Exploring Semantic Features to Augment Local Feature Localization

Christian Leopoldseder Sahan Paliskara Pedro Roig A. Patricia Sung
ETH Zürich, Switzerland
{cleopolds,spaliskara,pedror,pasung}@ethz.ch

Abstract

Visual localization is a key topic in computer vision and specifically in the development of autonomous vehicles. Many methods for visual localization are based on local features (e.g. SIFT), which struggle with ambiguities, appearance changes, and computational cost. We propose a localization method that identifies likely camera pose estimations using visual semantic features present in query images. These can be used to augment other localization methods by narrowing down their search space and as a consequence resolve ambiguities and reduce computational cost. In the process, we design a novel scoring system to measure similarity between arbitrary sets of bounding boxes. The results show potential for this pipeline to indicate hot-spots which can be used as starting search regions to enhance computational efficiency of other localization pipelines.

1. Introduction

Localization in the context of computer vision is defined as the estimation of camera pose within an environment based on one or more images. In self-localization applications such as autonomous driving, it is advantageous to use visual localization to complement other means of localization, e.g. GPS.

Visual localization is generally approached by building a database from a large set of images that models the environment in which the camera is to be localized. This database is used to determine the camera's pose based on previously unknown images.

There are several methods to build and query such database. One approach commonly used in practice [15, 16] is based on local features like SIFT [10]. In that case, 2D features are extracted from a given query image and matched to the database which comprises a 3D SfM model [15]. As the search space in the 3D world is 6-dimensional and the number of local features that need matching needs to be high for acceptable accuracy, this kind of localization does not scale well [15, 21]. Furthermore, ambiguities arise easily when depending solely on local features [22, 20].

Another approach would be to extract semantic features from the images. In that regard, we identified two types of semantic information for localization: semantic segmentation and semantic object detection. The problem with semantic image segmentation is that image segments are easily misclassified. This is especially a problem when dynamic objects are classified as static objects. Also, the segmentation is not particularly discriminative in monotonous environments when used on its own for localization purposes [18]. On the other hand, a problem with semantic object detection is that the objects being detected might not occur in the environment frequently enough to calculate a useful pose estimation.

Therefore, we aim to reduce the burden of the outlined challenges by combining these approaches. In this paper we propose to support existing approaches for visual localization by utilizing semantic object detection to reduce the search space of another localization method to certain *hot-spots*. Using those hot-spots, computational power can be focused onto smaller subspaces which are most likely to contain the correct camera pose. We are specifically interested in the applicability of this idea for autonomous cars and thus use traffic signs as a stand-in for an arbitrary kind of semantic object. Since cameras are typically the cheapest types of sensors for visual localization, we are interested in a purely image-based approach for this work. In addition, we decided to move the detection of traffic sign in images into the scope of our work since we wanted to get a practical feel for all parts of the pipeline.

Our **contribution** in this paper is to explore the viability of using visual semantic features to augment other localization methods by reducing their search space or by resolving visual ambiguities that would otherwise result in similar probability values for multiple pose estimations. In the process, we are presenting a novel scoring method to measure the similarity between two sets of bounding boxes.

2. Related Work

Localization based on Local Features. Traditional approaches to localization involve using local features like SIFT [10] to find a match between the 2D query images and the 3D points of a Structure-from-Motion (SfM) model. One can then use these matches to estimate the pose of the camera in the environment. If the localization scenario is not too challenging, these methods can lead to pretty accurate localization re-

sults [19]. They can also be further improved in various ways [3, 22, 13, 19] to deal with ambiguities or by incorporating depth information [23, 4] to better cope with perspective distortion. However, these methods are prone to failure when there are strong viewpoint or appearance changes. In these cases there is usually not sufficient overlap between the local features in the query image set and the structural features in the database. The reason being simply the lack of these appearance and viewpoint changes being part of the original data-set [14]. We aim to augment these approaches by filtering the full search space using visual semantic features. This could reduce the number of local features necessary to accurately estimate the camera pose, resolve ambiguities, and alleviate their problems regarding appearance changes.

Localization with Landmarks. As matching approaches based on local features also tend to be fairly computationally expensive, they are not always fit for tasks such as autonomous driving. Such tasks specifically require localization to be done very rapidly, preferably in real-time. Some available solutions to this problem rely on structures that are perspective and condition invariant called *landmarks* such as poles [17, 11]. These approaches detect such landmarks in sensory data and match them to the landmarks in the (previously created) map. By reducing the amount of features needed, one can localize in the environment more efficiently. In practice these approaches unfortunately have the problem that such landmarks are often not abundant in the environment, which makes it not particularly effective for accurate localization on its own [17]. Therefore, we propose visual semantic features as landmarks only to augment other localization methods.

3. Method

Our pipeline is split into map creation and camera pose estimation. We initially create a *landmark map* to represent the environment from a set of monocular mapping images. To create the map, the camera pose has to be known for each image in the mapping set. Note that it is computationally expensive to create this map. However, this is unproblematic since this step only needs to be completed once and it can be done offline. Given a query image, we then calculate *match scores* for a set of pose hypotheses, which can be used to get a pose estimation. This step would be done during the operation of the vehicle and the computational cost is thus significant.

3.1. Map Creation

The landmark map is created over three major steps: *Object Detection*, *Detection Matching*, and *Point Triangulation*. The following sections will explain these steps in detail.

3.1.1 Object Detection

During this step an image is processed to locate and classify visible traffic signs. We call a located traffic sign in an image a *detection*, which consists of a sign type, a 2D center point, and a height and width in pixels. We considered two approaches for this task.

Figure 1: Template matching steps. Bottom left: template being matched. Left: top matches for all different template scales. Middle: point clusters. Right: filtered clusters and resulting bounding boxes

Deep Learning Approach. Initially, a deep learning approach was considered. We experimented with pre-trained models [25, 8] by fine-tuning them to a traffic sign dataset. Specifically, we used the *German Traffic Sign Detection Benchmark* (GTSDB) [6], which contains images with 42 different types of annotated traffic signs. We trained for over 40k epochs, but the results were very unsatisfying as there were problems with too many false positives and wrongly classified traffic signs.

One likely reason for the large amount of misclassifications is that our model has been trained to detect 42 different traffic signs, some of which are very visually distinct, combined with the fact that the training set is not large enough to compensate for that. Some results of the deep learning approach can be found in the appendix.

Template Matching Approach. We eventually decided to use template matching as our object detection method and chose our templates to be the ideal depiction of the traffic signs that occur in the mapping images.

In template matching we take a template image—in our case an image of a traffic sign—and iterate it along all possible locations in the the query image, calculating a similarity value for each position. We use the off-the-shelf implementation of template matching in OpenCV [2]. While undergoing this process, we also scale the template since we do not know how large the signs will be in the image. Since many traffic signs are not rectangular, masks that specify the relevant area of the template are required to achieve acceptable performance. A threshold is applied to the calculated similarity values to obtain top matches for each scale. The resulting set of points is clustered and the clusters are filtered by their variance and the similarity values of the points in that cluster. We then assume that each remaining cluster represents a traffic sign and that the point in a cluster with the highest similarity value is the centroid of the traffic sign. The process is visualized in figure 1. By visual inspection of the outputs of both object detection approaches it was clear that the template matching

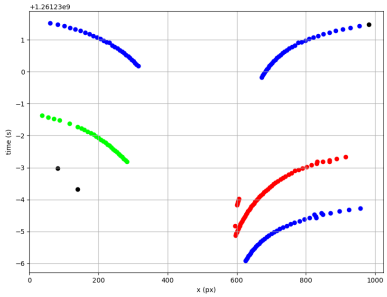

Figure 2: Detection matching between frames. Each point represents a detection in an image taken at a certain point in time. x is the x-coordinate of the detection in the image. Traffic signs of the same type are clustered over time. Colors indicates traffic sign type. Outliers are shown in black.

outperformed the deep learning approach in our case. Keep in mind that we are using traffic signs as a stand-in for arbitrary semantic objects in the environment. Other stationary objects that are present in the environment, e.g. lane markings or traffic lights in the case of road environments, could be used as well.

3.1.2 Detection Matching

The objective of this step is to determine how detections of each image are moving from one frame to another. Initially we wanted to use a nearest neighbor matching method. However, this approach turned out to be more complex to implement than we thought.

As an alternative, we match the detections by placing their (2D) center points in 3D euclidean space, with the third axis being time, and by then clustering those points over time using the Density-Based Spatial Clustering of Applications with Noise (DBSCAN) algorithm. This approach has the added benefit of filtering outliers, which are likely false positive detections from the previous step due to either coincidental or unpredictable structures in the images that activate our detection algorithm. A 2D visualization of the 3D euclidean space combining pixel locations and time can be seen in figure 2.

3.1.3 Point Triangulation

To estimate the position of the traffic signs in 3D space, we use COLMAP [15, 16]. COLMAP is an SfM tool that enables the creation of a 3D map from a sequence of images.

As inputs we use the mapping image set together with their detection sets and the detection matches that were generated previously. Additionally, we provide COLMAP with the camera poses for each mapping image. With this information COLMAP is able to triangulate the center points of the traffic signs in 3D space. We use the triangulated 3D points to construct a set of landmarks. Using the information from which detections each 3D point was estimated we can determine the sign type and—by making the assumption that a traffic sign faces opposite to the driving direction—the orientation of each landmark. As a result, for each landmark we can store a 3D position, a sign type, and a direction vector.

3.2. Camera Pose Estimation

Once the landmark map is generated, we are able to use it to estimate a set of the most likely camera poses given just one query image. We do this by creating a set of *pose hypotheses* around the mapped traffic signs and calculate the *expected view* for each of the potential poses. Afterwards, the expected views are compared to the actual query image and a *match score* for each pose hypothesis is calculated.

3.2.1 Pose Hypotheses

A pose hypothesis consists of a position (x,y-coordinates, z is assumed to be zero) and a yaw angle (pitch- and roll-angles assumed to be zero). We create a set of evenly spaced camera pose hypotheses around the mapped landmarks given a position step size ΔP and an angle step size $\Delta\theta$. That means $\frac{360^\circ}{\Delta\theta}$ pose hypothoses will be considered for each position.

3.2.2 Expected View of Potential Poses

During this step, we calculate the expected view for each of the potential poses in the set of pose hypothoses.

Given a camera pose, the camera parameters and the landmark map, we can calculate the expected set of traffic sign detections. Specifically, we use the pinhole camera model to calculate the position of a traffic sign in the image. For that, we make a few assumptions about whether a landmark would be visible from a particular camera pose. We only include a traffic sign in the expected view if it is (1) in a certain range of the camera pose, (2) in front of the camera, (3) facing the camera within a certain angle threshold, and (4) within the image boundaries. To calculate height and width of the detection we assume fixed sizes of the traffic signs and consider the distance between the traffic sign and the camera.

3.2.3 Score Calculation and Pose Candidates

Given the set of detections of the query image D_Q and the set of expected detections of a specific potential pose D_E (which where obtained as described in the previous section), we can calculate a match score for that pose. For simplicity, assume for now that all detections in the two sets are of the same sign type.

Our detections essentially represent bounding-boxes and there are methods available to measure similarity between bounding boxes, like Intersection over Union (IoU) [9]. The problem with IoU, however, is that it is always zero if there is no overlap between the bounding boxes. This problem has been solved with Generalized IoU (GIoU) [12] or Distance-IoU [24]. These methods are well-suited for $|D_Q| = |D_E| = 1$, but issues

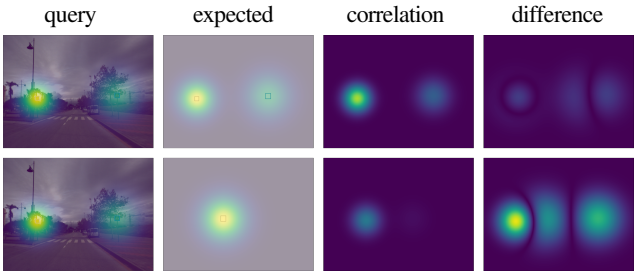

Figure 3: Examples of G_Q (query), G_E (expected), M_C (correlation), and M_D (difference) used for match score calculation. For M_D, the absolute values are visualized. Top row: high match score case. Bottom row: low match score case.

arise when looking at arbitrary sets, as in our case. To apply GIoU or Distance-IoU the bounding boxes in the two sets need to be matched to reduce it to individual pairs, which comes with its own problems. It would also still be restricted to the case $|D_Q| = |D_E|$. One could also merge all bounding boxes in both sets and consider them as a single large bounding box, wich would not require that condition. However, overlapping bounding boxes would then lose their weight, since they should be counted twice, but a merging reduces them to the same weight as the rest of the bounding box area.

As a solution, we propose a novel measure for similarity between two arbitrarily large sets of bounding boxes where we regard the correlation and difference between the sets.

Let H be the height and W be the width of the query images. For a given detection d with center point $c_d = \begin{bmatrix} x_d \\ y_d \end{bmatrix}$, height h_d and width w_d we define a *Gaussian Representation Matrix* $G(d) \in \mathbb{R}^{H \times W}$ which contains values of the probability density function of $\mathcal{N}(\mu, \Sigma)$ with $\mu = c_d$ and covariance matrix $\Sigma = 10 \cdot \begin{bmatrix} w_d^2 & 0 \\ 0 & h_d^2 \end{bmatrix}$.

Then, for all detections d we calculate

$$G_Q = \sum_{d \in D_Q} G(d) \qquad G_E = \sum_{d \in D_E} G(d)$$

We then define the correlation matrix $M_C = G_Q \circ G_E$ (where $\circ$ is the Hadamard product), and the difference matrix $M_D = G_Q - G_E$. Figure 3 shows examples for those matrices.

Let now $\Sigma : \mathbb{R}^{a \times b} \to \mathbb{R}$ be the grand absolute sum of a matrix, namely

$$\Sigma(A) = \sum_{i=1}^{a} \sum_{j=1}^{b} |A_{ij}| \geq 0$$

with which we can define the correlation score $S_C = \Sigma(M_C)$ and the absolute difference score $S_D = \Sigma(M_D)$.

Next, we select tight upper bounds for these two scores, namely

$$S_C \leq \sqrt{\Sigma(G_Q \circ G_Q) \cdot \Sigma(G_E \circ G_E)} = N_C$$

and

$$S_D \leq \Sigma(G_Q) + \Sigma(G_E) = N_D$$

which we can use to normalize S_C and S_D. The upper bound for S_D is obvious. The upper bound for S_C can be derived by combining Hölder's inequality and Jensen's inequality [7].

Finally, we can define a score

$$S = \frac{S_C}{N_C} - \frac{S_D}{N_D}$$

where $S \in [-1, 1]$, which we scale to the interval $[0, 1]$. This way, we have a way of measuring the similarity between two arbitrary sets of detections.

The advantage of this method is that we do not have the limitations of the aforementioned methods (IoU, GIoU, Distance-IoU) for comparing bounding boxes.

Moreover, taking into account both the correlation and the difference in this manner has the advantage that the score is lower if $|D_Q| = 0$ or $|D_E| = 0$, which means the existence of evidence is rewarded.

Since the detection sets contain detections of more than one type in practice, we calculate this score separately for each type. Finally, we calculate the mean of the separate scores, which is what we call the match score for a given pose hypothesis and a given query image.

We can now rank the pose hypotheses by their match score and assume that the more likely the pose corresponds to the query image, the higher its match score. This information can be used to generate a set of hot-spots which can be used as reasonable starting points for a more exact localization method.

4. Results

We chose to test our method using the images of the Malaga Urban Data Set [1]. This dataset has many advantages for us. It covers an urban environment, which means many traffic signs are visible, it consists of high-resolution, pre-rectified images, and it contains multiple images taken from the same location that are subject to appearance and perspective changes.

The dataset is split into 15 different routes or *extracts*. We decided to use extract 07 of the dataset as the set of mapping images, since it consists of a closed-loop route which is densely populated by traffic signs. Since extract 07 is a route overlapping itself, only one full loop was included in the mapping set and the rest of it was used as query images. The route covered by the Malaga Urban Dataset is shown in figure 4. Furthermore, we tested the performance of our pipeline by using the overlapping parts of extracts 08, and 10 as query image sets. This way our pipeline is evaluated with images independent from the mapping image set. As a reference, we also included the mapping image set itself as a query image set.

4.1. Defining the Ground Truth

We defined the ground truth by calculating a cubic spline interpolation of the provided GPS data. We use that ground

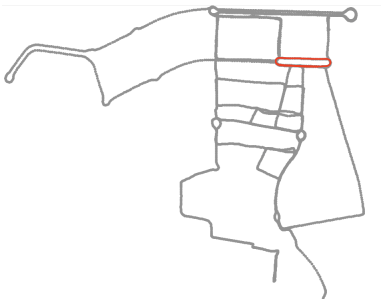

Figure 4: Route of the Malaga Urban Dataset. The section we are using for evaluation is highlighted in red.

truth for evaluating the accuracy of our localization method, but also to calculate the camera poses for the mapping image set.

However, we cannot use the GPS measurements of each extract directly since they all have their own local coordinate frame. The dataset also includes the same measurements in a global coordinate frame, which we can use to transform the measurements into a common local frame. Let L_i with $i \in \{7,8,10\}$ be the local frame of each extract and let G be the global frame. We then calculate the homogeneous transformations $H_{GL_i} \in SE(3)$. The translational part of H_{GL_i} can be calculated from the first GPS measurement of extract i since it always defines the origin of L_i. Then, the rotational part of H_{GL_i} can be calculated by selecting 3 linearly independent points from extract i and solving the resulting system of linear equations.

With the resulting transformations we can transform all GPS measurements into L_7:

$$\begin{bmatrix} p_{L_7} \\ 1 \end{bmatrix} = H_{GL_7}^{-1} H_{GL_i} \cdot \begin{bmatrix} p_{L_i} \\ 1 \end{bmatrix}$$

where p_{L_i} is a point in the respective local frame.

4.2. Mapping result

The COLMAP output from mapping the route of extract 07 is shown in figure 5. Note that there are some wrong landmarks registered, e.g. the ones in the blue circle, where a billboard caused random detections being given by our object detection. An example for a localization result from a query image is visualized as a heatmap in figure 6.

4.3. Evaluation

To asses the performance of the pipeline, we calculate the minimum localization error among the top x (rank) poses with the highest match score for each query image and visualize the median for each query image set. We regard separately the query images where one, two or three traffic signs are detected, since an increase of accuracy is expected when more landmarks are visible in the query image. The results are shown in figure 7.

As one would expect, the detection of **one landmark** in a query image is not sufficient to achieve accurate camera pose estimation.

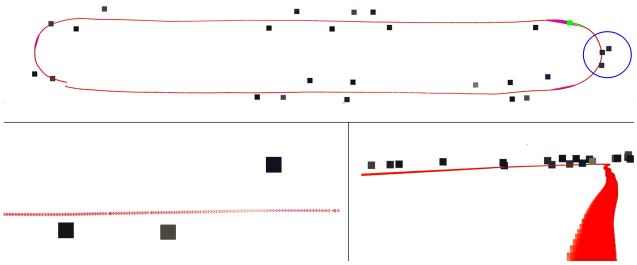

Figure 5: COLMAP output for mapping image set. Each point represents a traffic sign which has been triangulated by COLMAP. The red line indicates images that have been registered to COLMAP. Blue circle contains wrongly registered landmarks. Top: birds-eye view of the entire map. Bottom left: closer view of the map. Bottom right: approximation of the driver's perspective.

For the query images where **two landmarks** are detected, we see a significant improvement in the accuracy of the localization. We can see that in most cases our localization method is able to generate poses within 10 m of the actual pose when looking at the top 10 scores. These top scores could plausibly be used as a starting point for a more exact localization method.

Contrary to our expectations, the performance of our pipeline for query images where **three landmarks** were detected is notably worse than for images containing two and even just one landmark. After inspecting the query images of those cases we have found that query images with three detections contain more often than not false positive detections. Since there are only very few instances where there are actually three traffic signs present in the same image, we can conclude that the cases with false positive detections distort these results drastically.

4.4. Limitations

False positive detections. The biggest limitation for the accuracy of the model is the object detection. The template matching approach leads so several false positive traffic sign detections and their influence can be seen in our evaluation results.

One problem we noticed with our object detection approach is that sometimes traffic sign-alike shapes appear in the environment by chance. When this happens in several consecutive frames during map creation, our pipeline has no mechanism to filter those false positives out after they are matched together (as described in section 3.1.2).

Scalability. Our approach is based on creating an expected view for all pose hypotheses around the detected landmarks in the query image. As we implemented it, the approach is quite computationally expensive, and the computational cost grows with the size of the environment and the number of types of landmarks that should be detected. Additionally, the route we chose for our evaluation covers only a relatively small area and

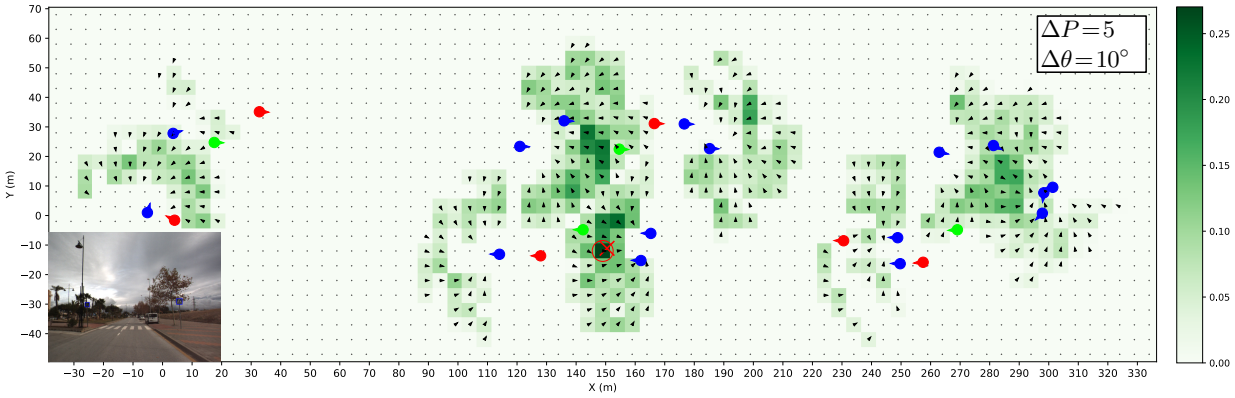

Figure 6: Heatmap visualizing the match score for each potential pose. Bottom left shows the corresponding query image. Colored dots are mapped landmarks (traffic signs). At each position (black dots) there is a fixed number of pose hypotheses with different orientations. Green-intensity indicates highest score of a pose hypothesis at that position. Orientation of highest score is indicated by small black arrows. Note that the color values are not the raw scores, but rescaled values to improve visualization.

only three types of traffic signs were present in the environment. Therefore the approach, as implemented in our project, could not be scaled easily to e.g. an entire city. However, we would like to point out that there is a lot of potential improvement in terms of computational efficiency in our pipeline. The most obvious improvements would be to implement the pipeline in C++ instead of Python and to parallelize the the pipeline by splitting the workload between multiple CPU cores.

5. Conclusions and Outlook

In this paper we explored the use of visual semantic features, specifically traffic signs, to estimate camera poses from individual monocular query images. This approach shows potential to be used to augment other visual localization methods to resolve ambiguities in camera pose estimation.

Further interesting research would be to investigate the localization performance of our pipeline in cases where more landmarks are present by either applying it to datasets that have more traffic signs in their environment, or by incorporating more types of landmarks into the pipeline. If more objects are present in the query image and are detected correctly, the signal-to-noise ratio will be higher and false positives will have less impact on the localization performance.

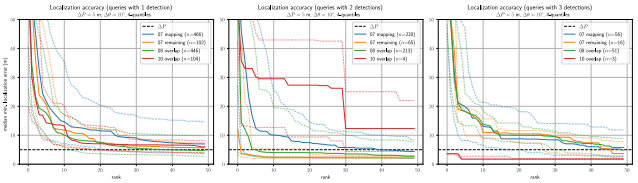

Figure 7: Evaluation results. Dashed lines show quantiles. n is the number of query images with the specified number of detections.

Since we only used single monocular images for querying, another interesting direction would be to use an image sequence for querying by using Monte Carlo localization [5].

Lastly, it would be interesting to see how specific existing localization pipelines perform when they are augmented by our pipeline, in order to directly compare the performances with and without our proposed augmentation.

Our code is published on Github[1].

6. Work distribution

Christian Leopoldseder: ground-truth estimation, refining traffic sign detection and classification using template matching, feature matching, wrapper code for COLMAP, expected view calculation, pipeline/heatmap visualization, code optimization

Sahan Paliskara: traffic sign detection and classification using deep learning, expected view of potential poses, score calculation, and an attempt at using non-linear optimization to improve pose estimations

Pedro Roig A.: traffic sign detection and classification using deep learning, potential poses around landmarks for localization, homogeneous coordinate transformations for evaluation

Patricia Sung: initial implementation of traffic sign detection using template matching, feature matching using nearest neighbor, performance evaluation, traffic sign orientation calculation

[1] https://github.com/leopoldsedev/3d-vision-semantic-localization

References

[1] José-Luis Blanco-Claraco, Francisco-Ángel Moreno-Dueñas, and Javier González-Jiménez. The málaga urban dataset: High-rate stereo and lidar in a realistic urban scenario. *The International Journal of Robotics Research*, 33(2):207–214, 2014. 4

[2] G. Bradski. The OpenCV Library. *Dr. Dobb's Journal of Software Tools*, 2000. 2

[3] F. Camposeco, T. Sattler, A. Cohen, A. Geiger, and M. Pollefeys. Toroidal constraints for two-point localization under high outlier ratios. In *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 6700–6708, 2017. 2

[4] Changchang Wu, B. Clipp, Xiaowei Li, J. Frahm, and M. Pollefeys. 3d model matching with viewpoint-invariant patches (vip). In *2008 IEEE Conference on Computer Vision and Pattern Recognition*, pages 1–8, 2008. 2

[5] Frank Dellaert, Dieter Fox, Wolfram Burgard, and Sebastian Thrun. Monte carlo localization for mobile robots. In *Proceedings of IEEE International Conference on Robotics and Automation (ICRA99)*, May 1999. 6

[6] Sebastian Houben, Johannes Stallkamp, Jan Salmen, Marc Schlipsing, and Christian Igel. Detection of traffic signs in real-world images: The German Traffic Sign Detection Benchmark. In *International Joint Conference on Neural Networks*, number 1288, 2013. 2

[7] Shai Covo (https://math.stackexchange.com/users/2810/shai covo). Is there such an inequality between product and integral for functions. Mathematics Stack Exchange. URL:https://math.stackexchange.com/q/22579 (version: 2011-02-18). 4

[8] Jonathan Huang, Vivek Rathod, Chen Sun, Menglong Zhu, Anoop Korattikara, Alireza Fathi, Ian Fischer, Zbigniew Wojna, Yang Song, Sergio Guadarrama, et al. Speed/accuracy trade-offs for modern convolutional object detectors. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 7310–7311, 2017. 2

[9] Paul Jaccard. The distribution of flora in the alpine zone. *New Phytologist*, 11:37 – 50, 02 1912. 3

[10] David Lowe. Distinctive image features from scale-invariant keypoints. *International Journal of Computer Vision*, 60:91–110, 11 2004. 1

[11] Xiaozhi Qu. *Landmark based localization: detection, matching and update of landmark with uncertainty analysis*. PhD thesis, Université Paris-Est, Marne-la-Vallée, France, 2016. 2

[12] H. Rezatofighi, N. Tsoi, J. Gwak, A. Sadeghian, I. Reid, and S. Savarese. Generalized intersection over union: A metric and a loss for bounding box regression. In *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 658–666, 2019. 3

[13] T. Sattler, B. Leibe, and L. Kobbelt. Efficient & effective prioritized matching for large-scale image-based localization. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 39(9):1744–1756, 2017. 2

[14] Johannes L Schönberger, Marc Pollefeys, Andreas Geiger, and Torsten Sattler. Semantic visual localization. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 6896–6906, 2018. 2

[15] Johannes Lutz Schönberger and Jan-Michael Frahm. Structure-from-motion revisited. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016. 1, 3

[16] Johannes Lutz Schönberger, Enliang Zheng, Marc Pollefeys, and Jan-Michael Frahm. Pixelwise view selection for unstructured multi-view stereo. In *European Conference on Computer Vision (ECCV)*, 2016. 1, 3

[17] Mohsen Sefati, M Daum, B Sondermann, Kai D Kreisköther, and Achim Kampker. Improving vehicle localization using semantic and pole-like landmarks. In *2017 IEEE Intelligent Vehicles Symposium (IV)*, pages 13–19. IEEE, 2017. 2

[18] Erik Stenborg, Carl Toft, and Lars Hammarstrand. Long-term visual localization using semantically segmented images. In *2018 IEEE International Conference on Robotics and Automation (ICRA)*, pages 6484–6490. IEEE, 2018. 1

[19] L. Svärm, O. Enqvist, F. Kahl, and M. Oskarsson. City-scale localization for cameras with known vertical direction. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 39(7):1455–1461, 2017. 2

[20] K. Schindler M. Pollefeys T. Sattler, M. Havlena. Large-scale location recognition and the geometric burstiness problem. In *2017 Conference on Computer Vision and Pattern Recognition (CVPR)*. CVPR, 2017. 1

[21] Christoffer Valgren and Achim Lilienthal. Sift, surf and seasons: Long-term outdoor localization using local features. 09 2007. 1

[22] D. Huttenlocher Y. Li, N. Snavely and P. Fua. Worldwide pose estimation using 3d point clouds. In *2012 European Conference on Computer Vision (ECCV)*. ECCV, 2012. 1, 2

[23] B. Zeisl, K. Köser, and M. Pollefeys. Automatic registration of rgb-d scans via salient directions. In *2013*

IEEE International Conference on Computer Vision, pages 2808–2815, 2013. 2

[24] Zhaohui Zheng, Ping Wang, Wei Liu, Jinze Li, Rongguang Ye, and Dongwei Ren. Distance-iou loss: Faster and better learning for bounding box regression, 2019. 3

[25] Álvaro Arcos-García, Juan A. Álvarez García, and Luis M. Soria-Morillo. Evaluation of deep neural networks for traffic sign detection systems. *Neurocomputing*, 316:332 – 344, 2018. 2