i < |PD| \text{ do} \)
3: \(fd_i \leftarrow \Delta t_i \triangleright \text{Ground plane projection of the forward direction, } d_i^{proj} \text{ is the same}\)
4: \(\text{if } \arccos(d_i^{proj} \cdot fd_i / \|d_i^{proj} \cdot fd_i\|) > Th_f \text{ then}\)
5: \(\text{discard }(f_i, d_i)\)
6: \(i \leftarrow i + 1\)
7: \(\text{end while} \triangleleft \text{ new set: } PD_1\)
8: \(d \leftarrow \{d_i | d_i \in PD_1\}\)

9: \((^{m}d, \text{cluster_id}) \leftarrow \text{Mean_shift}(d, \text{bandwith1}) \triangleright \text{Raw cluster center and cluster IDs for each } d_i \in d\)

10: \(\text{while } i < |^{m}d| \text{ do} \)
11: \(\delta_i \leftarrow \text{standard deviation of each cluster}\)
12: \(s_i \leftarrow \text{slope of each cluster center’s projection } ^{m}d_i^{proj}\)
13: \(i \leftarrow i + 1\)
14: \(\text{end while}\)
15: \(\text{sort all clusters by slopes and divide into } n \text{ subsets}\)
16: \(i \leftarrow 0, \text{clustd} \leftarrow \{\}\)

17: \(\text{while } i < n \text{ do} \)
18: \(\text{sort all clusters in subset } i \text{ by size}\)
19: \(\text{keep a cluster } ^{j}cd \text{ in clustd when it meets:}\)
20: \(a) \delta_j \leq Th_5\)
21: \(b) \text{size of } ^{j}cd \geq Th_n\)
22: \(i \leftarrow i + 1\)
23: \(\text{end while}\)
24: \(\text{clustd}_1 \leftarrow \{\}\)
25: \(\text{Keep } Th_m \text{ clusters into clustd}_1 \text{ every } Th_l \text{ m on the map}\)
26: \(\text{clustd}_2 \leftarrow \{\}\)
27: \(j \leftarrow 0\)
28: \(\text{while } j < |\text{clustd}_1| \text{ do} \)
29: \(\text{for a cluster } ^{j}cd \text{ in clustd}_1, \text{ take out 2D translations of all samples in } ^{j}cd: t_j\)
30: \((^{q}t_j, \text{cluster_id}) \leftarrow \text{Mean_shift}(t_j, \text{bandwith2}) \triangleright \text{Divide a cluster } ^{j}cd \text{ into } q \text{ sub-clusters}\)
31: \(\text{save corresponding direction cluster in clustd}_2\)
32: \(j \leftarrow j + 1\)
33: \(\text{end while}\)
2 Supplement to Jacobians

Visual reprojection error has already been applied widely in ORB-SLAM\cite{2,7,8}, relevant codes are public at \footnote{https://github.com/raulmur/ORB-SLAM2}, so we only explain point to IMLS surface error and direction projection error.

2.1 Point to IMLS surface error

In order to facilitate the following derivation of the formulas and jacobians, let:

$$ T_{nw} = \begin{bmatrix} R_{nw} & t_{nw} \\ 0 & 1 \end{bmatrix} $$

(2)

where \( n \) is the index of \( K_n \). At Step 3) in Sec. 4.2, and as is shown in Fig. 3(a) in main paper, we have denoted \( K_c \) as the keyframe to which the closest point of \( p_j^x \) belongs. And \( \cdot \) means that \( \cdot \) is at \( K_c \)'s coordinate.

There is no ready-made class for point to IMLS surface error in g2o\cite{6}, so we modified the raw Generalized ICP (GICP) \cite{10} class According to \cite{10}, different application of GICP depends on the information matrix. So here we further derive Eq.(3) in our paper:

$$ e_L(n,j) = \| (T_{cw}T_{nw}^{-1}p_j^x - c_p^j) \cdot \vec{n}_c^j \|^2 $$

(3)

$$ \iff e_{GICP} = c_{d_j}^T \cdot Q_j^{-1} \cdot c_{d_j} $$

(4)

where:

$$ c_{d_j} = T_{cw}T_{nw}^{-1}p_j^x - c_p^j $$

(5)

$$ Q_j = R_{c_j}^T \cdot \begin{bmatrix} \epsilon & 0 & 0 \\ 0 & \epsilon & 0 \\ 0 & 0 & 1 \end{bmatrix} \cdot R_{c_j} $$

(6)

\( R_{c_j} \) is rotation matrix relevant to normal \( \vec{n}_c^j \) (See \cite{10} for details). \( \epsilon \) is a small constant much less than 1, we set it to 0 in the code such that Eq. (4) totally equivalent to Eq. (3).

Jacobian matrix of Eq. (5) with respect to the pose of the two keyframes are:
\[ J_c = \frac{\partial c_j}{\partial T_{cw}} \]
\[ = \frac{\partial (T_{cw} T_{nw}^{-1} p_j^l - c_y)}{\partial T_{cw}} \]
\[ = \frac{\partial (T_{cw} T_{nw}^{-1} p_j^l)}{\partial T_{nw}} \]
\[ = \left[ - (T_{cw} T_{nw}^{-1} p_j^l)^\wedge I \right] \]
\[ = \left[ - (c_y p_j^l)^\wedge I \right] \] \hspace{1cm} (7)

\[ J_n = \frac{\partial c_j}{\partial T_{nw}} \]
\[ = \frac{\partial (T_{cw} T_{nw}^{-1} p_j^l - c_y)}{\partial T_{nw}} \]
\[ = \frac{\partial (T_{cw} T_{nw}^{-1} p_j^l)}{\partial T_{nw}} \]
\[ = \frac{\partial (T_{cw} w_{p_j^l})}{\partial T_{nw}} \]
\[ = R_{nw} \left[ R_{nw}^T (p_j^l)^\wedge - R_{nw}^T \right] \]
\[ = \left[ R_{nw} R_{nw}^T (p_j^l)^\wedge - R_{nw} R_{nw} \right] \]
\[ = \left[ R_{nw} (p_j^l)^\wedge - R_{nw} \right] \] \hspace{1cm} (8)

These jacobians are on SE(3) manifold, and is 3 \times 6. The first three columns are gradients of the rotation and the last three columns belong to the translation.

2.2 Direction projection error

We modified the original EdgeSE3ProjectXYZ in \( g^2o \) to implement our proposed direction projection error. Recall the error function : \( e_D(n) = \|v - \pi([R_{nw}\|D_w]\|^2 \Sigma \) \). For the convenience of notation, we rewrite this formula:

\[ e_D(n) = \|v - d_{w2}^{-1} K R_{nw} \hat{d}_w\|^2 \Sigma \]
\[ = \|v - d_{nz}^{-1} K \hat{d}_n\|^2 \Sigma \] \hspace{1cm} (9)

where:

\[ \hat{d}_n = R_{nw} \hat{d}_w = \begin{bmatrix} d_{nx} \\ d_{ny} \\ d_{nz} \end{bmatrix} \] \hspace{1cm} (10)

The jacobian with respect to \( R_{cw} \) on SO(3) manifold is a 3 \times 3 matrix:
\[ J_n = \frac{\partial(v - d_{nz}^{-1} K \hat{d}_n)}{\partial R_{nw}} \]

\[ = \frac{\partial(v - d_{nz}^{-1} K \hat{d}_n)}{\partial \hat{d}_n} \begin{bmatrix} 0 & f_x d_{nz} & -f_x d_{nz} \\ 0 & 0 & 0 \end{bmatrix} \]

\[ = \frac{\partial(v - d_{nz}^{-1} K \hat{d}_n)}{\partial \hat{d}_n} \begin{bmatrix} 0 & f_x d_{nz} & -f_x d_{nz} \\ 0 & 0 & 0 \end{bmatrix} \]

\[ -(R_{nw} \hat{d}_w)^\wedge \]

\[ \begin{bmatrix} 0 & d_{nz} & -d_{ny} \\ 0 & 0 & d_{nx} \\ d_{ny} & -d_{nx} & 0 \end{bmatrix} \]

\[ \begin{bmatrix} f_x d_{nx} d_{ny} & -f_x (1 + \frac{d_{nx}^2}{d_{nz}^2}) & \frac{f_x d_{nx} d_{ny}}{d_{nz}^2} \\ f_y (1 + \frac{d_{nx}^2}{d_{nz}^2}) & -\frac{f_y d_{nx} d_{ny}}{d_{nz}^2} & 0 \end{bmatrix} \]

(11)

Where \((\cdot)^\wedge\) means skew-symmetric matrix of \((\cdot)\). It is obvious that the jacobians of global joint energy function is just weighted sum of all above jacobians.
Table 1: RMSE(m) of APE for KITTI 00-10. For each sequence, we report RMSE of input pose, our JVLDLoc without direction priors and our full JVLDLoc respectively. As for LiDAR odometry, we test under different $N$ which is the number of scans kept as the local map.

| KITTI Odometry       | 00 | 01 | 02 | 03 | 04 | 05 |
|----------------------|----|----|----|----|----|----|
| ORB-SLAM2 [8]        | 1.354 | 0.166 | 6.899 | 0.693 | 0.178 | 0.783 |
| w.o. direction priors | 1.367 | 3.289 | 4.669 | 0.614 | 0.169 | 0.705 |
| w. direction priors  | 1.205 | 1.999 | 3.544 | 0.530 | 0.151 | 0.593 |
| IMLS-SLAM [4]        | 3.895 | 2.412 | 7.159 | 0.650 | 0.180 | 1.789 |
| w.o. direction priors | 1.687 | 2.032 | 2.769 | 0.573 | 0.181 | 0.926 |
| w. direction priors  | 1.307 | 1.766 | 2.622 | 0.534 | 0.145 | 0.706 |
| LiDAR odometry $N=5$ | 3.407 | 3.141 | 7.758 | 0.600 | 0.217 | 2.165 |
| w.o. direction priors | 1.724 | 2.901 | 3.482 | 0.573 | 0.201 | 1.010 |
| w. direction priors  | 1.324 | 1.788 | 2.360 | 0.542 | 0.160 | 0.693 |
| LiDAR odometry $N=1$ | 6.849 | 35.03 | 12.74 | 0.839 | 0.265 | 3.379 |
| w.o. direction priors | 2.027 | 5.932 | 4.414 | 0.764 | 0.186 | 1.286 |
| w. direction priors  | 1.616 | 3.097 | 3.015 | 0.585 | 0.175 | 0.802 |

Supplement to Experiments

Limited by the number of pages of formal submission, we put the remaining figures and tables of experiments here. The following content is still divided like the corresponding part of our main paper.

3.1 Improvements over Prior Map

We have set another two baselines as input prior poses including IMLS-SLAM[4] and our LiDAR odometry. The RMSE of trajectory are listed in Table 1, which
is the full version of Table 1 in main paper. Regardless of the quality of those four prior map, JVLDLoc all get lower drift on the average RMSE of KITTI 00-10.

Here we present more mapping comparison on KITTI sequences in Fig. 2, Fig. 3, Fig. 4, Fig. 5, Fig. 6. These mapping results qualitatively prove that our method can indeed alleviate the global drift.

3.2 Effects of Direction Priors

Besides Fig. 6 in our submission, Fig. 1 presents another intuitive trajectory comparisons on KITTI 01. Apparently, we can get less drift after long distance under direction priors.

3.3 Comparison to Other Methods on KITTI Odometry Dataset

As a complete to Table 3 in main paper, Table 2 shows comparison of localization results on full KITTI 00-10.

3.4 Ablation Study

Different Number of Direction Priors For each KITTI sequence’s relative rotation error and APE RMSE under different $V_{\text{interval}}$, see Table 3. Note that $V_{\text{interval}} = 0$ means no reduction to direction priors. Besides Fig. 7 in main paper, Fig. 8 shows similiar results in line with the conclusion in our paper. In this experiment, the input prior is LiDAR odometry when $N = 1$.

Different Scan Map Size of Keyframe This part we explore the influence of different scan map size $N$ of each keyframe when it constructs point to IMLS surface error in local mapping. In this experiment, the input prior is IMLS-SLAM[4]. Table 4 presents relative rotation error and RMSE on each KITTI
Table 2: Comparison to other methods on KITTI 00-10. \( t_{rel} \) and \( r_{rel} \) are relative translational error (\%) and rotational error (\(^\circ/100m\)). (NA : Results are Not Available from paper).

| Method                  | 00  | 01  | 02  | 03  | 04  | 05  |
|-------------------------|-----|-----|-----|-----|-----|-----|
| IMLS-SLAM [4]           | 0.50| 0.18| 3.90| 0.82| 0.10| 2.41|
| LIMO-PL [5]             | 0.99| NA  | NA  | 1.87| NA  | 6.45|
| DVL-SLAM [12]           | 0.83| NA  | 1.47| NA  | 1.11| NA  |
| TVL-SLAM-No calib [3]   | 0.59| NA  | 0.84| 6.56| 0.74| 2.16|
| TVL-SLAM-VL calib [3]   | 0.57| NA  | 0.86| 4.40| 0.67| 1.87|
| LIO-SAM [11]            | 253 | 31.0| 956 | 2.94| 0.60| 19.8 |
| JVLDLoc input IMLS      | 0.63| 0.15| 1.31| 0.88| 0.06| 1.77|

| Method                  | 06  | 07  | 08  | 09  | 10  |
|-------------------------|-----|-----|-----|-----|-----|
| IMLS-SLAM [4]           | 0.33| 0.08| 0.46| 0.33| 0.14| 0.46|
| LIMO-PL [5]             | 0.61| NA  | 0.56| NA  | 1.27| NA  |
| DVL-SLAM [12]           | 0.92| NA  | 1.26| NA  | 1.32| NA  |
| TVL-SLAM-No calib [3]   | 0.32| NA  | 0.32| 0.36| NA  | 0.37|
| TVL-SLAM-VL calib [3]   | 0.31| NA  | 0.28| NA  | 0.35| NA  |
| LIO-SAM [11]            | 0.39| 0.30| 0.49| 0.29| 0.48| NA  |
| JVLDLoc input IMLS      | 0.33| 0.07| 0.30| 0.33| 0.13| 0.33|

sequence. Fig. 7 also plots average relative translation error, rotation error and EMSE on KITTI 00-10. It is reasonable that the more scan kept in the local map, the better the localization result are. Because each current point cloud will have longer and stronger constraints on geometric consistency to its pose. However, when the size of local map is too large, the gain in accuracy won’t be as noticeable when \( N \) is small.
Table 3: The comparison of different number of direction priors on KITTI 00-10. each value in table is $r_{rel}(^\circ/100m)/APE(RMSE)(m)$.

| V_interval | 00    | 01    | 02    | 03    | 04    | 05    |
|------------|-------|-------|-------|-------|-------|-------|
| 0          | 0.199/1.616 | 0.137/3.097 | 0.135/3.015 | 0.161/0.585 | 0.125/0.175 | 0.133/0.802 |
| 5          | 0.185/1.555 | 0.139/3.071 | 0.139/3.116 | 0.176/0.593 | 0.126/0.182 | 0.131/0.733 |
| 10         | 0.189/1.659 | 0.143/3.242 | 0.140/3.164 | 0.174/0.590 | 0.127/0.182 | 0.139/0.798 |
| 15         | 0.203/1.528 | 0.143/3.116 | 0.150/3.073 | 0.174/0.569 | 0.131/0.181 | 0.135/0.867 |
| 25         | 0.215/1.790 | 0.135/2.953 | 0.144/3.387 | 0.173/0.579 | 0.131/0.187 | 0.144/0.724 |
| 50         | 0.215/1.678 | 0.138/3.099 | 0.141/2.987 | 0.184/0.559 | 0.128/0.174 | 0.156/0.941 |
| 100        | 0.225/1.737 | 0.136/3.020 | 0.141/2.730 | 0.187/0.559 | 0.123/0.193 | 0.138/0.888 |

| N          | 00    | 01    | 02    | 03    | 04    | 05    |
|------------|-------|-------|-------|-------|-------|-------|
| 5          | 0.071/0.302 | 0.202/0.521 | 0.230/3.131 | 0.217/1.828 | 0.250/0.933 | 0.169/1.455 |
| 10         | 0.074/0.295 | 0.188/0.475 | 0.253/3.194 | 0.234/1.792 | 0.227/0.891 | 0.170/1.445 |
| 15         | 0.076/0.307 | 0.178/0.478 | 0.277/3.571 | 0.225/1.990 | 0.242/0.938 | 0.176/1.511 |
| 25         | 0.077/0.313 | 0.191/0.510 | 0.279/3.711 | 0.233/1.970 | 0.255/0.956 | 0.180/1.553 |
| 50         | 0.074/0.310 | 0.175/0.437 | 0.265/3.298 | 0.229/1.748 | 0.256/0.961 | 0.178/1.472 |
| 100        | 0.074/0.311 | 0.199/0.553 | 0.273/3.412 | 0.224/1.820 | 0.219/0.901 | 0.176/1.466 |

Table 4: The comparison of different scan map size on KITTI 00-10. each value in table is $r_{rel}(^\circ/100m)/APE(RMSE)(m)$.

| N          | 00    | 01    | 02    | 03    | 04    | 05    |
|------------|-------|-------|-------|-------|-------|-------|
| 5          | 0.241/2.372 | 0.102/2.881 | 0.235/3.381 | 0.158/0.652 | 0.109/0.122 | 0.101/0.684 |
| 10         | 0.177/1.462 | 0.069/2.187 | 0.182/3.119 | 0.174/0.582 | 0.121/0.143 | 0.103/0.866 |
| 15         | 0.157/1.372 | 0.075/2.398 | 0.156/2.957 | 0.169/0.565 | 0.115/0.149 | 0.101/0.684 |
| 25         | 0.150/1.305 | 0.064/2.034 | 0.134/2.845 | 0.164/0.552 | 0.106/0.152 | 0.094/0.683 |
| 50         | 0.147/1.295 | 0.062/1.835 | 0.120/2.771 | 0.184/0.567 | 0.105/0.145 | 0.089/0.707 |
| 100        | 0.149/1.307 | 0.063/1.766 | 0.111/2.622 | 0.144/0.534 | 0.096/0.145 | 0.090/0.706 |

| N          | 00    | 01    | 02    | 03    | 04    | 05    |
|------------|-------|-------|-------|-------|-------|-------|
| 5          | 0.075/0.318 | 0.222/0.462 | 0.256/3.989 | 0.141/1.780 | 0.230/0.959 | 0.165/1.591 |
| 10         | 0.077/0.311 | 0.191/0.450 | 0.248/3.540 | 0.163/2.319 | 0.205/0.881 | 0.153/1.434 |
| 15         | 0.079/0.342 | 0.163/0.323 | 0.218/3.196 | 0.141/1.780 | 0.230/0.959 | 0.146/1.343 |
| 25         | 0.079/0.305 | 0.141/0.425 | 0.212/3.155 | 0.133/1.922 | 0.192/0.920 | 0.134/1.300 |
| 50         | 0.078/0.303 | 0.138/0.383 | 0.194/2.934 | 0.128/1.808 | 0.164/0.868 | 0.128/1.238 |
| 100        | 0.071/0.295 | 0.131/0.332 | 0.181/2.520 | 0.118/1.383 | 0.154/0.775 | 0.119/1.126 |
Fig. 2: Mapping result of JVLDLoc on KITTI sequences.
Fig. 3: Mapping result of JVLDLoc on KITTI 08.

Fig. 4: Magnified view of mapping result on KITTI 00, 05, 06 from top row to bottom row. Each column is the result of different methods. All above figures are the enlargement of the red box region of corresponding sequence in Fig. 2.
Fig. 5: Magnified view of mapping result on KITTI 07. Each column is the result of different methods. All above figures are the enlargement of the red box region of corresponding sequence in Fig. 2.

Fig. 6: Magnified view of mapping result on KITTI 08 of different methods. All above figures are the enlargement of the red box region in Fig. 3.
Fig. 7: The average result under different scan map size of keyframe on KITTI 00-10. From left to right is relative translation error, rotation error and APE respectively. The x label is values of different N.

Fig. 8: left: Comparison of APE on KITTI 09. The x-coordinate is frame index. right: Box plots of APE on KITTI 05. full refers to no downsampling, WOdirection refers to no direction priors. Other lines respectively correspond to different Vinterval. Take ds2.txt as an example, it is the APE when Vinterval = 2.
References

1. Andrew, A.M.: Multiple view geometry in computer vision. Kybernetes (2001)
2. Campos, C., Elvira, R., Rodríguez, J.J.G., Montiel, J.M., Tardós, J.D.: Orb-slam3: An accurate open-source library for visual, visual-inertial, and multimap slam. IEEE Trans. Robot. (2021)
3. Chou, C.C., Chou, C.F.: Efficient and accurate tightly-coupled visual-lidar slam. IEEE Trans. Intell. Transp. Syst. (2021)
4. Deschaud, J.E.: Imls-slam: scan-to-model matching based on 3d data. In: Proc IEEE Int. Conf. Rob. Autom. (ICRA). pp. 2480–2485. IEEE (2018)
5. Huang, S.S., Ma, Z., Mu, T.J., Fu, H., Hu, S.: Lidar-monocular visual odometry using point and line features. In: Proc IEEE Int. Conf. Rob. Autom. (ICRA). pp. 1091–1097 (2020)
6. Kümmerle, R., Grisetti, G., Strasdat, H., Konolige, K., Burgard, W.: g 2 o: A general framework for graph optimization. In: Proc IEEE Int. Conf. Rob. Autom. (ICRA). pp. 3607–3613. IEEE (2011)
7. Mur-Artal, R., Montiel, J.M.M., Tardós, J.D.: Orb-slam: a versatile and accurate monocular slam system. IEEE Trans. Robot. 31(5), 1147–1163 (2015)
8. Mur-Artal, R., Tardós, J.D.: Orb-slam2: An open-source slam system for monocular, stereo, and rgb-d cameras. IEEE Trans. Robot. 33(5), 1255–1262 (2017)
9. Rusu, R.B., Cousins, S.: 3D is here: Point Cloud Library (PCL). In: Proc IEEE Int. Conf. Rob. Autom. (ICRA). Shanghai, China (May 9-13 2011)
10. Segal, A., Haehnel, D., Thrun, S.: Generalized-icp. In: Robotics: science and systems. vol. 2, p. 435. Seattle, WA (2009)
11. Shan, T., Englot, B., Meyers, D., Wang, W., Ratti, C., Rus, D.: Lio-sam: Tightly-coupled lidar inertial odometry via smoothing and mapping. In: Proc IEEE Int. Conf. Intell. Rob. Syst. (IROS). pp. 5135–5142. IEEE (2020)
12. Shin, Y.S., Park, Y.S., Kim, A.: Dvl-slam: sparse depth enhanced direct visual-lidar slam. Auton. Robots 44, 115–130 (2020)
13. Yin, D., Zhang, Q., Liu, J., Liang, X., Wang, Y., Maanpää, J., Ma, H., Hyyppä, J., Chen, R.: Cae-lo: Lidar odometry leveraging fully unsupervised convolutional auto-encoder for interest point detection and feature description. arXiv preprint arXiv:2001.01354 (2020)