Probabilistic Data Association for Semantic SLAM with Loop Closure Detection

Aohan(Roger) Mei, Can Jiang, Manas Buragohain, Owen Winship, Yidong Du

***Abstract*— Typical Simultaneous localization and mapping(SLAM) methods utilize relatively low-level geometric information like corners, edges, and surfaces. As the advance of techniques of object detection, it's more prevalent to use semantic information which contains the landmark classes. While this information is usually used by a hard data association, which is independent of the continuous optimization using geometric and inertial information. This project utilizes semantic information in a subproblem of optimization as soft data association to achieve optimal performance. We also move one step forward with incorporating loop-closure detection by using bag-of-word method. The project is tested on the KITTI dataset.**

I. INTRODUCTION

Simultaneous localization and mapping(SLAM) is a well-known field in robotics, which aims to estimate the position of robot and mapping the unknown environment. Traditional SLAM methods use geometric features like corners[11], edges[13], and surfaces[10] which can not be interpreted meaningfully. As the development of methods for object recognition[17], some recent work extract both metric and semantic features but consider metric and semantic information separately which does not allow confidence of object recognition to influence the result.

SLAM methods in early age often use filtering which only considers the most recent pose. And work after that in a long while has been focusing on the simplification of filtering method. More recently, nonlinear optimization became the most important method of SLAM, while this method can be traced back to [15]. Recent state of art work explored the sparse linear algebra perspective [**Dellaertsq**] and iterative optimization method [12]. While these systems still rely on linearization of sensing and motion models.

Our project is mainly motivated by the work of Bowman et al [1]. This work utilizes iterative optimization method framework, and it utilizes inertial, geometric and semantic information together in a better way. Semantic, geometric and inertial information are considered as factors in factor graph in optimization process. And instead of ignoring the confidence of object detection, this work considers it as a discrete optimization sub-problem by computing the weights for landmarks which are used in pose graph optimization. No combination of landmark and observation in data association is discarded by using this method.

In long term and especially large environment SLAM, loop closure is also important to improve the robustness by recognizing the previous scene when re-observing the past scene which is not observed for long . The method of loop closure in this project is mainly inspired by the work of Lopez et. al.[6][14], which is based on bag of words and geometrical check.

In this paper we implement a Semantic SLAM framework tested by KITTI dataset, and the code can be found on https://github.com/YidongDu/Team-8-final-project. You could also have access to our project video on https://youtu.be/ssupiMpUU20. In section II we introduce our frond end which includes inertial odometry, visual odometry, landmark detector and loop closure. In section III we break down our back end which is the optimization process. And in section IV we show and analyse our result.

II. FRONT-END CONSTRUCTION

A. Inertial Odometry

Inertial information is obtained from an inertial measurement unit (IMU) and is used as a constraint between poses[1]. Between each pose, we pre-integrate batches of IMU measurements to produce a relative SE(3) transition matrix, which serves as a factor in GTSAM.

B. Visual Odometry

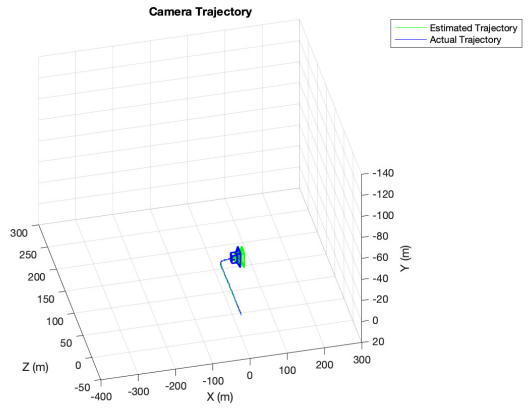

Fig. 1. Visual Odometry

C. Landmark Detector

Our project follows the MATLAB Monocular Visual Odometry tutorial to construct our visual odometry. After extracting SURF features of the images, we estimate the pose of the second view by estimatinng the essential matrix and decomposing it into camera location and orientation. We treat the GPS data as the ground truth and use the ground truth data to normalize our visual odometry data.

Fig. 2. Bounding Box and Class Scores predicted by YOLOv3 Network

Then we bootstrap estimating camera trajectory using global bundle adjustment. We eliminate outliers using the epipolar constraint. Then we find the 3D-to-2D correspondences between points triangulated from the previous two views and the current view, computing the world camera pose for the current view by solving the PnP problem.

At last, we estimate the remaining camera trajectory using windowed bundle adjustment and we outpt the visual odometry data as a relative SE(3) transition matrix which would later be added as factors in GTSAM.

During the construction process, we found the error would accumulate during the dataset scale increases. Hence, we slice the image dataset into serveral smaller sub-dataset to avoid drift.The visualization of one dataset slice is shown as Figure 1.

The detection of landmarks and its associated semantic features is broken down into two steps:

1) Object Detection: Departing from the original implementation of a deformable parts model (DPM) [4],[18],[3] we train a YOLOv3 [17] object detector network on the KITTI dataset. YOLO is a single stage object detection convolutional neural network, which simultaneously predicts multiple bounding boxes and object class probabilities for those boxes in a real-time setting.

We train the YOLO network using the KITTI 2D object detection data [7] for the classes mentioned in Table I. The trained network returns the pixel centroid of the bounding boxes, the dimensions of the bounding boxes, the class name along with the associated class probabilities for the detected landmarks which forms the basis of the semantic information.

Class Name
Car
Van
Truck
Pedestrian
Person_sitting
Cyclist
Tram
Misc

TABLE I

CLASS LABELS FOR KITTI DATASET

2) Stereo Depth Estimation: The KITTI dataset[8] provides measurements from a whole suite of sensors such as IMU, Cameras as well as Velodyne LIDAR for each keyframe. The LIDAR sensor provides an accurate point cloud representation of the surrounding location for each keyframe location. However,the LIDAR representation of the environment is sparse which requires us to iteratively search for the nearest set of points associated to a specific pixel location and approximate the depth at the specific pixel coordinate.In addition, LIDAR sensors are expensive to obtain and operate. For our project, we explore the possibility

Fig. 3. Disparity Map generated by Group-wise Correlation Network

of taking advantage of the stereo camera image data provided by the dataset [16] for estimating the real world depth at specific pixel location. We train the Group-wise Correlation Network (GwCNet) as proposed by Guo et. al.[9]. The GwCNet takes in input a pair of stereo images and gives the disparity map as an output. We then sample the disparity value at the pixel centroid coordinates of the object bounding boxes obtained from the previous step to calculate the relative world depth estimate from the camera using Equation (1).

$$depth = \frac{focal\ length(f) \times baseline(b)}{disparity} \tag{1}$$

A projective geometry problem can formulated as Equation (2)between the real-world 3D coordinate $X = (x, y, z, 1)^T$ and the pixel coordinates $x = (u, v, 1)^T$ as:

$$X = P_{rect}\, x \tag{2}$$

where,

$$P_{rect} = \begin{pmatrix} f & 0 & c_u & -f_u b \\ 0 & f & c_v & 0 \\ 0 & 0 & 1 & 0 \end{pmatrix} \tag{3}$$

is the rectified projection matrix between the left and right camera, f is the focal length, c_u and c_v are the camera centre pixel coordinates, and b is the baseline between the cameras.

Solving the Equation (2), provides us with the normalized 3D values, which we scale with the depth to obtain the relative 3D location of the landmark with respect to the current camera pose.

D. Loop Closure Detector

We build the loop closure detector based on the work of Dorian et al, which mainly contains five parts

1) Binary Features: In our project we use FAST features and BRIEF descriptors. For each FAST keypoint, we draw a square patch around them and compute a BRIEF descriptor. The BRIEF descriptor of an image patch is a binary vector where each bit is the result of an intensity comparison between two of the pixels of the patch. Given patch size $S_b = 48$, we also set the number of test to perform $L_b = 256$. For a point $\boldsymbol{p}$ in an image, its BRIEF descriptor is constructed as following:

$$B^i(\boldsymbol{p}) = \begin{cases} 1 & if\ I(\boldsymbol{p}+\boldsymbol{a_i}) < I(\boldsymbol{p}+\boldsymbol{b_i}) \\ 0 & otherwise \end{cases} \ \forall i \in [1..L_b] \quad (4)$$

where $B^i(\boldsymbol{p})$ is the i-th bit of the descriptor, $I(.)$ is the intensity of the pixel in the smoothed image, and a_i and b_i are the 2D offset of the i-th test point with respect to the center of the patch, with value in $[-\frac{S_b}{2}...\frac{S_b}{2}]\times[-\frac{S_b}{2}...\frac{S_b}{2}]$, randomly selected in advance. We sample a_i and b_i in the distributions $(a_i^j) \sim \mathcal{N}(0, \frac{1}{25}(S_b)^2)$ and $(b_i^j) \sim \mathcal{N}((a_i)^j, \frac{4}{625}(S_b)^2)$

2) Image Database Construction: [remember to cite Real Time Loop Detection with Bags of Binary Words] In order to detect loop closures, we use an image database composed of a hierarchical bag of words [6]. The bag of words is a technique that converts a visual vocabulary and a set of local features into a sparse numerical vector. We build the visual vocabulary offline, by using k-means++ [6] to cluster the descriptor space into W words. In hierarchical bag of words, the vocabulary is structured as a tree. To build the tree, we first extract descriptors from a set of training images. k-means is then performed on the descriptors to form k_w clusters. k-means is recursively performed on the clusters to form a tree with W leaves, where each leaf's centroid is a word of the vocabulary. During training, we weight each word w_i with it's inverse document frequency (idf)

$$idf(i) = log\frac{N}{n_i} \quad (5)$$

where N is the number of training images and n_i the number of occurrences of word w_i in the training images.

To create a bag of words vector, $v_t \in R^W$ from an image I_t, we first extract the set of descriptors. For each descriptor, we traverse the tree to find the closest word. At each level of the tree, we select the node with the minimal Hamming distance from the descriptor, until we reach the closest word. This allows us to calculate the term frequency (tf) of each word in I_t:

$$tf(i, I_t) = \frac{n_{iI_t}}{n_{I_t}} \quad (6)$$

where n_{iI_t} is the number of occurrences of word w_i in the image, and n_{I_t} is the total number of words in the image. The bag of words vector is constructed such that $v_t^i = tf(i, I_t) \times idf(i)$

3) Database Query: When the last image I_t is acquired, it is converted into the bag-of-words vector v_t. The database is searched for v_t, resulting in a list of matching candidates $< v_t, v_{t1} >, < v_t, v_{t2} >$ associated to their similarity scores $s(v_t, v_{tj})$, which is calculated as

$$s(v_1, v_2) = 1 - \frac{1}{2}|\frac{v_1}{|v_1|} - \frac{v_2}{|v_2|}| \quad (7)$$

We then normalize these scores with the best score we expect to obtain in this sequence for the vector v_t, obtaining the normalized similarity score η

$$\eta(v_t, v_{tj}) = \frac{s(v_t, v_t)}{s(v_t, v_{t-\Delta t})} \quad (8)$$

A threshold $\alpha = 0.6$ is defined. We then reject those matches whose $\eta(v_t, v_{tj})$ does not achieve the minimum threshold.

4) Match grouping: To prevent images that are close in time to compete among them when the database is queried, we group them into islands and treat them as only one match. Therefore, several matches $< v_t, v_{tni} >, \ldots, < v_t, v_{tmi} >$ are converted into a single match $< v_t, V_{Ti} >$ if the gaps between consecutive timestamps in t_{ni} , . . . , t_{mi} are small. The islands are also ranked according to a score H:

$$H(v_t, V_{Ti}) = \sum_{j=n_i}^{m_i} \eta(v_t, v_{tj}) \quad (9)$$

The island with the highest score is selected as matching group and continue to the temporal consistency step.

5) Geometric Consistency Check: We apply a geometrical check between any pair of images of a loop closing candidate. Since for every image we extract the strongest 128 feature points. We would rule out the candidates from the matched island which have lass than 12 correspondences.

III. Semantic SLAM

A. Expectation Maximization Formulation and Weights Computation

A SLAM problem considering the sata association as a latent variables can be stated as a maximum likelihood estimation of the sequence of poses $\mathcal{X} \triangleq \{\mathrm{x}_t\}_{t=1}^T$, the landmark positions $\mathcal{L} \triangleq \{l_m\}_{m=1}^M$, and the data association $\mathcal{D} \triangleq \{\alpha_k, \beta_k\}_{k=1}^K$ given the measurements $\mathcal{Z} \triangleq \{\mathrm{z}_k\}_{k=1}^K$:

$$\hat{\mathcal{X}}, \hat{\mathcal{L}}, \hat{\mathcal{D}} = \arg\max_{\mathcal{X},\mathcal{L},\mathcal{D}} \ \log\, p(\mathcal{Z}|\mathcal{X}, \mathcal{L}, \mathcal{D}) \quad (10)$$

Instead of using a hard decision on data associations, here we use a soft data association to utilize the whole density of $\mathcal{D}$ and rewrite the SLAM problem in expectation maximization (EM) formulation:

$$\begin{aligned}\mathcal{X}^{i+1},\mathcal{L}^{i+1} &= \arg\max_{\mathcal{X},\mathcal{L}} \mathbb{E}_{\mathcal{D}}\left[\log p(\mathcal{Z}|\mathcal{X},\mathcal{L},\mathcal{D})|\mathcal{X}^i,\mathcal{L}^i,\mathcal{Z}\right] \\ &= \arg\max_{\mathcal{X},\mathcal{L}} \sum_{\mathcal{D}\in\mathbb{D}} p\left(\mathcal{D}|\mathcal{X}^i,\mathcal{L}^i,\mathcal{Z}\right)\log p(\mathcal{Z}|\mathcal{X},\mathcal{L},\mathcal{D}) \\ &= \arg\max_{\mathcal{X},\mathcal{L}} \sum_{\mathcal{D}\in\mathbb{D}}\sum_{k=1}^{K} p\left(\mathcal{D}|\mathcal{X}^i,\mathcal{L}^i,\mathcal{Z}\right)\log p\left(\mathbf{z}_k|\mathbf{x}_{\alpha_k},\ell_{\beta_k}\right) \\ &= \arg\max_{\mathcal{X},\mathcal{L}} \sum_{k=1}^{K}\sum_{j=1}^{M} w_{kj}^i \log p\left(\mathbf{z}_k|\mathbf{x}_{\alpha_k},\ell_j\right)\end{aligned} \tag{11}$$

where $\mathbb{D}$ is the space of all possible values of $\mathcal{D}$. $w_{kj}^i \triangleq \sum_{\mathcal{D}\in\mathbb{D}(k,j)} p\left(\mathcal{D}|\mathcal{X}^i,\mathcal{L}^i,\mathcal{Z}\right)$ is the weight which quantifies the influence of soft data association. And $\mathbb{D}(k,j) \triangleq \{\mathcal{D}\in\mathbb{D}|\beta_k = j\} \subseteq \mathbb{D}$ is the set of all possible data association where measurement k is assigned to landmark j.

Because there is no such data association with regard to the inertial measurements and the data association of the geometric measurements is provided by the feature tracking algorithm, we only need to deal with the data association of the semantic measurements. Suppose $p(\mathcal{D}|\mathcal{X},\mathcal{L})$ is uniform and the semantic measurement data associations are independent across keyframes. Then we can divide the semantic SLAM problem using expectation maximization algorithm written as Equation (11) into two parts: (1) computing data association weights w_{ij}^t (the "E" step) and (2) solving for the sensor states (robot poses) $\mathcal{X}$ and the landmark positions $l_{1:M}^{p,i+1}$ (the "M" step). These two steps are shown as the following equations:

$$w_{kj}^{t,(i)} = \sum_{\ell^c\in\mathcal{C}}\sum_{\mathcal{D}_t\in\mathbb{D}_t(k,j)} \kappa^{(i)}\left(\mathcal{D}_t,\ell^c\right) \quad \forall t,k,j \tag{12}$$

$$\begin{aligned}\mathcal{X}^{(i+1)},\ell_{1:M}^{p,(i+1)} = \arg\min_{\mathcal{X},\ell_{1:M}^p} &\sum_{t=1}^{T}\sum_{\mathbf{s}_k\in\mathcal{S}_t}\sum_{j=1}^{M} -w_{kj}^{t,(i)}\log p\left(\mathbf{s}_k|\mathbf{x}_t,\ell_j\right) \\ &- \log p(\mathcal{G}|\mathcal{X}) - \log p(\mathcal{I}|\mathcal{X})\end{aligned} \tag{13}$$

where $\mathcal{G}$ is geometric information, $\mathcal{I}$ is inertial information, $\mathbb{D}$ is the set of all possible data associations for measurements received at timestep t, and $\mathbb{D}_t(i,j) \subseteq \mathbb{D}$ is the set of all possible data associations for measurements received at time t such that measurement i is assigned to landmark j.

At sensor pose x_t, given the association of semantic measurement s_k and landmark l_j, the measurement likelihood can be computed by:

$$p\left(\mathbf{s}_k|\mathbf{x}_{\alpha_k},\ell_{\beta_k}\right) = p\left(s_k^c|\ell_{\beta_k}^c\right)p\left(s_k^s|\ell_{\beta_k}^c,s_k^c\right)p\left(s_k^b|\mathbf{x}_{\alpha_k},\ell_{\beta_k}^p\right) \tag{14}$$

where s_k^c is detected class, s_k^s is the detection confidence, s_k^b is the bounding box, $p(s_k^c|l_j^c)$ corresponds to the confusion matrix of the object detector, $p(s_k^s|l_j^c,s_k^c)$ corresponds to detection score, and $p(s_k^b|\mathrm{x}_t,l_j^p)$ is assumed normally distributed with mean equal to the perspective projection of the centroid of the object onto the image plane and covariance proportional to the dimensions of the detected bounding box.

Instead of using training data with ground truth label to get a confusion matrix of our object detection network, we derived a method to get a pseudo confusion matrix from the object detection outputs of the data we tested on. Each object detection output is a landmark detection with the information of the 2D and 3D position of the landmark, the estimated class of the landmark, and the class-match score between each 8 classes and the landmark. If a landmark is claimed to be of a specific class, then the score between other 7 classes and the landmark can be interpreted as magnitude of confusion to claim the landmark as another class. In this sense, for some class j, we add all the score vectors of landmarks that is claimed to be of class j and then we normalize the cumulative score vector to get the result vector where the $i-th$ entry represents the probability of claiming a landmark with real class j to be of class i. Repeat this process to each class, and then such a pseudo confusion matrix can be obtained.

B. Pose Graph Construction and Optimization

To solve the optimization problem as Equation (13), here we use the pose-graph optimization method [2][14] with the graph having vertex for each sensor state x_t and for each landmark position l_i^p and having three kinds of factors: semantic factors, geometric factors and inertial factors.

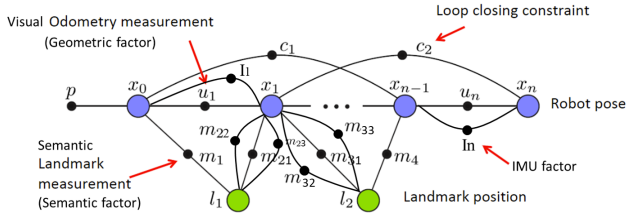

Fig. 4. Pose Graph Structure

1) Semantic Factors: The semantic measurements $s = (s^c, s^s, s^b)$ come from the object detection in front-end. A measurement s_k from sensor state x_t defines factors $f_{kj}^s(\mathrm{x}_t,l_j)$ for each visible landmark j. Since l^c is fixed in Equation (13), $p(s^c|l^c)$ and $p(s^s|l^c,s^c)$ are constant. Thus, the semantic factors can be expressed as:

$$\begin{aligned}f_{kj}^s(\mathcal{X},\mathcal{L}) &= -w_{kj}^{t,(i)}\log p\left(s_k^b|\mathbf{x}_t,\ell_j^p\right) \\ &= \left\|s_k^b - h_\pi\left(\mathbf{x}_t,\ell_j\right)\right\|^2_{\mathbf{R}_s/w_{kj}^{t,(i)}}\end{aligned} \tag{15}$$

where $h_\pi(\mathrm{x}_t,l_j)$ is the standard perspective projection of a landmark l_j onto a camera at pose x_t, $\mathbf{R}_s$ is the camera measurement covariance.

At each time step (or sensor state), the Mahalanobis distance is calculated for each landmark added to the graph

for each semantic measurement obtained from this sensor state. In the project, we only compute the Mahalanobis distance for landmarks with a positive relative X coordinate relative to the camera pose (sensor state). If the lowest Mahalanobis distance, among all the distance between all the tested landmarks and a specific semantic measurement, is not below the set threshold, a new landmark is initialized in the graph corresponding to this semantic measurement. Otherwise, new semantic factors will be added between this sensor state and each tested landmarks corresponding to this semantic measurement.

Sometimes there will be multiple bounding boxes being detected from one landmark, resulting in several semantic measurements with very close 3D position information. To reduce the influence of this kind of front-end data redundant, if the lowest Mahalanobis distance of two or more measurements is between them and a same candidate landmark, these lowest Mahalanobis distances are compared and only the measurement with the lowest "lowest Mahalanobis distances" remained, others being wiped out.

2) Geometric Factors: We have converted geometric point measurement (SURF) into relative SE(3) transformations between poses using monocular visual odometry. Now rewrite the term corresponding to geometric factors in Equation (13) as

$$-\log\, p(\mathcal{G}|\mathcal{X}) = -\Sigma_{t=1}^{T-1} \log\, p(\mathrm{g}_t|\mathrm{x}_t, \mathrm{x}_{t+1}) \tag{16}$$

where g_t is relative SE(3) transformations between sensor state (robot pose) x_t and x_{t+1} from visual odometry result.

Thus, the factor constraining the camera poses x_t and x_{t+1} is

$$f_t^g(\mathcal{X}) = ||[\log(\mathrm{g}_t^{-1}\mathrm{x}_t^{-1}\mathrm{x}_{t+1})]^\vee||_{\mathbf{R}_g} \tag{17}$$

where $(.)^\vee$ is the transformation from se(3) to $\mathbb{R}^6$, and $\mathbf{R}_g$ is the SE(3) measurement covariance.

3) Inertial Factors: The accelerometer and gyroscope measurements are also considered in optimization. By using method of preintegration, the relative pose difference(difference in position, velocity and orientation) of two keyframes can estimated. And this relative pose difference can be used by residuals expression as a function of poses of two keyframes provided in [5]. The expression is provided along with noise covariance

$$\begin{aligned} f_i^{\mathcal{I}}(\mathcal{X}) &= -\log p\left(\mathcal{I}_{ij}|\mathcal{X}\right) \\ &= \left\|\mathbf{r}_{\mathcal{I}_{ij}}\right\|_{\Sigma_{ij}}^2 \end{aligned} \tag{18}$$

The full pose graph optimization corresponding to Equation (13) is

$$\begin{aligned} \hat{\mathbf{x}}_{(1:T)}, \hat{\ell}_{1:M}^{p,(i+1)} = \underset{\mathcal{X},\ell_{1:M}^p}{\arg\min} & \sum_{k=1}^{K}\sum_{j=1}^{M} f_{kj}^s(\mathcal{X},\mathcal{L}) \\ & + \sum_{t=1}^{T} f_t^g(\mathcal{X}) + \sum_{t=1}^{T} f_t^{\mathcal{I}}(\mathcal{X}) \end{aligned} \tag{19}$$

We solve this within the iSAM2 framework [12].

IV. Result and Analysis

Our result after running GTSAM is shown as Figure 5.

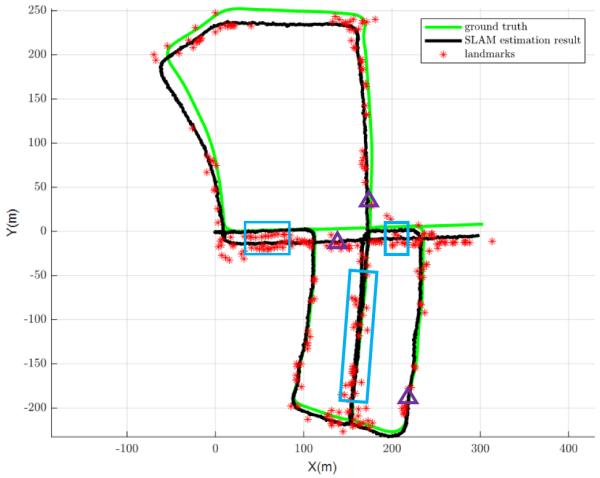

Fig. 5. Trajectory comparison between SLAM output and Ground Truth

The correct loop-closure candidates are selected out and marked with blue bounding box. From visual effect we could conclude that the bag-of-word loop closure detection algorithm works pretty well. However, there are also some misdetections which are sampled and marked by purple triangles. The analytical reason behind it might be:

- We did not conduct the temporal consistency check, which might cause some exceptions exists.
- The tree construction parameter might still need to be tuned to make the information traversing the tree much more informative and representative.

We also compare the MSE error without semantic information and with semantic information, which is displayed in Table II.

Semantic Information	MSE error of pose
No	50.125
Yes	35.3392

TABLE II
MSE ERROR COMPARISON

With MSE error decreasing when we incorporate semantic information, we could manifest the functionality of our algorithm.

V. Future Work

Due to the time limit, we only detect the loop closure pairs in our front end. In the future, we could add projection factors between loop closure pair nodes and merge them. We believe this step will further reduce the performance error, though it might slightly decrease the calculation speed.

References

[1] S. L. Bowman et al. "Probabilistic data association for semantic SLAM". In: *2017 IEEE International Conference on Robotics and Automation (ICRA)*. 2017, pp. 1722–1729.

[2] Frank Dellaert. *Factor graphs and GTSAM: A hands-on introduction*. Tech. rep. Georgia Institute of Technology, 2012.

[3] Charles Dubout and Francois Fleuret. "Deformable Part Models with Individual Part Scaling". In: Jan. 2013, pp. 28.1–28.11. ISBN: 1-901725-49-9. DOI: 10.5244/C.27.28.

[4] P. F. Felzenszwalb et al. "Object Detection with Discriminatively Trained Part-Based Models". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 32.9 (2010), pp. 1627–1645.

[5] Christian Forster et al. "IMU Preintegration on Manifold for Efficient Visual-Inertial Maximum-a-Posteriori Estimation". In: *Robotics: Science and Systems*. 2015.

[6] Dorian Gálvez-López and Juan Tardos. "Real-time loop detection with bags of binary words". In: Sept. 2011, pp. 51–58. DOI: 10.1109/IROS.2011.6094885.

[7] Andreas Geiger, Philip Lenz, and Raquel Urtasun. "Are we ready for Autonomous Driving? The KITTI Vision Benchmark Suite". In: *Conference on Computer Vision and Pattern Recognition (CVPR)*. 2012.

[8] Andreas Geiger et al. "Vision meets Robotics: The KITTI Dataset". In: *International Journal of Robotics Research (IJRR)* (2013).

[9] Xiaoyang Guo et al. "Group-wise Correlation Stereo Network". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2019, pp. 3273–3282.

[10] Peter Henry et al. "RGB-D Mapping: Using Kinect-Style Depth Cameras for Dense 3D Modeling of Indoor Environments". In: *International Journal of Robotic Research - IJRR* 31 (Apr. 2012), pp. 647–663. DOI: 10.1177/0278364911434148.

[11] J. A. Hesch et al. "Consistency Analysis and Improvement of Vision-aided Inertial Navigation". In: *IEEE Transactions on Robotics* 30.1 (2014), pp. 158–176.

[12] Michael Kaess et al. "iSAM2: Incremental smoothing and mapping using the Bayes tree". In: *The International Journal of Robotics Research* 31 (2012), pp. 216–235.

[13] D. G. Kottas and S. I. Roumeliotis. "Efficient and consistent vision-aided inertial navigation using line observations". In: *2013 IEEE International Conference on Robotics and Automation*. 2013, pp. 1540–1547.

[14] R. Kümmerle et al. "G2o: A general framework for graph optimization". In: *2011 IEEE International Conference on Robotics and Automation*. 2011, pp. 3607–3613.

[15] F. Lu and E. Milios. "Globally Consistent Range Scan Alignment for Environment Mapping". In: *Auton. Robots* 4.4 (Oct. 1997), pp. 333–349. ISSN: 0929-5593. DOI: 10.1023/A:1008854305733. URL: https://doi.org/10.1023/A:1008854305733.

[16] Moritz Menze and Andreas Geiger. "Object Scene Flow for Autonomous Vehicles". In: *Conference on Computer Vision and Pattern Recognition (CVPR)*. 2015.

[17] Joseph Redmon and Ali Farhadi. "YOLOv3: An Incremental Improvement". In: *CoRR* abs/1804.02767 (2018). arXiv: 1804.02767. URL: http://arxiv.org/abs/1804.02767.

[18] Menglong Zhu et al. "Active Deformable Part Models Inference". In: Sept. 2014. DOI: 10.1007/978-3-319-10584-0_19.