Supplementary Material for VSO: Visual Semantic Odometry

Konstantinos-Nektarios Lianos [1] Johannes L. Schönberger [2]
Marc Pollefeys [2,3] Torsten Sattler [2]

[1] Geomagical Labs, Inc., USA [3] Microsoft, Switzerland
[2] Department of Computer Science, ETH Zürich, Switzerland
nelianos@geomagical.com {jsch,marc.pollefeys,sattlert}@inf.ethz.ch

This supplementary document presents the following additional material:

1. Detailed derivations for both the semantic observation likelihood model (Sec. 1) and our EM-based optimization (Sec. 2).
2. Implementation details for integrating the semantic framework into existing VO systems, as well as details on weighting the residual terms based on the uncertainty of the 3D scene points (Sec. 3).
3. Additional results of our method to provide a better intuition on the effectiveness of the different components (Sec. 4).

1 Observation Likelihood Formulation for Visual Semantic Odometry

Our visual semantic odometry framework obtains a set of camera poses $\{\hat{T}\}$ and 3D points $\{\hat{X}\}$ by optimizing an energy consisting of a base cost term and a semantic cost term

$$\{\hat{X}\}, \{\hat{T}\} = \arg\min \; E_{base} + \lambda E_{sem} \;\; , \tag{1}$$

where λ weights the different terms. Here, the base cost corresponds to the term produced by a direct or indirect visual odometry approach, while the semantic cost term is the contribution of our work. It is defined as

$$E_{sem} = \sum_{k} \sum_{i} e_{sem}(k, i) \;\; , \tag{2}$$

where each term $e_{sem}(k, i)$ associates the camera pose T_k and the point P_i, represented by its label Z_i and location X_i, with the semantic image observation S_k (*i.e.*, the semantic segmentation of the k-th image). The semantic cost term $e_{sem}(k, i)$ is defined as (*c.f.* Eq. 5 in Sec. 3.2 of the paper)

$$e_{sem}(k, i) = \sum_{c \in \mathcal{C}} w_i^{(c)} \log(p(S_k | T_k, X_i, Z_i = c)) \;\; , \tag{3}$$

where $p(S_k|T_k, X_i, Z_i = c)$ is the observation likelihood. As defined in Eq. 4 in Sec. 3.2 of the paper, we model the observation likelihood as

$$p(S_k|T_k, X_i, Z_i = c) \propto e^{-\frac{1}{2\sigma^2} DT_k^{(c)}(\pi(T_k, X_i))^2} \tag{4}$$

for the semantic observation S_k, given the camera pose T_k and point P_i with position X_i and label c. Here, $\pi(T_k, X_i)$ is the projection of point P_i into the semantic segmentation S_k using the camera pose T_k. The label estimate of a point P_i is defined as (*c.f.* Eq. 6 in Sec. 3.2 of the paper)

$$w_i^{(c)} = \frac{1}{\alpha} \prod_{k \in \mathcal{T}_i} p(S_k|T_k, X_i, Z_i = c) \ . \tag{5}$$

In the following, we derive likelihood model $p(S_k|Z_i = c, X_i, T_k)$.

We begin our derivation by marginalizing over the values for the pixel projections $u_{i,k} = \pi(T_k, X_i)$, leading to

$$p(S_k|Z_i = c, X_i, T_k) = \sum_{u_{i,k}} p(S_k, u_{i,k}|Z_i = c, X_i, T_k) \ . \tag{6}$$

Applying Bayes' rule yields

$$p(S_k|Z_i = c, X_i, T_k) = \sum_{u_{i,k}} p(S_k|u_{i,k}, Z_i = c, X_i, T_k) p(u_{i,k}|Z_i = c, X_i, T_k) \ . \tag{7}$$

Next, we exploit two conditional independences in our formulation, namely

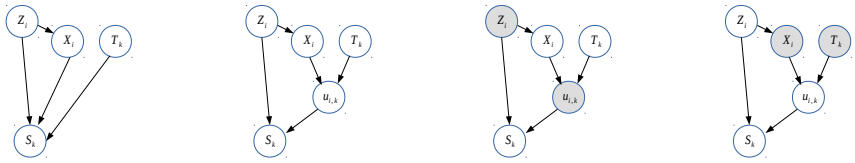
$$S_k \perp X_i, T_k \mid u_{i,k}, Z_i \ \text{ and } \ u_{i,k} \perp Z_i \mid T_k, X_i \ . \tag{8}$$

These independencies can be easily tracked in Fig. 1. The point projection $u_{i,k}$ and point label Z_i constitute the Markov Blanket of the observation $S_{i,k}$. As we know, a variable is independent of any other variables when conditioned on its Markov Blanket [5]. Also, the projection location $u_{i,k}$ is fully determined by the 3D point position X_i and the camera pose T_k, if there is no semantic observation.

Recall that if $a \perp b$ then $p(a|b) = p(a)$. Therefore, $p(S_k|u_{i,k}, Z_i = c, X_i, T_k) = p(S_k|u_{i,k}, Z_i = c)$ and $p(u_{i,k}|Z_i = c, X_i, T_k) = p(u_{i,k}|X_i, T_k)$. As a result, we obtain

$$p(S_k|Z_i = c, X_i, T_k) = \sum_{u_{i,k}} p(S_k|u_{i,k}, Z_i = c) p(u_{i,k}|X_i, T_k) \ . \tag{9}$$

The second term $p(u_{i,k}|X_i, T_k)$ in Eq. 9 corresponds to the re-projection error likelihood used by indirect methods, where it is usually modeled as a 2D Gaussian distribution $p \sim N(\mu_p, \Sigma_p)$. For direct methods, the corresponding term is typically modeled as the photometric error over a square patch (the discretization of an isotropic Gaussian) instead of a single pixel. Considering a patch rather than a single pixel typically leads to more stable results and is also applicable



(a) Initial graphical model for a residual $e_{sem}(i,k)$.

(b) Introduction of the point projection variable $u_{i,k}$, which depends on the 3D point location and the camera pose.

(c) $u_{i,k}$ and Z_i "cut" the flow of information from $\{X_i, T_k\}$ to S_k, casting them independent.

(d) If not conditioned on the semantic observation, X_i and T_k are sufficient to determine $u_{i,k}$.

Fig. 1: A graphical model perspective of the variables affecting a residual, along with the conditional independencies in (c) and (d). In bold are the conditioned variables.

to our case, *i.e.*, we could determine the semantic reprojection error by taking a small patch in the semantic segmentation S_k into account. However, we model the semantic cost as a zero re-projection Gaussian and evaluate the likelihood at a single pixel position for computational efficiency. As a result, we obtain

$$p(S_k|Z_i = c, X_i, T_k) = p(S_k|u_{i,k}, Z_i = c) \ , \tag{10}$$

where the term $p(S_k|u_{i,k}, Z_i = c)$ is the probability of semantic observation S_k, given the projection $u_{i,k}$ of the 3D point **and** that its associated semantic class is $Z_i = c$.

Assuming a uniform prior for each class in the semantic observation S_k, we obtain

$$\begin{aligned} p(S_k|u_{i,k}, Z_i = c) &\propto p(u_{i,k}, Z_i = c|S_k) \\ &= p(u_{i,k}|S_k, Z_i = c)p(Z_i = c|S_k) \ . \end{aligned} \tag{11}$$

The likelihood $p(u_{i,k}|S_k, Z_i = c)$ models the probability of the 3D point projecting to pixel position $u_{i,k}$, given the label $Z_i = c$ of the point and the semantic segmentation S_k. Intuitively, $p(u_{i,k}|S_k, Z_i = c)$ should decrease the further $u_{i,k}$ is to its closest pixel in S_k with label c. As described in Sec. 3.2 of the paper, we thus use a distance transform on the binary image obtained by only considering label c. This results in the following model for $p(u_{i,k}|S_k, Z_i = c)$:

$$p(u_{i,k}|S_k, Z_i = c) = \frac{e^{-\frac{1}{2\sigma^2} DT_k^{(c)}(u_{i,k})^2}}{\alpha_{k,c}} \ . \tag{12}$$

Here, $\alpha_{k,c}$ is a normalization factor such that the sum over all pixels equals to 1, *i.e.*,

$$\alpha_{k,c} = \sum_{u_{i,k}} e^{-\frac{1}{2\sigma^2} DT_k^{(c)}(u_{i,k})^2} \ . \tag{13}$$

The likelihood $p(Z_i = c|S_k)$ is the probability of the occurrence of class c in the semantic segmentation S_k. Naturally, it is the proportion of pixels with label c in the image S_k. For the computation, we employ the previously introduced Gaussian probability of a pixel given a label:

$$\begin{aligned} p(Z_i = c|S_k) &= \frac{\sum_{u_{i,k}} e^{-\frac{1}{2\sigma^2} DT_k^{(c)}(u_{i,k})^2}}{\sum_{c'} \sum_{u_{i,k}} e^{-\frac{1}{2\sigma^2} DT_k^{(c')}(u_{i,k})^2}} \\ &= \frac{\alpha_{k,c}}{\alpha_k} \end{aligned} \tag{14}$$

using the normalization factor

$$\alpha_k = \sum_{c'} \sum_{u_{i,k}} e^{-\frac{1}{2\sigma^2} DT_k^{(c')}(u_{i,k})^2} \quad . \tag{15}$$

Inserting Eqs. 12 and 14 into Eq.11 yields

$$\begin{aligned} p(S_k|u_{i,k}, Z_i = c) &= p(u_{i,k}|S_k, Z_i = c) p(Z_i = c|S_k) \\ &= \frac{e^{-\frac{1}{2\sigma^2} DT_k^{(c)}(u_{i,k})^2}}{\alpha_{k,c}} \cdot \frac{\alpha_{k,c}}{\alpha_k} \\ &= \frac{e^{-\frac{1}{2\sigma^2} DT_k^{(c)}(u_{i,k})^2}}{\alpha_k} \\ &\propto e^{-\frac{1}{2\sigma^2} DT_k^{(c)}(u_{i,k})^2} \quad . \end{aligned} \tag{16}$$

Using Eq. 10, we can derive the observation likelihood as

$$p(S_k|Z_i = c, X_i, T_k) \propto e^{-\frac{1}{2\sigma^2} DT_k^{(c)}(u_{i,k})^2} \quad , \tag{17}$$

which is the observation likelihood used in the paper (*c.f.* Eq. 4 above and Eq. 4 in the paper).

2 Optimization Using Expectation Maximization

As mentioned in Sec. 3.3 of the paper, we optimize the semantic term E_{sem} (*c.f.* Eq. 2) of our cost function via expectation maximization (EM). In the following, we derive our proposed EM approach.

From a probabilistic point of view, visual odometry attempts to find the Maximum Likelihood estimate (MLE) for the set of parameters θ that best explains the set of observations $\mathcal{O}$:

$$\begin{aligned} \hat{\theta} &= \underset{\theta}{\operatorname{argmax}} \log P(\mathcal{O}|\theta) = \underset{\theta}{\operatorname{argmax}} \log P(\mathcal{I}|\theta) + \log P(\mathcal{S}|\theta) \\ &= \underset{\theta}{\operatorname{argmin}} E_{base}(\theta) + \lambda E_{sem}(\theta) \quad , \end{aligned} \tag{18}$$

where λ is a tunable parameter that weights the semantic error $E_{sem}(\theta)$ relative to the base error $E_{base}(\theta)$, *i.e.*, relative to the cost of the visual odometry pipeline that is used. Here, the set θ of parameters corresponds to the set of camera poses and 3D points, *i.e.*, $\theta = \{\{X\}, \{T\}\}$. We imply that maximization of the likelihood is equivalent to minimizing an odometry cost functional. The cost $E_{base}(\theta)$ of the baseline odometry depends on the images $\mathcal{I} = \{I_k\}$ observed by the cameras. The semantic cost $E_{sem}(\theta)$ optimizes the same parameters, but uses the semantic observations $\mathcal{S} = \{S_k\}$ instead of the image observations. In the following, we focus on optimizing $E_{sem}(\theta)$ and ignore $E_{base}(\theta)$.

For computational tractability, we make the simplifying assumption that the two observations I_k (the observed image) and S_k (the image's semantic segmentation) made by camera k are independent, *i.e.*, $p(I_k, S_k|\theta_k) = p(I_k|\theta_k)\, p(S_k|\theta_k)$[1]. Intuitively, this is equivalent to assuming a "perfect" semantic classifier that produces the same semantic segmentation from the same camera pose, independently of the actual viewing conditions (illumination, noise in the image capture process, *etc.*). To facilitate the semantic optimization process, we introduce a set of latent variables $\mathcal{Z} = \{Z_1, Z_2, ..., Z_N\}$ with one latent variable Z_i per 3D point P_i (with 3D position X_i). The latent variables correspond to the label Z_i of point P_i. Since a variable Z_i only regards a single point, we have $p(\mathcal{Z}|X_i) = p(Z_i|X_i)$. We incorporate latent variables in the semantic term by marginalization, resulting in

$$E_{sem}(\theta) = \log p(\mathcal{S}|\theta) = \log \sum_{\mathcal{Z}} p(\mathcal{S}, \mathcal{Z}|\theta) \ . \tag{19}$$

Here, $\sum_{\mathcal{Z}}$ denotes the sum over the combination of all possible values of the set $\mathcal{Z}$. For N points and thus N latent variables, we have $\sum_{\mathcal{Z}} f(Z_1, ..., Z_N) = \sum_{Z_1} \sum_{Z_2} ... \sum_{Z_N} f(Z_1, ..., Z_N)$, where the notation $\sum_{Z_i}$ is a shorthand for $\sum_{Z_i = c, c \in \mathcal{C}}$.

Closed-form maximum likelihood (ML) estimation of the above cost function is intractable as the latent variables Z_i, the camera poses $\mathcal{T}$, and the 3D point positions X_i depend on each other: Z_i depends on the projection of its corresponding 3D point into multiple images and thus on X_i and the poses of the cameras observing P_i. In turn, Z_i constrains the plausible point positions X_i and poses of its observing cameras when we enforce semantic consistency between the label of point P_i and the semantic segmentations it projects into. Therefore, we apply the EM algorithm in order to iteratively find the ML solution $\hat{\theta}$ for our semantic cost. Following EM, we iteratively maximize the following functional

$$\begin{aligned} E_{EM}(\theta) &= E_{\mathcal{Z}|S,\theta_{old}}[\log p(\mathcal{S}, \mathcal{Z}|\theta)] \\ &= \sum_{\mathcal{Z}} p(\mathcal{Z}|\mathcal{S}, \theta_{old}) \log p(\mathcal{S}|\mathcal{Z}, \theta) p(\mathcal{Z}|\theta) \ , \end{aligned} \tag{20}$$

where θ_{old} is the value of the set of of parameters obtained from the previous optimization step. The term $p(\mathcal{Z}|\theta)$ is a prior on the labelling of the 3D point cloud based on the scene structure. For simplification, we assume that this prior

[1] θ_k is the set of point and pose parameters affecting observations S_k and I_k.

follows a uniform distribution. In this case, the term does not influence the optimization and we can ignore it, yielding

$$E_{EM}(\theta) = \sum_{\mathcal{Z}} p(\mathcal{Z}|\mathcal{S}, \theta_{old}) \log p(\mathcal{S}|\mathcal{Z}, \theta) \ . \tag{21}$$

The EM method proceeds to estimate the first term $p(\mathcal{Z}|S, \theta_{old})$ using the previous estimate of the parameters (E-step). Thus, the E-step estimates $p(\mathcal{Z}|S, \theta_{old})$ based on projecting the point positions from the previous iteration into the semantic segmentations using the camera poses from the previous iteration. Afterwards, the parameter values are updated while the term $p(\mathcal{Z}|S, \theta_{old})$ remains fixed (M-step).

We aim to bring the functional of Eq. 21 into a sum-of-squared-errors form, with residuals involving a point P_i and a camera frame k, as to facilitate optimization by existing least-squares solvers such as Ceres [1]. Towards this goal, we factor the likelihood $p(\mathcal{S}|\mathcal{Z}, \theta)$ appearing inside the logarithm of Eq. 21 as

$$p(\mathcal{S}|\mathcal{Z}, \theta) = \prod_{k} p(S_k|\mathcal{Z}, \theta_k) \ . \tag{22}$$

Assuming that the semantic segmentations S_k are independent of each other, we split the terms into their individual $\{i, k\}$-factors

$$p(\mathcal{S}|\mathcal{Z}, \theta) \propto \prod_{k} \prod_{i} p(S_k|Z_i, X_i, T_k) \ . \tag{23}$$

The likelihood $p(S_k|Z_i, X_i, T_k)$ is defined in Eq. 9. For cleaner notation, we write it as $f_k(X_i, T_k, Z_i)$ in the following. For the above derivation, we made the necessary simplifications of uniform priors and independent observations, necessary to impose sparsity to the objective function.

Inserting Eq. 23 into the objective function in Eq. 21 leads to

$$\begin{aligned} E_{EM}(\theta) &= \sum_{\mathcal{Z}} p(\mathcal{Z}|\mathcal{S}, \theta_{old}) \sum_{k} \sum_{i} \log f_k(X_i, T_k, Z_i) \\ &= \sum_{k} \sum_{i} \sum_{\mathcal{Z}} p(\mathcal{Z}|\mathcal{S}, \theta_{old}) \log f_k(X_i, T_k, Z_i) \ . \end{aligned} \tag{24}$$

By definition of the latent variables in $\mathcal{Z}$, they are independent and each variable Z_i depends only on its point P_i. As derived in the following, the functional in Eq. 24 can therefore be written as

$$E_{EM}(\theta) = \sum_{k} \sum_{i} \sum_{Z_i = c} \left[p(Z_i = c|\mathcal{S}, \theta_{old}) \log f_k(X_i, T_k, Z_i = c) \right] \ . \tag{25}$$

Continuing from Eq. 24, we have:

$$\begin{aligned} E_{EM}(\theta) &= \sum_{\mathcal{Z}} p(\mathcal{Z}|\mathcal{S},\theta_{old}) \sum_k \sum_i \log f_k(X_i, T_k k, Z_i) \\ &= \sum_k \sum_i \sum_{\mathcal{Z}} p(\mathcal{Z}|\mathcal{S},\theta_{old}) \log f_k(X_i, T_k, Z_i) \\ &= \sum_k \sum_i \sum_{\mathcal{Z}} \Big[\prod_j p(Z_j|\mathcal{S},\theta_{old}) \log f_k(X_i, T_k, Z_i) \Big] \ . \end{aligned} \tag{26}$$

The last step is based on the conditional independence label assumption $Z_i \perp Z_j \mid \{S, \theta\}$ (similar to Eq. 23). To continue, we need to expand the sum over the set $\mathcal{Z}$, which, as defined previously, is the sum over all values for all (categorical) variables in set $\mathcal{Z}$:

$$\begin{aligned} &E_{semantic}(\theta) = E_{EM}(\theta) \\ &= \sum_k \sum_i \Big[\sum_{Z_1} ... \sum_{Z_i} ... \sum_{Z_N} [p(Z_1|\mathcal{S},\theta_{old}) ... p(Z_i|\mathcal{S},\theta_{old}) ... p(Z_N|\mathcal{S},\theta_{old}) \\ &\qquad \cdot \log f_k(X_i, T_k, Z_i)] \Big] \ . \end{aligned} \tag{27}$$

The terms $f_k(X_i, T_k, Z_i)$ and $p(Z_i|\mathcal{S},\theta_{old})$ depend only on Z_i, so they can be pulled outside the sum over the variables Z_j:

$$\begin{aligned} E_{semantic}(\theta) = \sum_k \sum_i \Big[&\sum_{Z_i} p(Z_i|\mathcal{S},\theta_{old}) \log f_k(X_i, T_k, Z_i) \\ &\cdot \Big(\sum_{Z_1} ... \sum_{Z_N} \big(p(Z_1|\mathcal{S},\theta_{old}) ... p(Z_N|\mathcal{S},\theta_{old}) \big) \Big) \Big] \ . \end{aligned} \tag{28}$$

Each term $p(Z_j|...)$ in the rightmost sum depends on one summation variable, so the sum of products can be re-organized to a product of sums:

$$\begin{aligned} E_{semantic}(\theta) = \sum_k \sum_i \Big[&\sum_{Z_i} p(Z_i|\mathcal{S},\theta_{old}) \log f_k(X_i, T_k, Z_i) \\ &\cdot \Big(\underbrace{\sum_{Z_1} p(Z_1|\mathcal{S},\theta_{old})}_{=1} ... \underbrace{\sum_{Z_N} p(Z_N|\mathcal{S},\theta_{old})}_{=1} \Big) \Big] \ . \end{aligned} \tag{29}$$

By the definition of marginalization, we have $\sum_{Z_j} p(Z_j|\theta_{old}) = 1$. Therefore, every summation inside the rightmost parenthesis equals to 1, resulting in

$$\begin{aligned} E_{semantic}(\theta) &= \sum_k \sum_i \Big[\sum_{Z_i} p(Z_i|\mathcal{S},\theta_{old}) \log f_k(X_i, T_k, Z_i) \Big] \\ &= \sum_k \sum_i \sum_{Z_i} w_i^{(Z_i)} \log f_k(X_i, T_k, Z_i) \\ &= \sum_k \sum_i e_{sem}(i,k) \ , \end{aligned} \tag{30}$$

while *weights* $w_i^{(c)}$ *not converged* **do**
  E-step: For each point i and label $c \in \mathcal{C}$,
    compute $w_i^{(c)}$ as in Eq. 5.
  M-step: Fix $w_i^{(c)}$ and optimize Eq. 1.
end

Algorithm 1: Expectation maximization (EM) using alternating camera and point optimization with semantics.

with $w_i^{(Z_i)} = p(Z_i|\mathcal{S}, \theta_{old})$ as the probability of point P_i being of class Z_i. The result is in sparse sum-of-squared-errors form, solved efficiently as described in the paper.

All in all, the EM optimization is shown Alg. 1.

3 Implementation Details

In this section we provide some additional components of the implementation, necessary in order to enable reproducibility of the results.

3.1 Incorporating the Void Class

The label space used in our work (*c.f.* Sec. 3 of the paper) includes a *void* class that has no influence on the optimization. Because this category does not represent a specific object, we weight its specific likelihood term $p(S_k|X_i, T_k, Z_i = \mathit{void})$ as zero by setting $w_i^{(void)} = 0$. From Eq. 5 of the paper, we thus have

$$e_{sem}(i,k) = \sum_{\substack{c \in \mathcal{C} \\ c \neq void}} w_i^{(c)} \frac{1}{2\sigma^2} DT_k^{(c)}(\pi(T_k, X_i))^2 \quad . \tag{31}$$

This enables us to easily ignore classes by considering them as part of the *void* class. In addition, we consider a thin band between classes as part of the *void* class to account for the uncertainty of the classifier in this area. Thereby, points close to the semantic boundaries have smaller influence on the optimization, because a portion of their label vector contains the *void* class. In our experiments, we use a thin 1-pixel band, as our employed semantic classifier provides a relatively accurate labeling. Note that using a wider band enables our framework to use coarser semantic labelings, *e.g.*, as defined by polygons as in Cityscapes dataset.

3.2 Covariance-Based Residual Weighting

Our framework enables medium-term continuous tracking of points. Leveraging the full potential of medium-term tracking and robustly handling noise and outliers in our framework requires to model the point location uncertainty, as

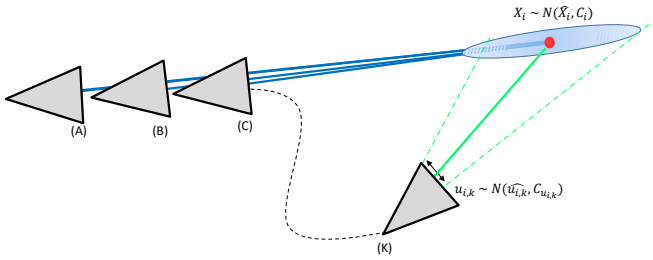


Fig. 2: Example of point position uncertainty. Blue and green lines represent the correspondences from the base odometry and the semantic constraints, respectively. Point X_i was generated and optimized by the cameras (A), (B) and (C). Due to the invariant nature of semantics, it can possibly be re-detected by a camera (K). However, the projection might have a significant uncertainty. Thus, it is necessary to take this uncertainty into account by appropriately weighting the resulting semantic constraint.

illustrated in Fig. 2. In our system, each point thus has an associated position covariance, based on the cameras from which it is triangulated. The position covariance is typically high along the major viewing direction. To account for the point position uncertainty in the semantic optimization, we introduce a weight $w_{i,k}$ for each semantic residual $e_{sem}(i,k)$.

In detail, we assume a point with position X_i has been optimized and can thus be considered a Gaussian random variable $X_i \sim \mathcal{N}(\hat{X}_i, \Sigma_i)$. Given a camera k with pose T_k, the projection of X_i is $u_{i,k} \sim \mathcal{N}(\hat{u}_{i,k}, \Sigma_{u_{i,k}})$, where the covariance $\Sigma_{u_{i,k}}$ is computed using uncertainty propagation. We also define the semantic image gradient at pixel location $u_{i,k}$ as $J_{img}(u_{i,k}) \in \mathbb{R}^2$:

$$\begin{aligned} J_{img}(u_{i,k}) &= \frac{\partial e_{sem}(i,k)}{\partial u_{i,k}} \\ &= \sum_{Z_i=c} w_i^{(c)} DT_k^{(c)}(u_{i,k}) \frac{\partial DT_k^{(c)}(u_{i,k})}{\partial u_{i,k}} \quad . \end{aligned} \tag{32}$$

Here, it is important to account for unstable residuals $e_{sem}(i,k)$ when $J_{img}(u_{i,k})$ is sensitive to small deviations in X_i. Therefore, we consider the expected image gradient of $u_{i,k}$ as $E_{u_{i,k}}[J_{img}(u_{i,k})]$, *i.e.*, the average image gradient over the projected elliptical area of the point uncertainty. It is implemented by uniformly sampling 2D point locations from the distribution $u_{i,k}$ and averaging the values of the gradients at these locations. During the optimization, we only employ the central pixel projection $\hat{u}_{i,k}$, such that the gradient of the optimizable residual is actually $J_{img}(\hat{u}_{i,k})$. In our formulation, a residual $e_{sem}(i,k)$ receives a high weight $w_{i,k}$, if the expected gradient $E_{u_{i,k}}[J_{img}(u_{i,k})]$ is similar to the optimizable gradient $J_{img}(\hat{u}_{i,k})$. The residual weight $w_{i,k}$ is computed as

$$\begin{aligned} w_{i,k} = \min(1, \max(0, & \frac{|E_{u_{i,k}}[J_{img}(u_{i,k})]|}{|J_{img}(\hat{u}_{i,k})|} \\ & \cdot \cos(\angle E_{u_{i,k}}[J_{img}(u_{i,k})], J_{img}(\hat{u}_{i,k})))) \quad , \end{aligned} \tag{33}$$

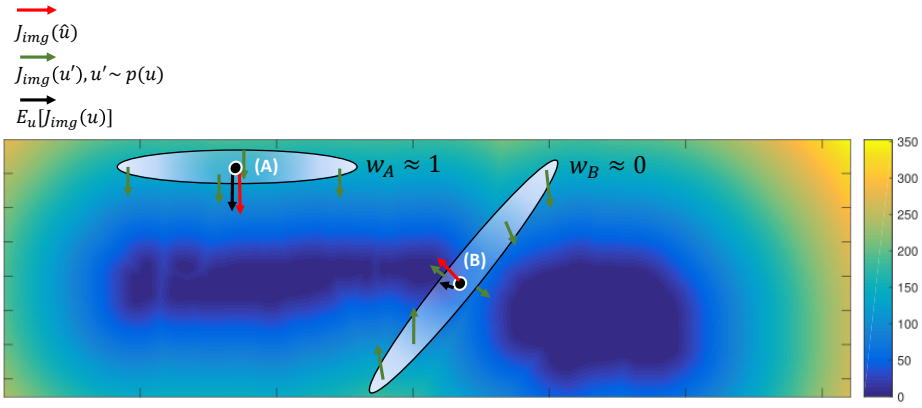


Fig. 3: Weighting computation using the semantic image gradients. Illustrated is the distance transform for the *car* class and two points A and B labeled as *car*. The black dots with white outline represent the center of projection $\hat{u_{i,k}}$, and the surrounding ellipses their projected covariance $\Sigma_{u_{i,k}}$. In red is the image gradient of the mean projection, $J_{img}(\hat{u}_{i,k})$, in green are the gradients at the sampled point locations, $J_{img}(u)$ with $u \sim u_{i,k} = p(u)$, and in black the expected image gradient $|E_{u_{i,k}}[J_{img}(u_{i,k})]|$. The weight compares the image gradient of the central pixel $\hat{u}$ with the expected image gradient over the distribution $p(u_{i,k})$. For point A, the expected image gradient has the same direction with the central image gradient. Thus, the resulting weight will be close to 1. For point B, the sampled gradients have different directions. The expected image gradient has a small norm and, therefore, the weight $w_{i,k}$ is small.

where $\angle$ calculates the angle between two vectors. The weight is truncated to the $[0, 1]$ interval to handle outliers, *e.g.*, if $|J_{img}(\hat{u}_{i,k})| \to 0$. Note that for efficiency, the weight is computed only once, when the semantic point-camera constraints are established. A graphical illustration is shown in Fig. 3. Recall that the optimizable constraint $e_{sem}(i, k)$ uses the mean estimate of point position $\hat{X}_i$ and the covariance is only used for the weight computation. The covariance weighting is especially important for PhotoBundle [2], which does not properly handle outliers. We found that our proposed covariance weighting is not strictly necessary for ORB-SLAM2 [4] due to better outlier handling in its pipeline. We thus did not use covariance weighting for ORB-SLAM2 in our experiments.

3.3 System Integration

Our framework is general in that it can be integrated into existing VO systems, either direct or indirect. In our experiments, we focused on window-based systems, which are more accurate due to re-linearization of past states. In the following, we describe how our semantic constraints can be integrated into existing VO systems in a minimally invasive manner.

Establishing Semantic Constraints. As stated in the paper, we maintain a semantic visibility list $V_{sem}(k)$ for each frame k: A candidate point P_i is inserted

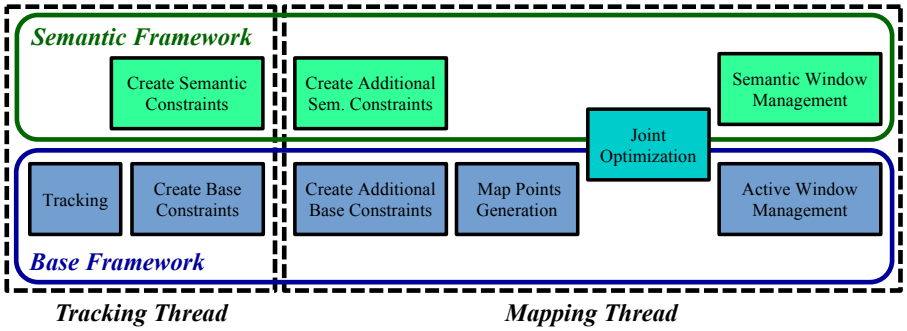


Fig. 4: Our visual semantic odometry framework: The green box depicts the semantic pipeline (ours), which is coupled with an existing base odometry pipeline (blue). The tracking and mapping threads are executed in parallel. Within a thread, the operations are executed sequentially from left to right. Notice that this high-level outline for the base system covers the architecture of most window-based visual odometry systems.

into $V_{sem}(k)$ if

$$\sqrt{e_{sem}(k,i)} \leq m \ , \tag{34}$$

i.e., if the projection of the point i into the image k is sufficiently close to an area with the semantic class of the point. Recall from Eq. 3 that the semantic cost is scaled by an uncertainty parameter σ. Thus, Eq. 34 accounts for trajectory drift causing reprojection errors and semantic segmentation errors of up to $m \cdot \sigma$ pixels away from the correct class. Allowing a certain amount of error when establishing semantic constraints is necessary to be able to handle drift.

Integration. The overall implementation of our system is illustrated in Fig. 4. In the tracking thread, the pose of the current frame k is first estimated by the base pipeline. We found no benefit in using semantics for frame-to-frame pose estimation, as existing direct and indirect methods already have very good short-term tracking performance. After creating the constraints in the base framework, we continuously attempt to create semantic constraints for every new frame to the points in $V_{sem}(k-1)$ using a threshold of $m = 0.7$. Additionally, a constraint is automatically added to $V_{sem}(k-1)$ if a point is in $V(k)$, *i.e.*, if the point is tracked by the base framework. In the mapping thread, we establish additional semantic constraints for the latest semantic keyframe. This is to establish constraints missed in frame-to-frame tracking due to noisy semantic segmentations, occlusions, or the limited field-of-view of the camera. For this test, we set a larger threshold of $m = 2$ and check if the point has a robust label estimate, *i.e.*, $\max w_i^{(c)} \geq 0.5$, to filter out semantically unstable points. This filtering step eliminates the need for a robust cost function in the optimization, resulting in faster convergence. The motivation for a more relaxed threshold m is that we aim to establish correspondence also if the trajectory is drifting. After performing joint optimization (*c.f.* Alg. 1), a frame exiting the active window of the base framework either becomes a semantic keyframe or it is discarded completely.

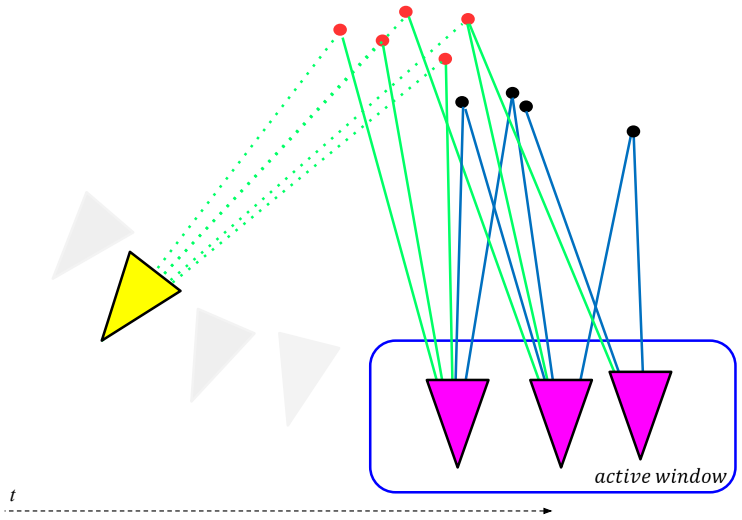


Fig. 5: Joint base-semantic optimization. The frames in the active window, which are used for optimization in the base pipeline, are shown in magenta. The base constraints are illustrated in blue solid lines. The semantic constraints are illustrated in green solid lines. The red points are semantic landmarks that only participate in semantic constraints. Their positions are kept fixed in the optimization. The yellow camera is inside the semantic window and is responsible for maintaining the list of semantic landmarks. The correspondences in dotted green lines are not used in the optimization.

As mentioned before, our proposed semantic cost is under-constrained by itself and thus not suitable to refine the 3D points jointly with the cameras. We therefore only optimize the positions of 3D points participating in semantic constraints as long as they are observed in images of the active window of the base framework, *i.e.*, as long as they are also associated to photometric or geometric constraints. The positions of points not visible in frames of the active window are *fixed* during the optimization (see also Fig. 5).

4 Additional Experimental Results

In order to facilitate a deeper understanding of the behavior of our semantic constraints, we provide additional results of our approach on the KITTI [3] dataset, using stereo ORB-SLAM2 as the base framework.

Runtime. The average run-times with / without semantics for ORB-SLAM2 and PhotoBundle are 0.27s / 0.27s and 0.71s / 0.48s for the optimization backend (the tracking front-end is not affected by our semantic constraints), respectively. A laptop with a quad-core 2.50GHz CPU, 8GB RAM, and a NVIDIA 1080 GPU was used for all evaluations. On average, PhotoBundle uses 944 semantic constraints while ORB-SLAM2 uses only 35 (low enough to have only a minor

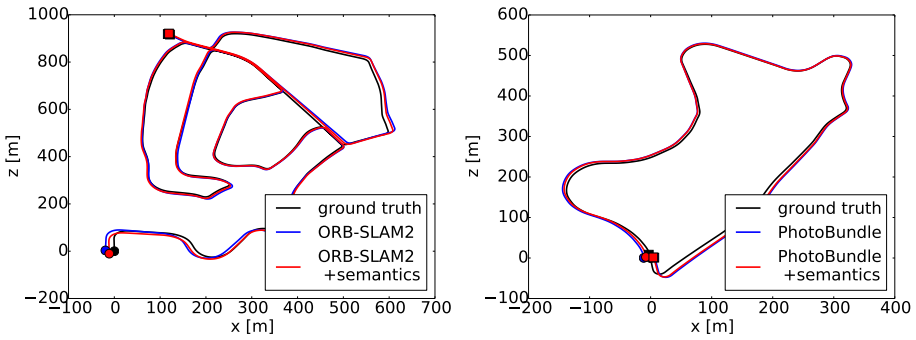


Fig. 6: Trajectories for (left) KITTI 02 and (right) KITTI 09.

impact on the overall runtime). The frame rate changes depending on the number of semantic constraints in the optimization, which is scene and motion dependent (see Tab. 2).

KITTI trajectories. Fig. 6 shows trajectory plots overlaid over the ground truth trajectories. As can be seen, using our semantic constraints reduces the drift of the estimated trajectories.

Impact of the parameters σ and λ. As stated in the paper, our system consists of two tunable parameters: The weight λ (Eq. 3 in the paper) controls the relative scale of the semantic terms with respect to the base terms, while the parameter σ (Eq. 4 in the paper) implements the concept of classification uncertainty. A higher value for σ downscales the semantic residual e_{sem} (Eq. 5 in the paper) and also causes the semantic point i to have a more ambiguous label w_i (Eq. 6 in the paper), which further reduces the impact of the semantic term. Also, a higher value for σ increases the search radius for semantic correspondences, which accounts for an inaccurate semantic segmentation (Eq. 34).

Similar to other established VO systems (*e.g.*, ORB-SLAM), the choice of parameters is empirical and should be tuned. Roughly, σ is measured in pixels and, for an image size around 1200×300, should be around 10-30px with an imperfect classifier, and up to 10px for a more accurate one. Very low values render drift correction impossible and also over-penalize weakly localized points (recall that map points are fixed when semantic constraints are optimized). The weight λ depends on the type of base cost (reprojection/photometric) and the classifier performance.

Fig. 7 visualizes how the two parameters affect the accuracy of our framework for KITTI sequence 09. A smaller value of σ better captures the uncertainty in the segmentation, but, due to non-determinism in the parallelized optimization, introduces variance in the result. As the classifier does not perform equally well for all classes, introducing per-class uncertainties σ_c should further improve accuracy. In the future, we plan to learn to adjust the parameters online from the data, *e.g.*, through class-specific uncertainty parameters $\sigma(c)$ tailored to the used segmentation algorithm.

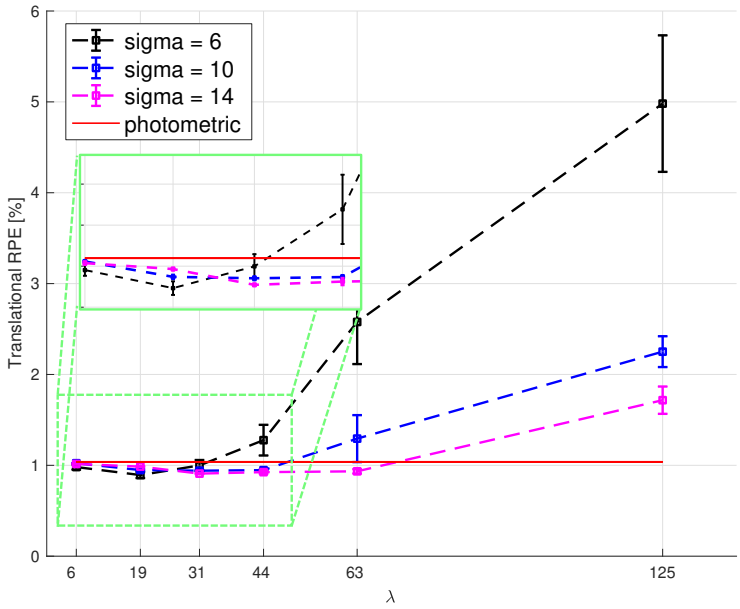


Fig. 7: Translational RPE with zoom-in (green) as a function of the tunable variables for PhotoBundle, evaluated on KITTI sequence 09. The bars denote the standard deviation for the different settings, obtained by repeating the experiment multiple times.

Exclusion of dynamic objects. As explained in the experimental section of the paper, using semantic constraints can deteriorate the performance if the objects are moving. This effect is prevalent in KITTI sequence 04 due to a moving car in front of the camera. To study the impact of the car object in this specific case, we exclude the *car* class from the optimization by considering its segments as part of the *void* class. The results are shown in Tab. 1. The translation estimate is improving by ignoring the class with moving objects. Still, due to drastic illumination changes, the classification accuracy is poor (*c.f.* Fig. 8) and thus the accuracy remains slightly below the base framework.

Table 1: Accuracy results for KITTI sequence 04, with and without using the *car* class to establish semantic constraints

	ORB-SLAM2		**+semantics**		**+semantics, -car**	
Seq.	t_{rel}	r_{rel}	t_{rel}	r_{rel}	t_{rel}	r_{rel}
04	1.19	0.13	1.30	0.10	1.23	0.13

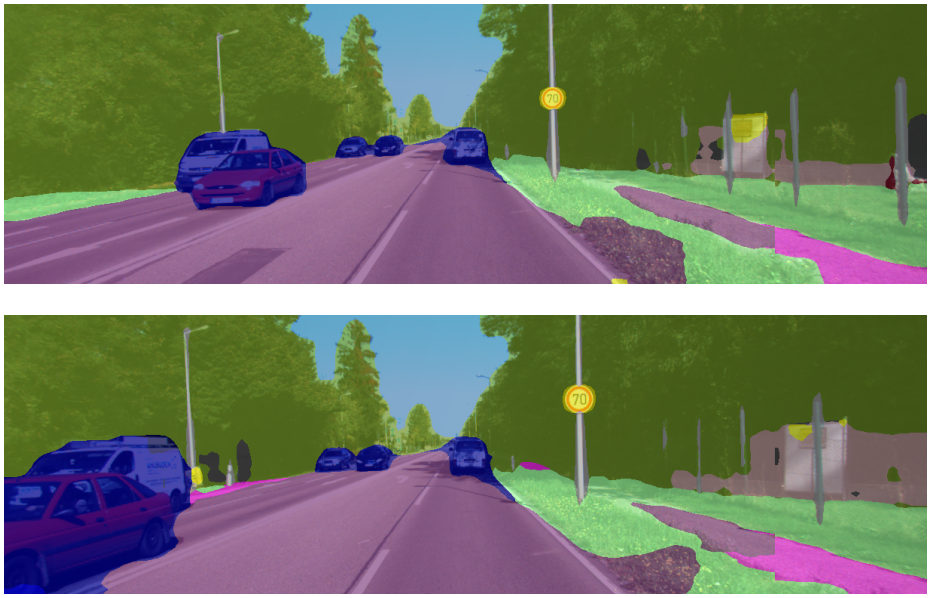


Fig. 8: Consecutive frames from KITTI sequence 04. Note that the semantic segmentation is highly inconsistent around the right sidewalk.

Table 2: Average number of optimizable semantic constraints per window optimization, for ORB-SLAM2+semantics, evaluated on the KITTI sequences

Sequence	00	02	03	05	06	07	08	09
#avg. sem. constraints	40	45	103	58	59	35	52	45

Impact of semantic constraints. Semantics are beneficial for long trajectories, where the map points change appearance but remain inside the field of view (as explained in Fig. 1 of the paper). A semantic constraint for a point-camera pair is optimized only if the point is no longer visible by the base framework. Tab. 2 shows the average number of constraints over all base+semantics optimizations. A higher number of semantic constraints typically leads to an overall higher improvement over the base framework (*c.f.* Tab. 1 and 2 in the paper).

Fig. 9 quantitatively shows for KITTI sequence 00 that points can be tracked over more frames using semantics compared to standard mono-ORB-SLAM2. This shows that even though ORB features [6] are scale invariant, the semantic framework can still track map points for a longer duration. This result clearly validates the claim that our semantic constraints allow us to establish medium-term constraints (which was shown indirectly already in the paper by demonstrating that semantic constraints reduce drift).

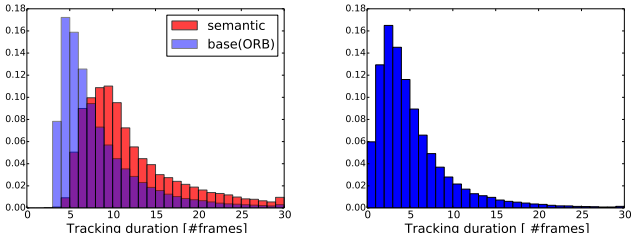


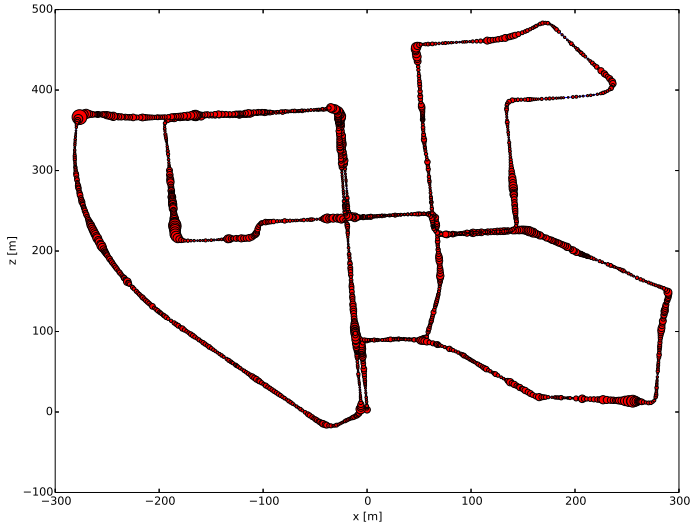
Fig. 9: Histogram over the number of frames for KITTI sequence 00 for which points can be tracked by mono-ORB-SLAM2(+semantics) (using $\sigma = 30$). The histograms for the base and semantic frameworks (left) show that our semantic constraints enable ORB-SLAM2 to track points over a longer period of time. This is also illustrated in the histogram over the differences in tracking duration between matching semantic and base points (right), where a positive value reflects a longer tracking duration using semantics. The y-axes of the plots correspond to the fraction of points in the map.

Fig. 10 illustrates in what type of environments our semantic constraints provide the largest improvement. The plots illustrate the number of semantic constraints per optimization, overlaid on the sequence trajectories. A red circle corresponds to an execution of base+semantic windowed optimization. Its location is the position of the latest keyframe of the optimization window, while the radius visualizes the number of optimizable semantic constraints. In other words, trajectory parts with larger red circles are optimized with more semantic constraints compared to parts with smaller circles. We can see that the number of semantic constraints increases with the distance covered within a straight trajectory segment. The number of semantic constraints increase as the camera moves on a straight road segment, because the points are no longer matchable using their appearance, but are trackable using their semantics. When the car turns at an intersection, the points leave the field of view and thereby the number of semantic constraints sharply decreases. The improvement observed when using semantics constraints (Tab. 1 and Fig. 5 in the paper) is correlated with the existence of long, straight road segments.

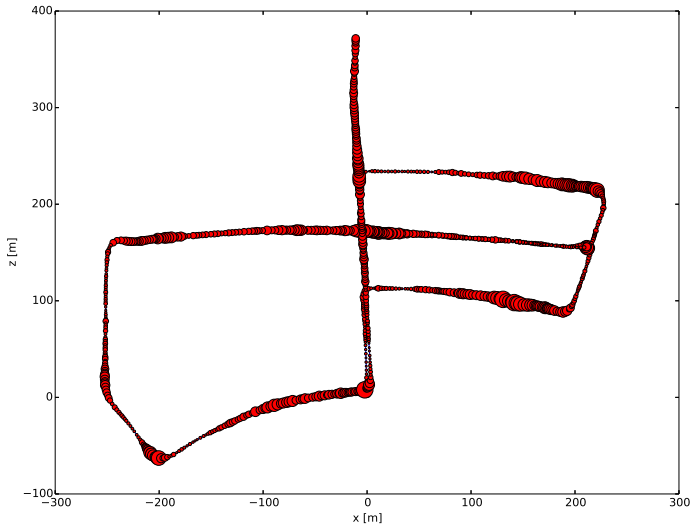
References

1. Agarwal, S., Mierle, K., Others: Ceres solver. http://ceres-solver.org
2. Alismail, H., Browning, B., Lucey, S.: Photometric Bundle Adjustment for Vision-Based SLAM. In: Asian Conference on Computer Vision (ACCV) (2016)
3. Geiger, A., Lenz, P., Urtasun, R.: Are we ready for Autonomous Driving? The KITTI Vision Benchmark Suite. In: Conference on Computer Vision and Pattern Recognition (CVPR) (2012)
4. Mur-Artal, R., Montiel, J.M.M., Tardos, J.D.: ORB-SLAM: a Versatile and Accurate Monocular SLAM System. IEEE Transactions on Robotics (T-RO) **31**(5), 1147–1163 (2015)
5. Pearl, J.: Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA (1988)

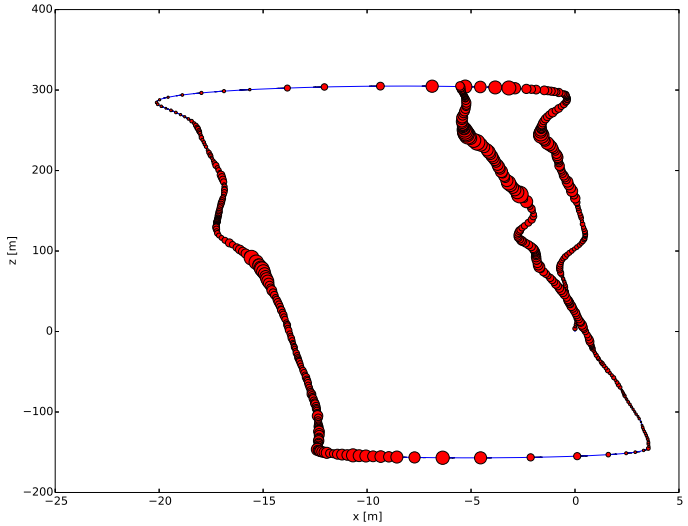
6. Rublee, E., Rabaud, V., Konolige, K., Bradski, G.: ORB: An efficient alternative to SIFT or SURF. In: IEEE International Conference on Computer Vision (ICCV) (2011)



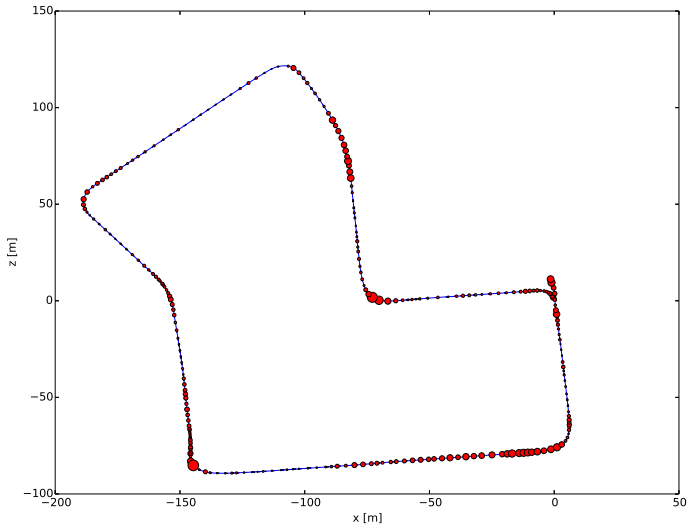
(a) Sequence 00. Maximum number of semantic constraints: 150



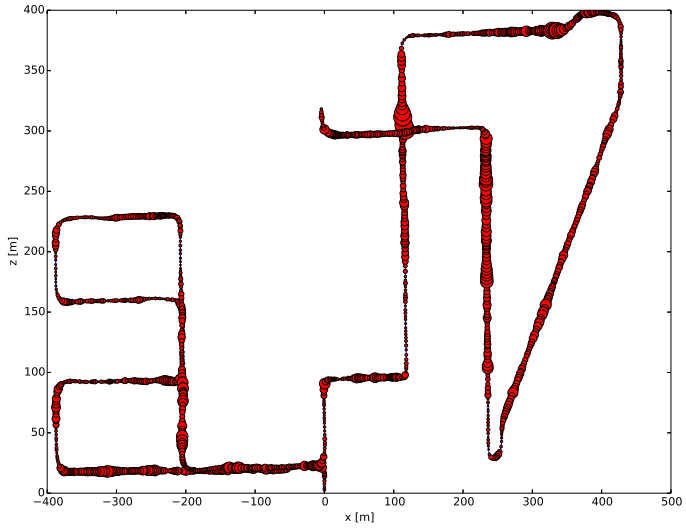
(b) Sequence 05. Maximum number of semantic constraints: 184



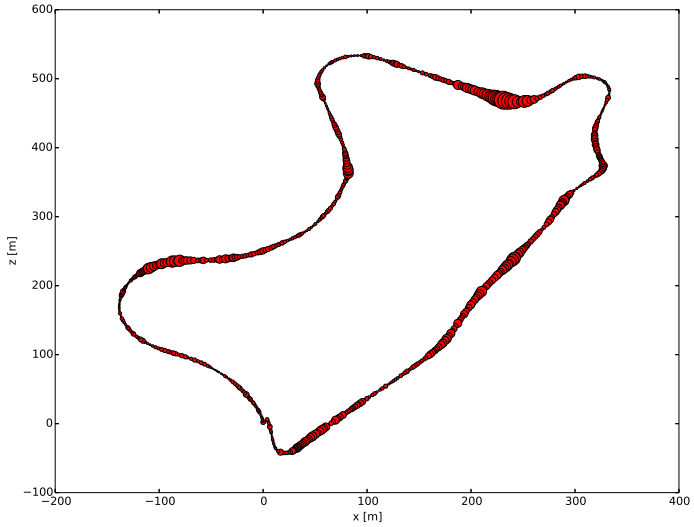
(c) Sequence 06. Maximum number of semantic constraints: 170



(d) Sequence 07. Maximum number of semantic constraints: 110



(e) Sequence 08. Maximum number of semantic constraints: 191



(f) Sequence 09. Maximum number of semantic constraints: 183

Fig. 10: Number of semantic constraints (red circles) overlaid on the camera trajectories for KITTI sequences 00 (a), 05 (b), 06 (c), 07 (d), 08 (e), 09 (f), for the ORB-SLAM2+semantics pipeline. A circle corresponds to an execution of base+semantic windowed optimization, and is located at the position of the latest keyframe during that optimization. Its radius corresponds to the number of optimizable semantic constraints used in this optimization. As can be seen, semantic constraints accumulate when the car is driving straight.