Information and Communication Technologies
Collaborative Project

"V-Charge"

Autonomous Valet Parking and Charging for e-Mobility

Grant Agreement Number 269916



V-CHARGE

Deliverable Number	2.3
Deliverable Name	Experimental verification of novel algorithms from D2.2
WP Number	2
Date of Preparation	26.05.2014
Date of Submission	31.05.2014
Editor/Beneficiary	Sattler/ETH Zürich

Contents

1 Introduction **3**

1.1 Task Description . . . 3

1.2 Overview of the Contributions . . . 3

1.2.1 Localisation using (Global) Semantic Information (ETHZ - CVG) . . . 3

1.2.2 Semantic Mapping Tool-Chain (UOXF.DF) . . . 6

2 Semantic Image-Based Localisation **7**

2.1 Image-Based Localisation . . . 8

2.2 Using Semantic Labels for the Query Images . . . 9

2.2.1 Training Semantic Classifiers . . . 9

2.2.2 Semantic-Based Feature Filtering . . . 9

2.2.3 Experimental Evaluation . . . 11

2.2.4 Discussion . . . 14

2.3 Using Semantically Annotated 3D Models . . . 16

2.3.1 Detecting Blocks of Buildings . . . 17

2.3.2 Experimental Evaluation . . . 17

2.3.3 Discussion . . . 18

2.4 Conclusion . . . 23

3 Semantic Mapping **24**

3.1 Global Overhead Map Image . . . 26

3.2 Active Learning for Label Classification . . . 28

3.3 Consistent Road Graph creation . . . 28

3.4 Semantic Map Structure . . . 30

3.5 Usability of the Semantic Map . . . 30

3.5.1 Transfer of Semantic map labels to different maps . . . 31

3.5.2 Annotation of Semantic Map to aid motion planning 31
3.6 Conclusion 32

4 Conclusion **32**

A Unsupervised Learning of Threshold for Geometric Verification in Visual-Based Loop-Closure **33**

B Joint 3D Scene Reconstruction and Class Segmentation **41**

C Class Specific 3D Object Shape Priors Using Surface Normals **50**

D Driven Learning for Driving: How Introspection Improves Semantic Mapping **59**

E Knowing When We Don't Know: Introspective Classification for Mission-Critical Decision Making **76**

F Reading the Road: Road Marking Classification and Interpretation **85**

1 Introduction

1.1 Task Description

The overall objective of this WP is to develop novel semantic mapping and concepts using visual perception, sporadic GPS inputs and a-priori map information - it is a substrate for WP3. Specific objectives are:

- To further the state of the art in map based localisation in urban settings using low cost sensors
- To create a tool-chain which can be used to create rich, useful and compact representation of the environments applicable to the V-Charge project
- To engage in an ongoing program of evaluation testing and refinement as the quality and utility of maps evolves.

This report covers the research performed by the consortium regarding the verification of novel semantic mapping and semantic localisation concepts based on visual features. In the following, we give a high-level overview over the content of this report and the progress made since Deliverable D2.2, grouped into the topics of localisation and mapping. Sec. 2 then gives a detailed evaluation of the impact of semantic information on image-based localisation. Afterwards, Sec. 3 describes the pipeline of UOXF.DF for the creation of semantic maps.

More details about the following contributions can be found in the appendix.

	Contributions	From	Link
1	Unsupervised Learning of Threshold for Geometric Verification in Visual-Based Loop-Closure, Gim Hee Lee, Marc Pollefeys, ICRA 2014	ETHZ-CVG	A
2	Joint 3D Scene Reconstruction and Class Segmentation, Christian Häne, Christopher Zach, Andrea Cohen, Roland Angst, Marc Pollefeys, CVPR 2013	ETHZ-CVG	B
3	Class Specific 3D Object Shape Priors Using Surface Normals, Christian Häne, Nikolay Savinov, Marc Pollefeys, CVPR 2014	ETHZ-CVG	C
4	Driven Learning for Driving: How Introspection Improves Semantic Mapping, Hugo Grimmett, Rudolph Triebel, Rohan Paul and Ingmar Posner, ISSR 2013	UOXF.DF	D
5	Knowing When We Don't Know: Introspective Classification for Mission-Critical Decision Making, Hugo Grimmett, Rohan Paul, Rudolph Triebel and Ingmar Posner , ICRA 2013	UOXF.DF	E
6	Reading the Road: Road Marking Classification and Interpretation, Bonolo Mathibela, Ingmar Posner, Paul Newman	UOXF.DF	F

1.2 Overview of the Contributions

1.2.1 Localisation using (Global) Semantic Information (ETHZ - CVG)

Image-based localisation is the problem of determining the position and orientation, i.e., the camera pose, from which a given image was taken with respect to a 3D model of the scene. In the V-Charge project,

this problem is for example encountered when the initial position of the car has to be determined in order to begin navigation. The common assumption is that the 3D model is a sparse point cloud reconstructed from a set of database images using Structure-from-Motion or SLAM techniques (as is the case in the V-Charge project). In this case, each point was reconstructed from visual features found in the database image, i.e., we can associate each 3D point with the image descriptors of the features to obtain information about its local appearance. Given a novel image, we can then establish 2D-3D correspondences between features found in this query image and 3D points in the model through the descriptors associated with the features and points. These 2D-3D matches in turn allow us to compute the camera pose. Classical image-based localisation approaches [1–5] rely solely on the visual features and their descriptors to establish the 2D-3D matches and ignore any semantic information present in the map and the query images. However, it has been shown that the majority of features found in a query image do not have a corresponding 3D point [4]. Consequently, classical image-based localisation approaches waste a lot of computational effort on features not likely to yield matches. As part of this report, we thus evaluate in Sec. 2 whether we can use semantic information to detect which regions in the query image show scene elements that are covered by the 3D model. Another problem of image-based localisation approaches that solely rely on the descriptors are locally similar structures found in different, unrelated parts of the scene. Since it is hard to decide which of the resulting ambiguous matches are correct and which are wrong, state-of-the-art approaches usually detect and remove ambiguous matches before attempting pose estimation. This filtering step comes at the expense of localisation effectiveness as fewer query images can be localized since fewer matches are used. In the second part of Sec. 2, we thus investigate how global semantic information that disambiguate structures with locally similar appearance can be used to improve the effectiveness of image-based localisation. This part of the report is based on unpublished work: Our results show that the gain in localisation efficiency that we can obtain through the use of semantic information is only marginal, especially given the additional computational effort necessary to obtain the semantic information. We show that the effectiveness can be improved significantly by considering semantic information in both the query images and the 3D model. At the same time, our approach will fail to work if the scene contains identical structures and we argue that we need more principled approaches that explicitly handle ambiguous structures, which does not seem to be possible using semantic information alone.

Since the second part of Sec. 2 can benefit from using semantic dense 3D models, we have explored the possibility of using convex multi-label segmentation formulations to generate such models, resulting in two publications [6,7]. In the first one we reconstruct and segment the environment jointly to get better geometry and get a consistent segmentation. In this work, we have looked at two intrinsically ill-posed problems, image segmentation and dense 3D modeling from images. Strong regularizers are therefore required to constrain the solutions from being 'too noisy'. Unfortunately, these priors generally yield overly smooth reconstructions and/or segmentations in certain regions whereas they fail in other areas to constrain the solution sufficiently. We argue that image segmentation and dense 3D reconstruction contribute valuable information to each other's task. As a consequence, we propose a rigorous mathematical framework to formulate and solve a joint segmentation and dense reconstruction problem. Image segmentations provide geometric cues about which surface orientations are more likely to appear at a certain location in space whereas a dense 3D reconstruction yields a suitable regularization for the segmentation problem by lifting the labeling from 2D images to 3D space. We show how appearance-based cues and 3D surface orientation priors can be learned from training data and subsequently used for class-specific regularization. Experimental results on several datasets highlight the advantages of our joint formulation. Details on this approach can be found in Appendix B.

In the second paper [7], we have looked at dense 3D reconstruction of real-world objects containing textureless, reflective and specular parts, which is a challenging task. Using general smoothness priors such as surface area regularization can lead to defects in the form of disconnected parts or unwanted indentations. We argue that this problem can be solved by exploiting the object class specific local surface orientations, e.g., a car is always close to horizontal in the roof area. Therefore, we formulate an object class specific shape prior in the form of spatially varying anisotropic smoothness terms. The parameters of the shape prior are extracted

from training data. Our shape prior formulation directly fits into our previously proposed volumetric multi-label reconstruction approach. This allows a segmentation between the object and its supporting ground. In our experimental evaluation we show reconstructions using our trained shape prior on several challenging datasets. Details on this approach can be found in Appendix C.

Closely related to the image-based localisation problem is the problem of visual place recognition: Given a database of images depicting a set of places and a novel query image, determine which database images show the same place as the query image. In the V-Charge project, the visual place recognition problem is encountered during loop-closure for Simultaneous localisation and Mapping (SLAM) when we try to determine whether we have already visited a place while either reconstructing the scene or tracking the motion of the car. Identifying loop-closures, i.e., previously recorded images depicting the current location of the car, allows us to refine the map by computing the geometric constraint that relates the existing database images with the current camera view. In the recent years, many works [8–12] have considered this problems. Most of these approaches rely on vocabulary-trees [13] to select potential loop-closure image pairs purely based on the appearance similarity between the images and do not take the geometric relation between the images into account. As a result, an additional step of geometric verification is required to determine whether a potential loop-closure pair actually depicts the same physical place. Geometric verification refers to the process of checking the consistency of the feature correspondences between the image pairs. This is achieved by counting the number of inliers, i.e., the number of feature matches that are consistent with geometric relation between the two images estimated using RANSAC. The 5-point [14] and 8-point [15] algorithms are commonly used to compute the geometric relation between an image pair from a monocular camera by estimating the epipolar geometry between the images. Similarly, the absolute orientation [16] algorithm is used for stereo cameras, and the 3-point [12] and pose estimation [17] algorithms for multi-camera setups. Usually, an image pair is accepted as a valid loop-closure if the number of inliers exceeds a pre-defined threshold. Obviously, the choice for this threshold is critical to the success of identifying the correct loop-closures. A threshold which is too high results in missing out many correct loop-closure image pairs, resulting in worse 3D maps, while a threshold which is too low results in many false positives, which again negatively impacts the quality of the 3D maps. Usually, the threshold is chosen empirically in the existing works for visual-based loop-closures [8–12]. Unfortunately, the optimal value for the threshold is dataset dependent and it is hard to manually find a good threshold for each scene. In order to solve this problem, we have developed an unsupervised method that learns the threshold for geometric verification directly from the observed geometric verification inlier counts for all the potential loop-closure image pairs [18]: We observed that the inlier counts from all the potential loop-closure image pairs, normalized with the maximum inlier count, can be approximated with a two components Log-Normal mixture distribution. We use the Expectation Maximization (EM) algorithm to learn the hidden variables representing the parameters of the mixture model, and the latent variables representing the state of loop-closure or non loop-closure. The intersection of the Log-Normal mixture distributions is the optimal threshold for geometric verification, i.e., the threshold that gives the minimum false positive and false negative loop-closures. Our algorithm degenerates when there are too few or no loop-closures and we propose the chi-square test to detect this degeneracy. We have experimentally verified our method on several large-scale datasets collected from the multi-camera setup used in the V-Charge project as well as with a stereo camera setup. Since our method essentially uses a sort of global semantic information (in the form of a set of correct loop-closures and false positive loop-closures) to improve visual feature-based place recognition, which plays an essential role both during the construction of the map and during localisation, our work fits well into the scope of this deliverable. We have attached our publication [18] as Appendix A of this report and we plan to integrate this approach into the current V-Charge pipeline.

1.2.2 Semantic Mapping Tool-Chain (UOXF.DF)

Significant progress in autonomous driving during recent years has inspired a view that successful autonomous operation in complex, dynamic environments critically depends on a-priori available semantic maps representing ostensibly permanent aspects of the environment (see, for example, [19]). Indeed, this is considered an important aspect in the context of the V-Charge project, where the car operating in a parking lot could leverage a higher-level understanding of road marks available in a semantic map. For instance, the knowledge that there is an upcoming pedestrian crossing should be taken into account in the vehicle's current motion plan and speed. The labelling of parking spots is crucial for efficient assignment of parking spaces and parking planning. Owing to their safety-critical nature, these high level maps are typically created manually, but hand-labelling is an expensive process which scales linearly with the number of routes to label. Using machine learning classifiers reduces the number of data required for hand-labelling (for the training set) but requires a human to check for errors. Even though classifiers reduce the computational load on humans, traditional classification-based approaches require the use of an extensive repository of ground-truth labelled images for training, which most of the time induces annotations by hand. A prominent approach to tackling such challenges is that of active learning on global map images, where classification results are iteratively refined by asking a human supervisor for ground-truth labels in ambiguous cases, and incorporating the added information into classifier training [20]. This allows for classifiers to achieve a similarly high rate of precision whilst simultaneously reducing the required number of hand-labelled examples, and giving the operator a chance to check and approve the detections. As part of the specific tasks of WP2, in Sec. 3 we describe the tool-chain developed by UOX.DF to build semantic maps automatically whose core relies on active learning for road mark classification (e.g., parking spot, pedestrian crossings). We explain the mechanisms necessary to build a high level representation of a typical navigation environment by combining metric and topological graphs through the general *'road graph'* concept. In this deliverable, we also address the problem of building road graphs from the vehicle localisation as an alternative solution to image-based road detection when there is insufficient visual information about lane marks. We report evaluation results on experiments conducted in the BOSCH parking house located in Stuttgart which fulfils the precision requirements of the V-Charge project in terms of correct label detection. In addition, we evaluate the accuracy of the semantic map for its future integration by transferring the detected labels from the semantic map to the current vehicle frame when the car drives during a different navigation experience at the parking lot.

The approach adopted in the semantic mapping part of this report (cf. Sec. 3) is supported extensively by our recent research published in the following papers, which can be found in Appendix D and Appendix E. In the paper attached as Appendix E we show, in a robotics context, that several of the classification frameworks commonly used in detection of traffic and road labels are unrealistically overconfident in their assessment of class membership. To characterise this attribute, we introduce the notion of an introspective capacity of a classification framework: the ability to associate a confidence which realistically reflects how qualified the classifier is to make a particular class decision with each individual test instance. Our experimental results imply that GPCs possess a more pronounced introspective capacity than other classification frameworks commonly employed in robotics (LogitBoost and Support Vector Machines (SVM)) at the expense of higher computational complexity. We attribute this to their accounting, at test time, for predictive variance over the space of feasible classification models. This is in contrast to other commonly employed classification frameworks which often only consider a one-shot (ML or MAP) solution.

In the paper attached in Appendix D we show that the characterisation of the classification uncertainty harbours significant benefits for active learning as compared to more traditional approaches. In this respect, our contribution is three-fold. Firstly, the notion of introspective classification introduced earlier is shown to be particularly important in the context of active learning, where a reliable estimate of the classification uncertainty is required. We do this by showing an improvement in both classification performance and learning rate over a non-introspective classifier. Secondly, an efficient version of the Gaussian Process Classifier, namely the Informative Vector Machine (IVM) is used, which makes the approach particularly useful for

robotics applications with large amounts of data. We show visual examples of where it is confused and where it is confident, and use it to create the first offline semantic mapping algorithm via active learning. And finally, we present an elegant, information-theoretic solution to the problem of increasing memory requirements by forgetting the least informative data, which further increases the significance of the approach for mobile robotics applications.

A different approach for road marking classification and interpretation is proposed in our paper attached in Appendix F. Unlike previous work which focuses on finding arrows and painted traffic signs on roads, classifying solid and dashed lines, intersections and zebra crossings, in this paper we presented a CRFs based framework for situational awareness based on reading and interpreting road markings. Specifically, we classified road markings into seven distinct classes: single boundary, double boundary, separator, zig-zag, intersection, boxed junction and special lane. Our system achieved a precision of between 74% and 93% and corresponding recall of 70% to 94% across all classes. Overall, our system allows a vehicle to read and interpret the road by understanding the rules of the road encoded in the type of painted road markings visible. We certainly think that this approach can be integrated in the V-Charge semantic mapping pipeline for the detection of lanes.

2 Semantic Image-Based Localisation

Image-based localisation is the problem of determining the position and orientation, commonly referred to as the camera pose, from which a given image was taken. In the following, we assume that we are given a 3D model of the scene obtained using Structure-from-Motion, i.e., a sparse map of the environment reconstructed from a set of database images. In this case, every 3D point in the map has been reconstructed from feature points found in at least two images. Consequently, we can associate each 3D point with the local image descriptors of its corresponding features. Given a novel query image, we can again extract local features in it and use their descriptors to obtain 2D-3D matches between 2D features in the image and 3D points in the model. These 2D-3D matches then allow us to compute the camera pose relative to the 3D point cloud [15]. The main challenge for image-based localisation methods is to efficiently compute the 2D-3D matches as there are usually thousands of image features while the map can be comprised of millions or more 3D points. At the same time, localisation methods need to be effective in terms of localizing query images under a wide range of viewing conditions, i.e., they need to find enough 2D-3D matches to enable camera pose estimation. Usually, the camera pose is estimated using a n-point pose solver inside a RANSAC loop [21]. Since the number of required RANSAC iterations grows exponentially with the percentage of wrong matches, effective image-based localisation techniques need to ensure that most of their matches are correct.

State-of-the-art approaches for efficient and effective image-based localisation all rely on nearest neighbor search in the descriptor space to establish the matches. As shown in [4], most of the features found in a query image have no corresponding 3D scene point, i.e., a significant amount of computation is wasted on features not representing any scene structures. Obviously, features found on movable objects such as pedestrians or cars are very unlikely to yield correct matches as the 3D model contains only the static elements of the scene. Using semantic scene information, we should be able to easily identify such scene elements unlikely to lead to matches and remove features found on them before performing nearest neighbor search. In this part of the report, we will thus investigate the use of such semantics for image-based localisation. Our approach to this problem is inspired by both the semantic maps and the object detection approach described in Deliverable D2.2: We first label regions in the query image to obtain semantic information about the objects depicted in them. We then filter out features falling into regions that show objects that are not contained in the 3D model.

State-of-the-art localisation methods are also based on the implicit assumption that they are able to distinguish between correct and incorrect nearest neighbors by considering only individual descriptors. This

assumption holds for models that contain a few million 3D points and depict landmark scenes or urban areas with visually unique buildings. However, it is often violated for larger scenes and general urban environments [3] since such scenes contain very similar local structures in different, unrelated parts of the models, making it harder to distinguish between correct and incorrect nearest neighbors. This problem is very relevant to the V-Charge project as indoor parking decks are usually comprised of many structures with locally similar appearance. Semantic information about the scene, taking the global structure of the scene into account, should help us to distinguish between locally similar scene elements. In this part of report, we will thus also evaluate whether we can use semantic scene information to disambiguate matches. In order to do so, we use global semantic maps consisting of individual scene elements. In contrast to the semantic maps described in Deliverable D2.2 and the semantic information used for the problem above, our maps do not contain generic labels such as 'building', 'tree', or 'car' but rather groups of spatially close 3D structures.

The rest of this section is structured as follows: We first review the image-based localisation approach that we will use for part of the experimental evaluation. In Sec. 2.2, we explain how to obtain semantic labels for the query images and use them to filter out features that do not have corresponding 3D points. Sec. 2.3 then considers the problem of disambiguating locally similar scene elements.

This part of the report is based on unpublished work. Our results clearly demonstrate that semantic information can be useful for image-based localisation. However, they do not improve the state-of-the-art significantly enough compared to the computational overhead induced by using semantics.

2.1 Image-Based Localisation

For the first part of semantic localisation, filtering out features based on semantic labels before performing localisation, we chose to use the prioritized matching method from [5] since it represents the state-of-the-art in efficient image-based localisation while achieving a localisation effectiveness similar to methods that are orders of magnitude slower [5].

Starting with a fixed quantization of the descriptor space into 100k cells, so-called visual words, obtained through approximate k-means clustering, the method from [5] assigns the descriptors of the 3D points to their closest word in an offline step using approximate nearest neighbor search. For each word, it stores the descriptors assigned to it and references to the corresponding 3D point positions. Given a novel query image, the method assigns the features found in the image to their closest words. For each feature, the method then determines the number of 3D point descriptors assigned to the same word as the feature. Since the method restricts the nearest neighbor search for a query feature to all point descriptors mapped to the same word, the number of point descriptors directly gives the search costs for that feature. The method then sorts the features in ascending order of search costs. Considering the features in the query image in this ordering, the approach finds for every feature f with descriptor $\mathbf{d}_f$ the two nearest neighboring 3D point descriptors $\mathbf{d}_p$ and $\mathbf{d}_q$, i.e., $\|\mathbf{d}_p - \mathbf{d}_f\|_2 < \|\mathbf{d}_q - \mathbf{d}_f\|_2$. A 2D-to-3D match between the feature f and the 3D point p corresponding to the nearest neighbor is established if Lowe's ratio test [22] is passed, i.e., if

$$\|\mathbf{d}_p - \mathbf{d}_f\|_2 < 0.7 \cdot \|\mathbf{d}_q - \mathbf{d}_f\|_2 \ . \tag{1}$$

In order to recover matches lost due to quantization errors, i.e., matches between features and 3D points assigned to different visual words, [5] uses an active search mechanism: Once a new 2D-to-3D correspondence is found, they find a constant number of spatial nearest neighbors around the matching point. These neighboring points are then matched against the features in the query image using a smaller vocabulary which induces fewer quantization errors, where the ratio test is again used to reject wrong or ambiguous correspondences. Once a total of 100 matches is found (comprising of 2D-to-3D and 3D-to-2D correspondences), [5] terminates the descriptor matching process. The method then uses co-visibility relations between the matching 3D points to detect and remove correspondences that passed the ratio test but are likely to be

wrong. The remaining matches are then used to estimate the camera pose using the 6-point DLT algorithm [15] to compute the full projection matrix of the camera inside a RANSAC loop [21]. For further details about the approach, we refer the interested reader to [4, 5].

2.2 Using Semantic Labels for the Query Images

The descriptor matching stage quickly becomes the bottleneck for image-based localisation when considering large-scale scenes. As outlined above, most features found in a query image have no corresponding 3D point in the model and thus introduce unnecessary computational overhead. In this section, we consequently focus on detecting the parts of the query image that depict structures not present in the 3D model, e.g., pedestrians or cars. Any features found in these image regions can safely be discarded, which should result in faster descriptor matching times as fewer features need to be considered. At the same time, removing these features should not impact the localisation effectiveness, i.e., the ability of the localisation method to register many images.

In order to detect image regions not contained in the 3D model, we use semantic classifiers. In the following, we first explain how we train these classifiers. We then describe how the resulting semantic labels are used to decide whether a 2D feature should be considered for correspondence search. Finally, we experimentally evaluate the resulting pipeline.

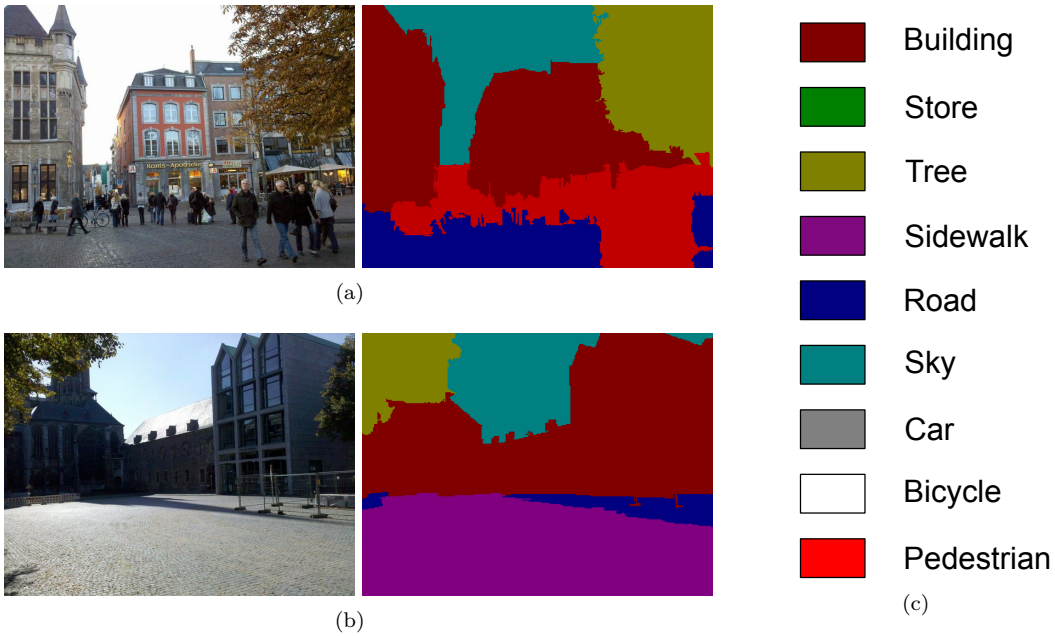
2.2.1 Training Semantic Classifiers

In order to decide whether we want to use a feature for correspondence search, we need to determine which types of scene elements are depicted in the region covered by the feature's descriptor. We obtain this knowledge through semantic labels assigned to different regions of the query image using the *Associate Hierarchical Random Fields* (AHRF) method [23]. AHRF is based on supervised learning, i.e., it uses a set of training images, each with ground truth labels for its image regions, to learn a set of classifiers that predict which image region should be associated with which label. For each training image, [23] first compute two levels of super pixels, where the second level is obtained from the first level, creating a hierarchical representation of the image together with the raw pixel values. [23] then extract different types of features, such as textons [24], local binary patterns [25], and multi-scale dense SIFT descriptors [22], from the training images, which are used to define unary energy potentials for the elements in the hierarchical representation. Besides these unary potentials, which link labels with image features, pairwise potentials are used to try to enforce similar labels for spatially related image regions in the hierarchy. [23] then learn weights for the different types of potentials. Given a new image, its hierarchical decomposition, and the features extracted in it, the learnt weights are then used to compute a label for each element in the hierarchy by finding the maximum a posteriori labeling [23], which can be done in a few seconds per image. Fig. 1 shows examples for images from the Aachen dataset [26] (cf. Sec. 2.2.3) and their computed labels.

For further details about the approach, we refer the interested reader to [23].

2.2.2 Semantic-Based Feature Filtering

Like most image-based localisation approaches [2–5, 26], we use SIFT features [22] due to their robustness against changes in viewpoint, illumination, and against partial occlusions. For each keypoint detected in the scale space pyramid constructed by the algorithm, its SIFT descriptor is defined as the histogram of the gradients found in the descriptor window, a quadrangular region that is centered around the position of the keypoint and aligned with the keypoint's orientation (cf. pink quads in Fig. 2). The descriptor window has



(a)

(b)

(c)

Figure 1: (a), (b) The semantic labels obtained using the method from [23], trained on images from [27], for two of the query image from the Aachen dataset [26]. (c) The semantic labels used.

side length $c \cdot \sigma$, where σ is the scale of the feature and c is some implementation-dependent constant. The histogram is then thresholded and normalized to remove the impact of very strong gradients.

Consider the case that a moving objects covers a large part of the descriptor window. In this case, the feature descriptor will mainly contain gradient information describing the boundary of the object and texture changes within the object. As a result, it will not have a corresponding point descriptor in the 3D model since the Structure-from-Motion process that was used to generate the model reconstructs only static objects. Consequently, we do not need to consider such descriptors for image-based localisation as they will only add computational costs without yielding valid 2D-3D matches that can be used for camera pose estimation. In order to detect and remove those features before applying the localisation method from [5], we use the semantic labels obtained using [23]: Let $\mathcal{L}$ be the set of labels and let $\mathcal{L}_S \subset \mathcal{L}$ and $\mathcal{L}_M \subset \mathcal{L}$ be the labels associated with static and movable elements, respectively. Since the rectangular descriptor window needs to be aligned to the feature's orientation and since gradients found at positions away from the center of the descriptor window are down-weighted by the SIFT algorithm [22], we approximate the regions covered by the descriptor window by a circle of radius $\frac{c}{2} \cdot \sigma$, which simplifies the implementation due to the rotational invariance of circles (cf. green circles in Fig. 2). We then build a histogram over the semantic labels associated with the pixels falling inside the circle, counting the number c_l of occurrences for each label $l \in \mathcal{L}$. In order to decide whether to use a descriptor for localisation or not, we consider the ratio

$$\frac{\sum_{l \in \mathcal{L}_S} c_l}{\sum_{l \in \mathcal{L}_S} c_l + \sum_{l \in \mathcal{L}_M} c_l} \tag{2}$$

of pixels labeled with one of the labels from $\mathcal{L}_S$ to the total number of pixels falling into the circle. If the ratio is greater or equal to a given threshold $\tau \in [0, 1]$, i.e., if

$$\frac{\sum_{l \in \mathcal{L}_S} c_l}{\sum_{l \in \mathcal{L}_S} c_l + \sum_{l \in \mathcal{L}_M} c_l} \geq \tau \ , \tag{3}$$

we keep the corresponding feature. Otherwise, we remove it before attempting image-based localisation.

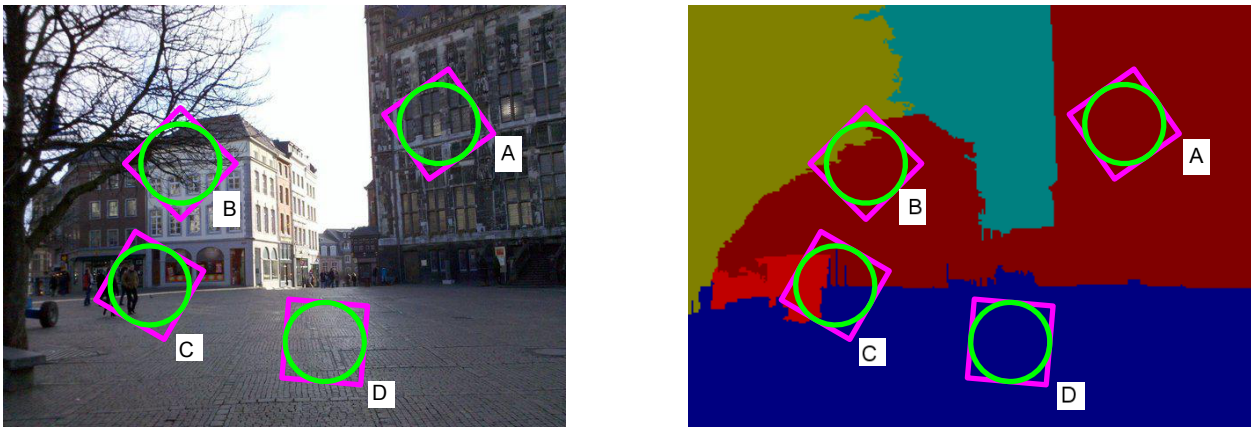


Figure 2: Illustration of our feature filtering step, using the coloring scheme from Fig. 1(c) for the semantic labels. For each of the four features, the pink rectangle denotes the image region used to compute the SIFT descriptor. We use the region defined by the green circle to determine which semantic structures are covered by the descriptor. Descriptor **A** is found in a part of the image marked as building (dark red) and should thus be used for correspondence search. While the green circle of descriptor **B** also contains a small region labeled as a tree (yellow), it should still be used for localisation due to the robustness of SIFT descriptors to partial occlusions. In contrast, the green circles of descriptors **C** and **D** mostly contain regions not present in the model and should thus not be considered for image-based localisation.

Notice that our definition of a semantic-based feature filtering step ignores all spatial relationships between pixels with the same labels in order to better deal with wrong labels.

2.2.3 Experimental Evaluation

We evaluate using semantic-based feature filtering for image-based localisation on two challenging datasets: The Aachen model [26] and an extended version of the Vienna dataset [1] that is commonly used to evaluate localisation approaches [1–5].

The Aachen dataset depicts part of the historic inner city of Aachen, Germany and was reconstructed from 3047 database images taken with a few consumer-level cameras using a public Structure-from-Motion software [28, 29]. The model contains slightly more than 1.5 million 3D points with a total of more than 7.2 million associated image descriptors [26]. The 369 query images were taken over a timespan of roughly two years with a mobile phone and thus capture seasonal changes, different lighting conditions, as well as the typical defects of photos taken with mobile phones, e.g., blur or over- and underexposure [26].

The extended Vienna dataset [1] was reconstructed from 2267 database images taken with a single, calibrated camera and depicts six landmark scenes[1] in the city of Vienna, Austria. It consists of 2.15 million 3D points with a total of more than 9 million associated image descriptors. We use 650 query images obtained by downloading photos from Panoramio that depict parts of the scenes covered in the model. As for the Aachen dataset, all query images have a maximum resolution of 1600×1600 pixels.

In order to train the semantic classifiers, we trained the method from [23] (cf. Sec. 2.2.1) on 1774 photos from a publicly available dataset containing Street View images and ground truth labels [27]. The dataset offers labels for buildings, stores, trees, sidewalks, roads, sky, cars, bicycles, and pedestrians (cf. Fig. 1(c)).

[1]The regular Vienna dataset commonly used in the literature only consists of three landmarks.

We purposely trained the classifiers on an unrelated dataset as it is impractical to re-train them for each new dataset, which is due to the considerable manual effort involved in obtaining labeled training images for a new dataset. Thus it is important to evaluate whether classifiers trained on an unrelated dataset generalize well.

Given a new query image, we first compute the semantic labeling using the trained classifiers. We then use the feature filtering approach described in Sec. 2.2.2 to remove features found in the query image that correspond to categories not present in the 3D model. We consider features found on sidewalks, roads, cars, bicycles, and pedestrians as unstable[2], i.e., we set $\mathcal{L}_M = \{$'sidewalk', 'road', 'car', 'bicycle', 'pedestrian'$\}$ and $\mathcal{L}_S = \{$'building', 'store', 'tree', 'sky'$\}$. After performing semantic filtering, we use an implementation of [5] that was optimized for cache consistency (and thus achieves faster matching times when performing nearest neighbor search inside a visual word) to perform image-based localisation for all remaining features. After matching, the camera pose is estimated using the 6-point DLT algorithm [15] to compute the full projection matrix of the uncalibrated query images inside a SPRT-RANSAC loop [30] that applies local optimization [31]. We consider a query image as successfully localized if the best camera pose found by RANSAC has at least 12 inliers [2–5]. We use the publicly available SiftGPU implementation [32] to extract SIFT features in the query images.

During the experiments, we measure both the localisation effectiveness, i.e., the number of query images that could be localized, and the localisation efficiency, i.e., the average time required to localize a query image. The latter measure includes the time to perform semantic filtering, visual word assignments, correspondence search, and camera pose estimation, but does not include neither the time required to extract the SIFT feature nor the time required to obtain the semantic labeling of the query image. Due to the random nature of RANSAC, we repeated every experiment 10 times and report the total number of query images that can be localized at least once. For each query image, we average the time required for localisation using all repetitions in which the image was localized.

We would like to choose the threshold parameter τ for semantic filtering (cf. Eqn. 3) such that we obtain a good compromise between localisation effectiveness and efficiency. Since the run-time required to filter out features does not depend on the value of τ, the gain in efficiency for the localisation method is directly related to the percentage of features remaining in the query image after applying the filtering step. Fig. 3 details the effect of the threshold on both the number of images that can be localized and the percentage of features remaining after the filtering procedure. As can be seen in Fig. 3(a),(c), the number of query images that can be localized remains rather constant for both datasets, even though there is a slight drop in localisation effectiveness for larger values of τ. Furthermore, it can be observed that there is only a small difference between the number of images that can be localized without applying the semantic filtering and with the filtering step: For the Aachen dataset, this difference is at most 5 images (about 1.4% of all query images) while the maximal difference is 10 images for the Vienna dataset (about 1.5% of the query images). This shows that the semantic labels provided by [23] are accurate enough to prevent loosing too many important 2D-3D matches due to mislabeling, even though the classifiers were trained on an unrelated dataset. While the labels are precise enough to maintain the localisation effectiveness of the approach from [5], Fig. 3(b),(d) shows that between 60% and 90% of the features pass the semantic filtering step for the majority of the localized query images for both datasets. As can be expected, increasing the threshold τ increases the fraction of features rejected by the filter as it requires the corresponding descriptor windows to contain more and more regions depicting parts of the scene that are actually contained in the model. Based on the results from Fig. 3, we choose $\tau = 0.8$ since larger values result in a (small) drop in localisation effectiveness while smaller values filter out fewer features, which should translate to a smaller gain in efficiency. For all the following experiments, we will use $\tau = 0.8$.

[2]Sidewalks and roads were selected because we experienced that the descriptors of features found on them are extremely viewpoint dependent, making it unlikely to obtain a correct match for any such features. At the same time, we found that the descriptors of features found at the boundary of buildings and sky were very robust to changes in viewpoint since the part of the descriptor modeling the sky remained very similar even for larger differences in viewpoint.

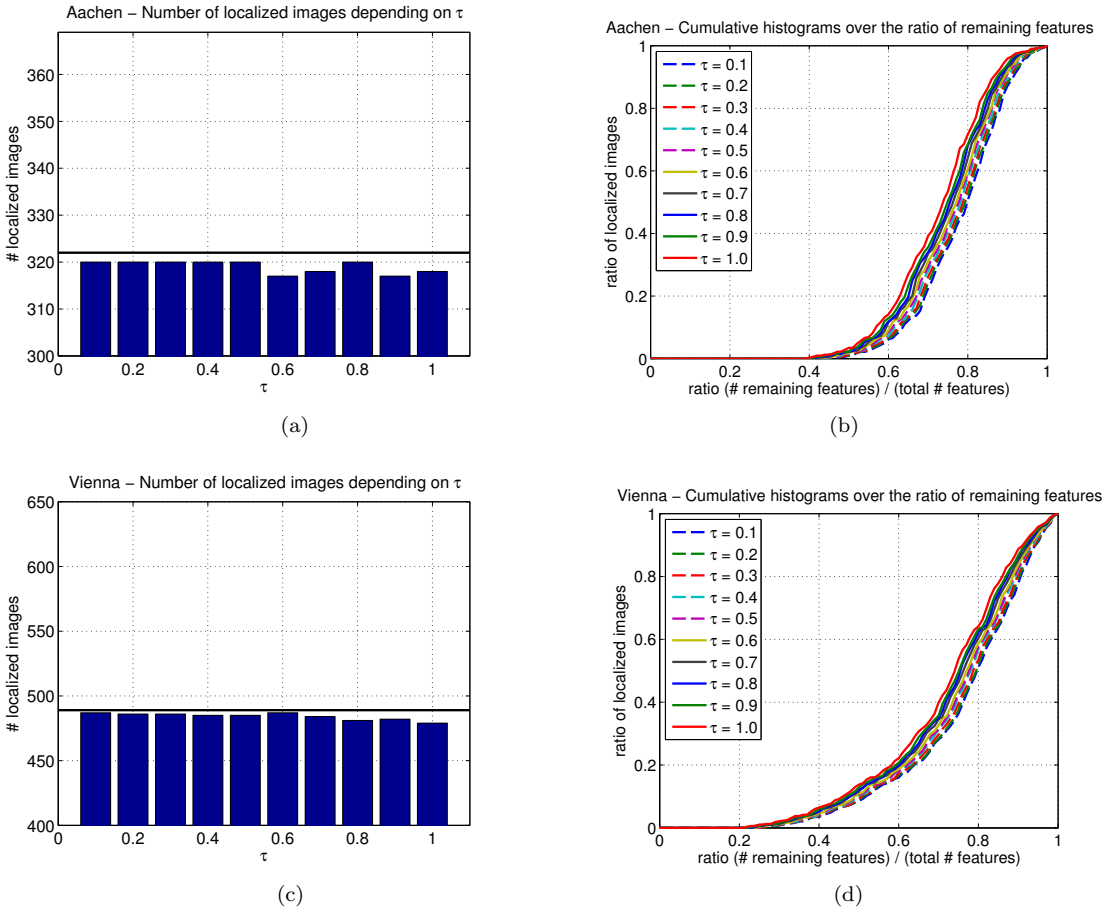


Figure 3: (a), (c) The number of query images that can be localized depending on the semantic ratio test threshold τ for the Aachen and Vienna datasets, respectively. The black line denotes the number of images that can be localized without semantic feature filtering. (b), (d) Cumulative histograms over the ratio of query features remaining after performing feature filtering with different threshold values.

Our motivation for filtering features before performing image-based localisation was based on the reasonable assumption that removing features that will not lead to valid 2D-3D matches will decrease the localisation times. In the following, we thus investigate the impact of semantic filtering on the run-time of image-based localisation in more detail. Fig. 4 compares the cumulative distribution over the run-times for the query images that can be localized with and without using semantic filtering. The figure shows the cumulative distributions when considering different parts of the localisation pipeline from [5], namely computing the nearest visual word for all features passing the semantic test (blue curve), additionally performing 2D-3D matching (red curve), computing the visual word assignments, 2D-3D matches, and the camera pose (green curve), as well as additionally performing semantic filtering (black curve). As expected, using fewer features leads to faster visual word assignment times as fewer assignments need to be computed. Interestingly, the 2D-3D matching stage is comparable or slightly faster when not using semantic filtering, even though more image features are used. The reason for this behavior is that the localisation approach from [5] combines both 2D-to-3D and 3D-to-2D matching and chooses the matching direction depending on whether evaluating the next 2D feature or the next 3D point is cheaper. As a consequence of using fewer features in the query image due to semantic filtering, 3D-to-2D matching becomes cheaper and the algorithm starts preferring 3D-to-2D search over the other matching direction more often. This is illustrated in Fig. 5, which compares the fraction of descriptor comparisons caused by 3D-to-2D matching among all descriptor comparisons when using semantic filtering with the fraction observed when using all features. The fact that 2D-3D matching is faster when using all features indicates that many of the 3D-to-2D matching steps do not lead to valid correspondences. Notice that the localisation approach from [5] already only uses 3D points for 3D-to-2D matching that are likely to be visible in the query image. Thus, it seems very unlikely that improving the semantic labels, e.g., by using more fine-grained categories, will allow us to avoid these unnecessary matching steps. While overall the times required to compute the visual word assignments, 2D-3D matches, and the camera pose are comparable with and without using semantics, not using filtering is overall faster due to the overhead required for semantic filtering (cf. black curves in Fig. 4). However, our current implementation is not very optimized and could be improved by hierarchically clustering areas in the label image with similar labels in order to accelerate the computation of the semantic ratios faster for features that cover larger image regions, i.e., features detected on a larger scale. Yet, even when semantic filtering would induce no costs, our results indicate that using semantic labeling for localisation does not offer a large enough improvement to warrant the additional effort required to obtain the labels, especially when considering that computing the semantic labels for a query image takes multiple seconds[3] [23].

2.2.4 Discussion

The results presented in Sec. 2.2.3 are interesting as they show that using higher level scene understanding, in the form of semantic labels, does not automatically improve localisation efficiency. This contradicts our initial understanding as we assumed that removing features that cannot lead to valid 2D-3D matches should improve the localisation times without modifying the localisation algorithm. As shown in Fig. 3, using semantic filtering does remove some features, so in theory we should be able to harness the resulting potential, e.g., by optimizing the trade-off between 2D-to-3D and 3D-to-2D matching. However, the potential speed-up that can be gained is not very large as the majority of features passes the semantic filtering step[4].

The approach discussed above uses semantic labels only for features found in the query image but not for the 3D points in the model. By associating labels also with the points in the model (which could be done

[3]The computational effort required to determine the labels is independent of the size of the 3D model. As we consider larger models, the number of required descriptor comparisons grows linearly in the number of 3D points. Thus, we can expect that correspondence search will become the dominant cost factor at some point. Yet, based on our results it is questionable if using semantic filtering will improve the run-times as the majority of features found in a query image passes the filtering stage

[4]In contrast to the query image used in our experiments, we can expect to find more movable objects in the images taken by the V-Charge car. At the same time, the cameras used by the car cover a much wider field of view, i.e., the fraction of features found on movable objects should not vary much compared to our experiments.

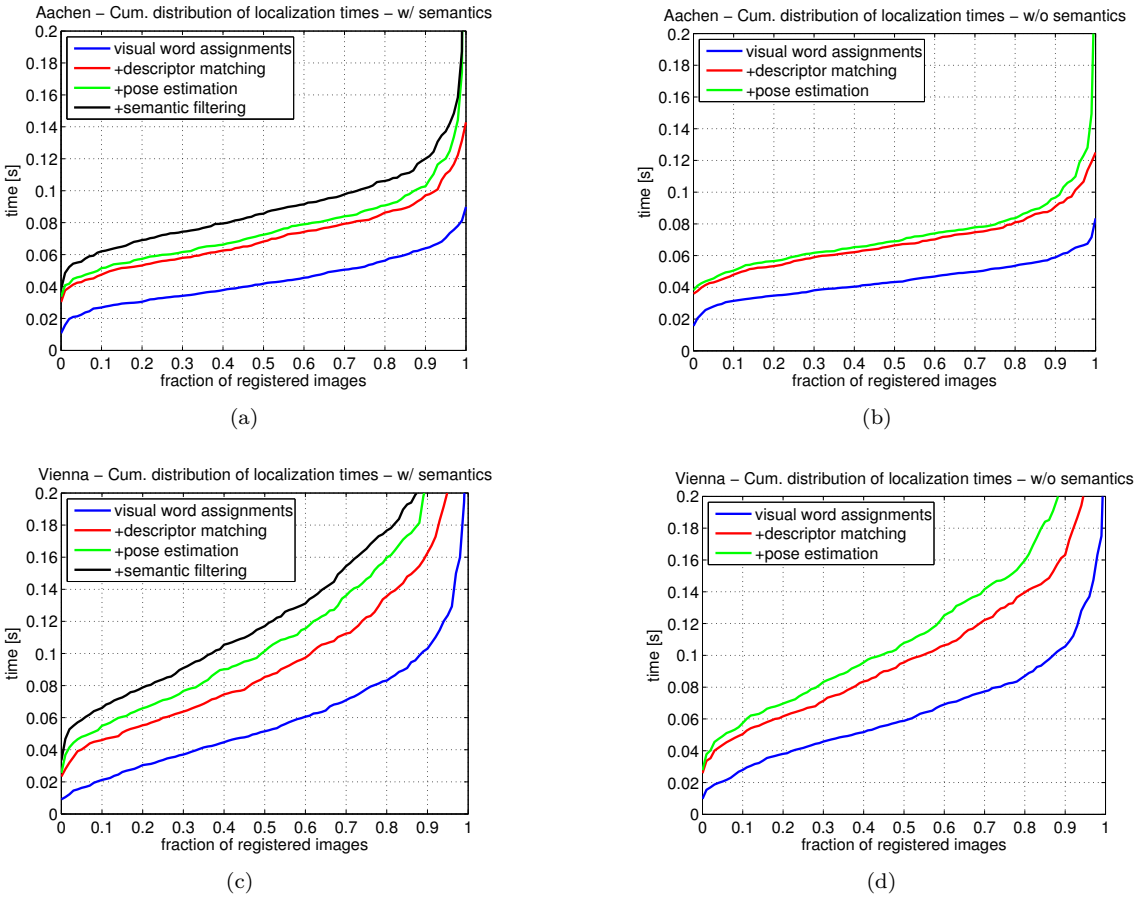
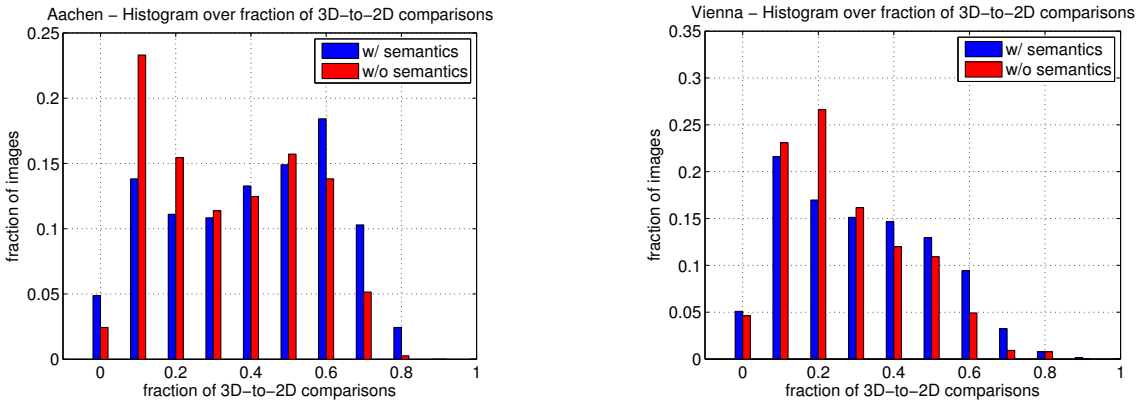


Figure 4: (a), (b) The cumulative distribution of run-times for the query images from the Aachen dataset that can be localized with and without semantic filtering, respectively. The blue curve denotes the cumulative distribution over the time required to assign each query feature to its closest visual word. The red curve denotes the run-time distribution for computing the visual word assignments and computing the 2D-3D matches. Similarly, the green and black curves are obtained by also considering the time required for pose estimation and semantic filtering. (c), (d) The same plots for the Vienna dataset.



(a) (b)

Figure 5: Histograms over the fraction of 3D-to-2D descriptor comparisons among all descriptor comparisons with and without using semantic feature filtering for (a) Aachen and (b). Using semantic labels to filter out features decreases the costs for 3D-to-2D matching. As a result, more 3D point-to-2D feature descriptor comparisons are performed when using semantic filtering.

in an offline step) would allow us to avoid comparing a feature and a 3D point that belong to different object categories, e.g., building and tree, thus reducing the number of required descriptor comparisons and decreasing the overall run-times. However, descriptors extracted from different categories are usually rather different from each other, i.e., we could probably achieve the same result by using a fine grained clustering of the descriptor space, at least when using only broad categories such as 'building', 'tree', and 'road'. Semantic labels might be more powerful when using more fine grained categories. However, the scenes encountered in the V-Charge project, indoor parking decks and outdoor parking lots, usually contain only a limited amount of different types of objects.

Based on our results and the discussion above, we conclude that the use of semantic labels does not improve localisation efficiency significantly enough to make further research in this direction worthwhile.

2.3 Using Semantically Annotated 3D Models

As we consider 3D models depicting larger and larger scenes, the distance between the descriptor $\mathbf{d}_f$ of a query feature f and its second nearest neighboring point descriptor $\mathbf{d}_q$ decreases since more and more locally similar structures are contained in unrelated parts of the scene [3]. Consequently, Lowe's ratio test employed to reject wrong and ambiguous matches (cf. Eqn. 1) will reject more and more correct matches [3,5]. In order to distinguish between locally similar structures found in different parts of the model, we thus need to take some sort of global context into account. Notice that using generic semantic labels, i.e., labels distinguishing between buildings and trees, windows and walls, is not sufficient as they do not distinguish between locally similar structures. At the same time, fine-grained semantic labels, e.g., differentiating between different building elements such as windows, doors, and balconies, or between different types of objects in the scene such as traffic signs, also cannot be used to solve this problem since the ambiguities arise from similar scene structures such as nearly identical windows. Notice that the problems caused by Lowe's ratio test are also prevalent for scenes containing many similar and repetitive structures, such as the indoor parking decks considered in the V-Charge project. Preventing ambiguous matches is thus a very important problem for the project.

During the reconstruction process, Structure-from-Motion and SLAM approaches start by matching features between images in order to establish 2D-2D matches, which in turn are used to recover the relative motion between the cameras. As in the localisation method discussed in Sec. 2.1, Lowe's ratio test is used to reject wrong matches. Thus, it is reasonable to assume that most ambiguous 2D-3D correspondences come from similar structures found in different parts of the scene instead from similarities occurring in the same part of the scene, e.g., the same building facade. We can thus easily prevent the ratio test from rejecting more and more correct matches for larger scenes or models containing many repetitive scene structures by matching the features found in a query image only against the points belonging to the structures depicted in the image instead of the whole model. In Sec. 2.3.1, we thus describe how to segment the 3D model into individual structures. Given information about the scene parts depicted in the query image, we show in Sec. 2.3.2 that matching only against the visible structures enables us to localize more images than standard localisation approaches matching against the whole model. In this report, we aim at detecting scene structures on a building block level. However, the proposed segmentation method should generalize well as long as the desired structures are spatially separated.

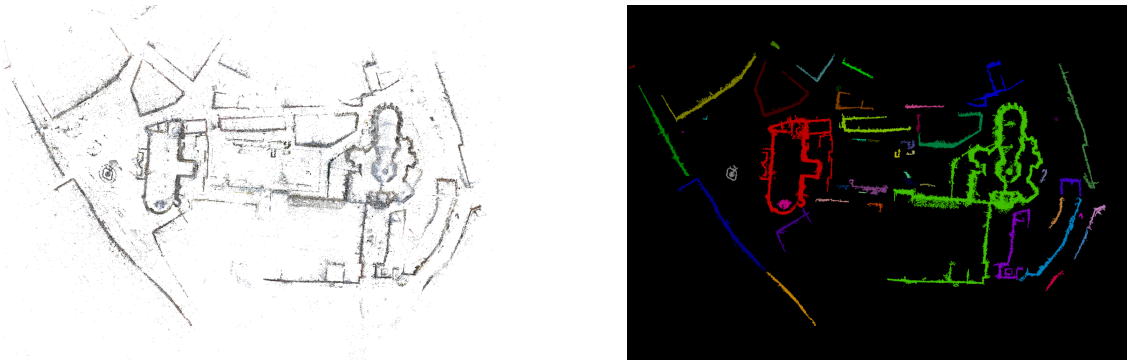
2.3.1 Detecting Blocks of Buildings

Ideally, we would like to match the image features against the individual buildings. However, it is hard to segment out an individual building when it directly borders on the next buildings. Instead, we aim at identifying blocks of adjoint structures. Since our models are geo-registered, they are aligned such that the buildings are orthogonal to the known ground plane. We thus project all 3D points in our Structure-from-Motion models into the ground plane. Grouping these points into connected components then yields exactly the blocks of buildings we are interested in.

Since the 3D models obtained through Structure-from-Motion are sparse point clouds, we need to quantize our ground plane in order to be able to detect connected components. We use a regular grid of squares of side length 0.5m to obtain this quantization and count the number of points projected into each cell. Due to errors and inaccuracies in the reconstruction process, Structure-from-Motion point clouds often contain outlier points. To avoid connecting separate structures through such outliers, we only consider cells containing at least 10 points. Similarly, we are only interested in buildings and would like to ignore smaller structures. Thus, we also ignore all cells whose associated 3D points have a maximal height difference of less than 1m. We use the remaining cells to find connected components, where two cells are considered connected if one cell is among the eight neighboring cells of the other. Finally, we only keep connected components if they consisting of at least 200 points (over all cells) as we are only interested in larger structures such as buildings and statues. Fig. 6 shows the result of this segmentation procedure for the Aachen model, where each detected building block is depicted using a unique color.

2.3.2 Experimental Evaluation

In order to show that matching against the blocks of buildings obtained using the segmentation method described above enables us to localize more images and handle the problems caused by Lowe's ratio test, we consider the 47 query images from the Aachen dataset that could not be localized using the method from [5] without semantic filtering. Using the building labels shown in Fig. 6(b), we manually label the regions in the query images according to the blocks of buildings that are visible in the images. For each feature found in such a region, we exhaustively compare its descriptor against the descriptors of the 3D points belonging to the corresponding building block. After determining the two distinct 3D points belonging to the two nearest neighboring descriptors, we again apply Lowe's ratio test. The remaining 2D-3D correspondences are again used for RANSAC-based pose estimation. We consider a query image to be successfully localized if the pose found by RANSAC has at least 12 inliers.



(a) (b)

Figure 6: (a) Top-down view of the Aachen model. (b) Building labeling obtained by projecting the 3D points of the Aachen model into the ground plane and detecting connected components. Each of the 48 detected buildings is marked in an individual color.

Out of the 47 query images that could not be localized using [5] without semantic filtering, the method described above was able to localize the 22 images depicted in Fig. 7 and Fig. 8, i.e., it was able to register 46.8% of the images that could not be localized before. This clearly shows that segmenting the 3D model into building blocks and matching against individual parts of the point cloud instead of all points improves localisation effectiveness and allows us to better handle 2D-3D matches that were previously rejected by as too ambiguous by Lowe's ratio. Looking at the query images that still could not be localized, shown in Fig. 9 and Fig. 10, we notice that they were either taken under extreme lighting conditions (all database images were taken during the day), have only little overlap with the scene parts covered by the 3D model, or show the buildings from viewpoints that differ strongly from the viewpoints from which the database images were taken. In all of these cases, the missing invariance of the SIFT detector and descriptor to viewpoint and illumination changes makes it hard to find enough correct matches to facilitate pose estimation, i.e., Lowe's ratio test did not cause the localisation method to fail to register these images. We expect that using descriptors and detectors that can handle the above mentioned image defects would allow us to register at least some of these query images.

2.3.3 Discussion

The results from Sec. 2.3.2 clearly demonstrate that current state-of-the-art localisation methods do not harness the full potential of local descriptors as they have problems handling ambiguities caused by locally similar scene structures found in unrelated parts of the scene. Using semantic information in the 3D map allows us to properly handle this ambiguities under the assumption that we have a classifier that can associate regions in the query image with their corresponding parts in the model.

In order to demonstrate that semantic information can improve the effectiveness of image-based localisation methods and to avoid possible misclassifications, we manually labeled the query images. However, we would need to obtain the labels automatically for our approach to be of practical use, i.e., we would need to train a classifier that distinguishes between different blocks of buildings. Unlike the approach discussed in Sec. 2.2, where we were able to train a classifier on a single dataset and re-use it for all other datasets, we would need to train separate classifiers for each new model or when we enlarge the reconstruction. This requires us to obtain a ground truth labeling for a set of training images from which the classifier can be learned. Obviously, manually creating this ground truth labeling is infeasible as manually labeling the 47 query images used in Sec. 2.3.2 already took nearly a day. We could project the 3D points from the segmented point cloud into



Figure 7: The first 12 of the 22 query images from the Aachen dataset that can be localized when matching against the individual building blocks depicted in the images but not using the method from [5]. The semantic segmentation of the query images were obtained by hand. The colors in the semantic images correspond to those from Fig. 6, while regions not depicting any building block are colored in black.



Figure 8: The last 10 of the 22 query images from the Aachen dataset that can be localized when matching against the individual building blocks depicted in the images but not using the method from [5]. The semantic segmentation of the query images were obtained by hand. The colors in the semantic images correspond to those from Fig. 6, while regions not depicting any building block are colored in black.

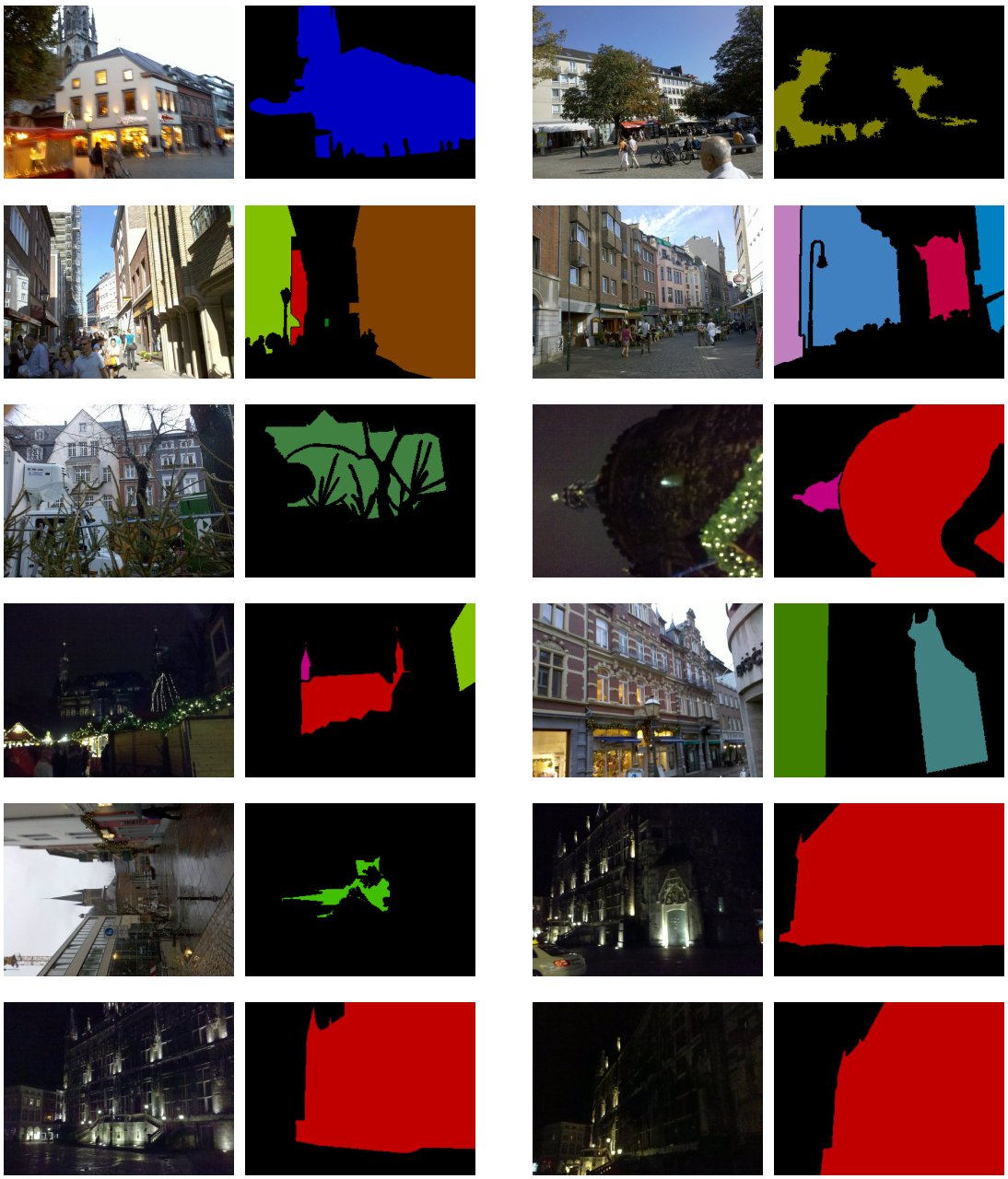


Figure 9: The first 12 of the 25 query images from the Aachen dataset that could not be localized with neither the method from [5] nor when matching against the individual building blocks.

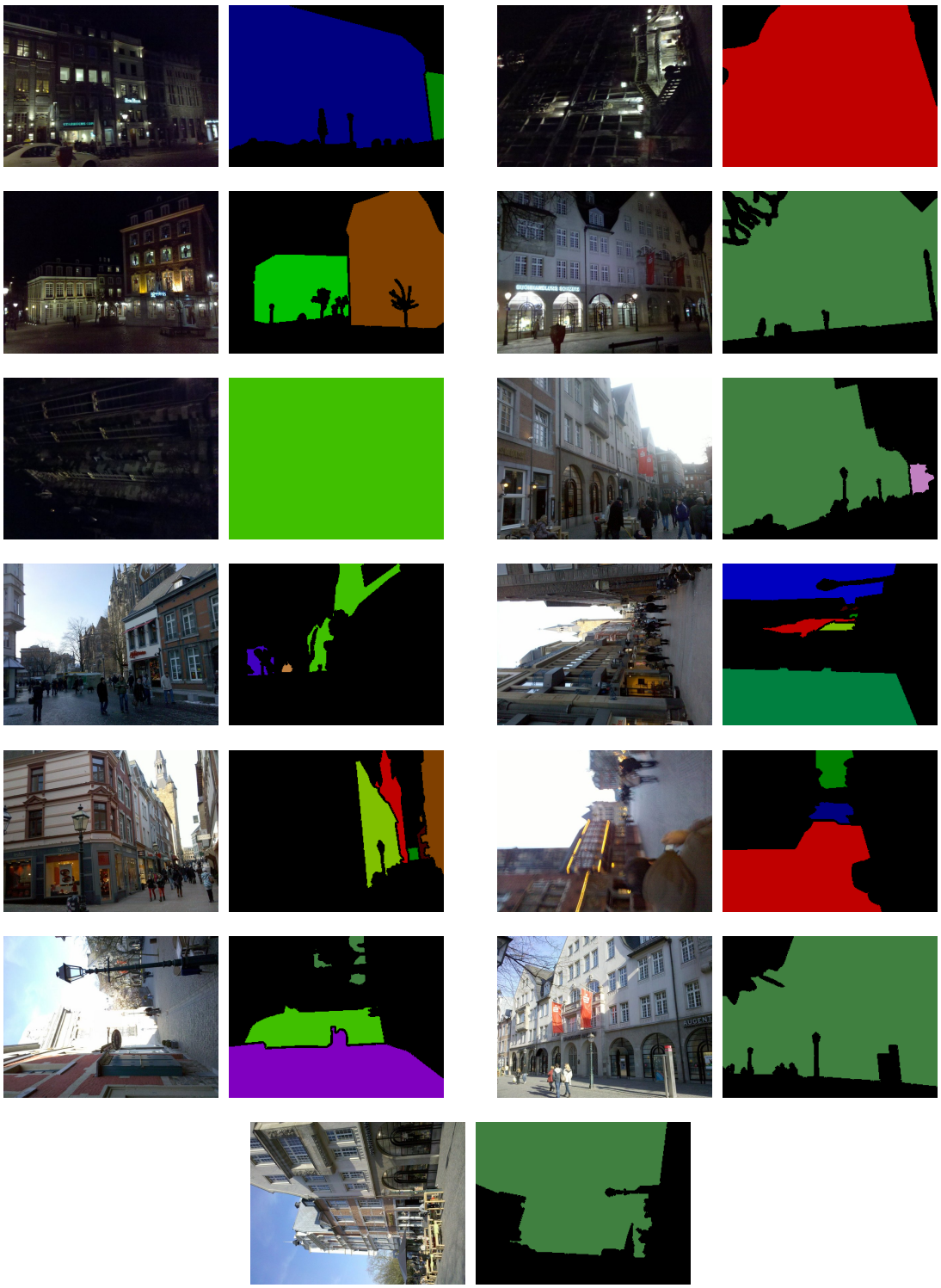


Figure 10: The last 13 of the 25 query images from the Aachen dataset that could not be localized with neither the method from [5] nor when matching against the individual building blocks.

the training images to obtain information about which regions in the training images correspond to which building blocks. However, this approach has two problems: First, the 3D point cloud is sparse, i.e., we would not obtain a continuous labeling in the training images but only gain information about discrete regions, which complicates the training of the classifiers. Second, projecting the points into images does not take occlusions caused by objects not covered by the model into account, e.g., persons or trees in front of a building might be ignored and their corresponding image regions would wrongly be labeled as belonging to one of the building blocks. We should be able to rely on the joint reconstruction and class segmentation approach from [6] (cf. Appendix B) that simultaneously generates a dense reconstruction of the scene together with a semantic labeling. The resulting model and its labels could be projected into the database images in order to handle to problem of occlusions. At the same time, we can easily transfer the labels from the sparse to the dense model since both models reside in the same coordinate system. We can then directly transfer the labels into the image regions obtained by projecting the dense model, enabling us to automatically generate labeled training images from which we can learn the semantic classifiers.

A central remaining question is whether we are actually able to learn discriminative classifiers once we have obtained labeled training images, i.e., whether the classifiers are actually able to distinguish between the different building blocks. For scenes such as those depicted in the Aachen and Vienna models, we expect that this is indeed possible since they contain rather unique buildings. However, for scenes containing mostly repetitive or even identical structures, as is the case in many indoor parking decks, it is very unlikely that the classifiers would be able to distinguish between these extremely similar structures. As these ambiguities are again caused by repetitive elements, we conclude that it would be better to directly handle repetitive and identical scene structures in a principled manner instead of trying to learn semantic classifiers. For future work, we will thus investigate principled approaches to handle ambiguous feature descriptors.

2.4 Conclusion

In this part of the report, we have considered ways to integrate semantic information into a state-of-the-art image-based localisation pipeline in order to obtain novel localisation approaches based on the ideas presented in Deliverable D2.2. We have investigated whether we can use semantic labels computed for a query image to filter out features found in regions that depict objects that are not included in the 3D scene model such as cars or pedestrians. While we succeeded in detecting and removing them, we observed that there is only a very slight improvement in efficiency that can be gained. This is due to the fact that only a small fraction of features is actually found on such objects. At the same time, removing features changes some of the characteristics of the localisation algorithm and introduces additional overhead that cancels out the positive effects of considering fewer features. While we believe that we can use semantics to reduce localisation times by tuning the localisation algorithms, the maximum speed-up that can be gained does not seem large enough to justify the required effort.

In the second part of this section, we have demonstrated that we can use global semantic information to handle normally ambiguous matches caused by repetitive scene elements, resulting in a significant gain in effectiveness. This gain was possible by using semantic image labels that directly associate regions in the query image with their corresponding structures in the 3D model, thus allowing us to avoid ambiguities. Yet, this approach will fail if the structures in the 3D model are ambiguous themselves, i.e., if we have similar or identical objects in different parts of the model. We thus conclude that the problem of ambiguous structures is better handled in a principled manner that explicitly takes the existence of such structures into account. We plan to investigate such principled approaches in future work.

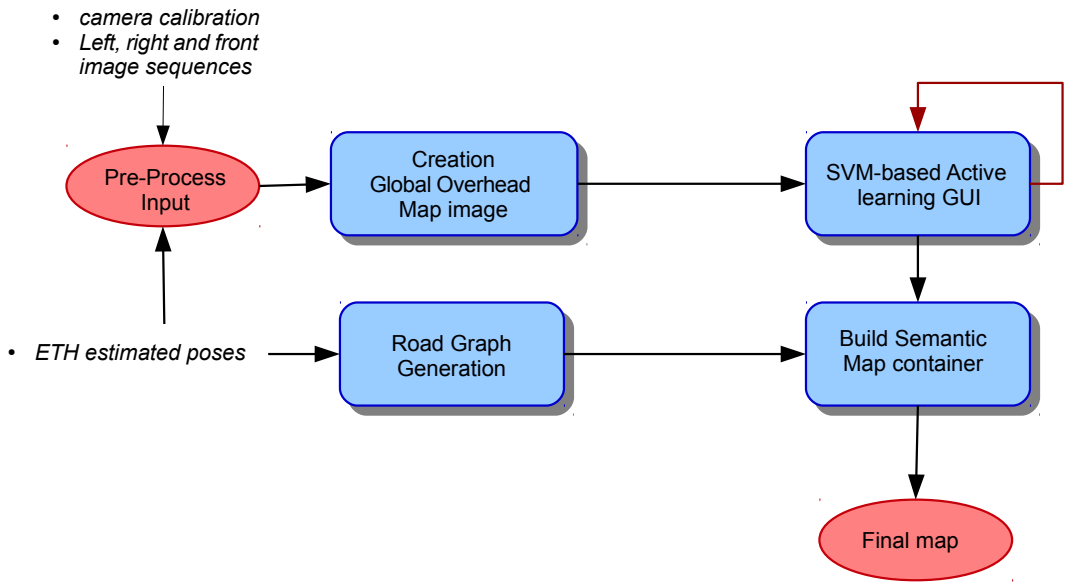
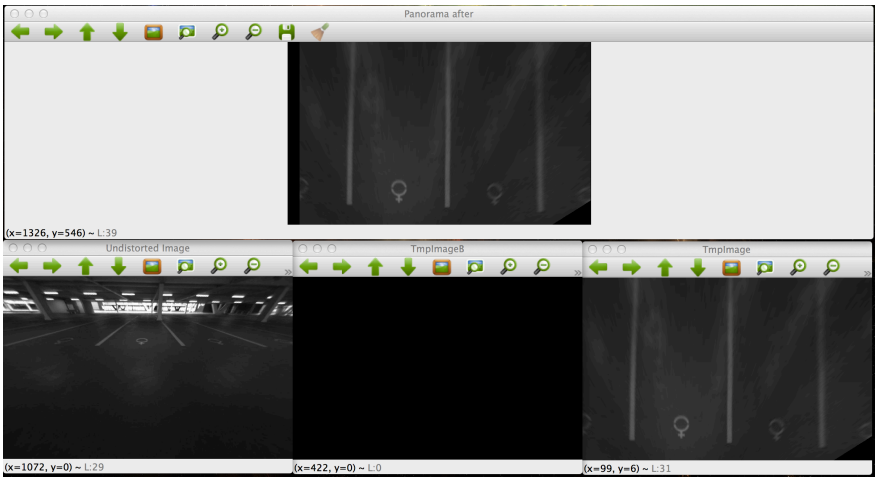


Figure 11: The semantic mapping pipeline developed by UOXF.DF team. At the *prepProcess* level, the image sequences are first rectified. Then, a *global overhead map image* of the parking lot is consistently composed following a local ground plane assumption by using the global estimated poses and a set of individual homographies. The resulting image is loaded from our *active learning interface* to detect road mark labels (e.g. parking spots, zebra crossings). After few active learning iterations, the pixel coordinates of the detected labels are back projected to 3D at the global reference. In parallel to the label classification, we build a road graph by smoothing the global estimated poses. Only key pose nodes are kept in the road a graph. The *semantic map container* is finally built with both topological and metric information: the former are used to define roads, lanes and ways as set of nodes. The second relates relative transformation between consecutive road nodes and labels relative to the road graph. The semantic map container is then delivered for future use to aid in tasks such as parking assignment and motion planning.

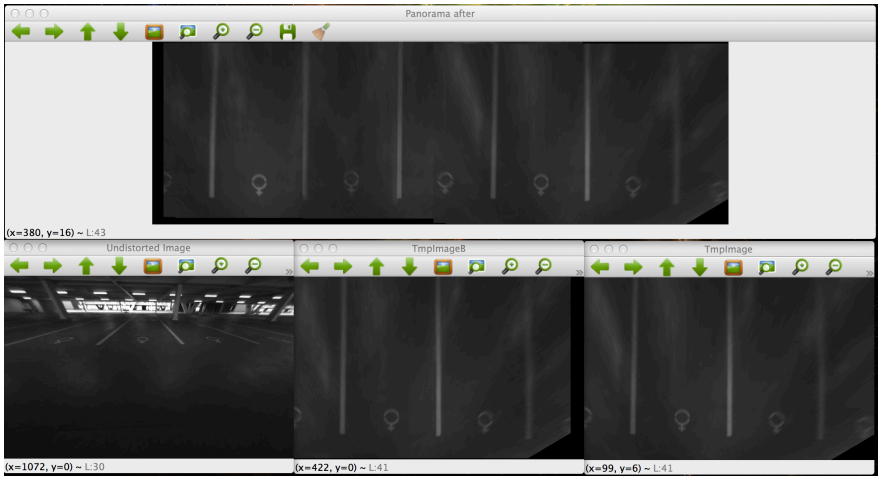
3 Semantic Mapping

Autonomous vehicles operating in places like parking lots can leverage of a higher-level understanding of the objects around them. For instance, the knowledge that there is an upcoming zebra crossing should be taken into account in the vehicle's current motion plan and speed. The labelling of parking spots is crucial in not only local-parking manoeuvres but also for efficient assignment of parking spaces. Because many of these useful semantic cues are fixed in place, it makes sense to build up reusable semantic maps of the areas in which we want our autonomous cars to operate. Usually these maps are created manually, which offers guarantees that are currently not available when using unsupervised machine learning classifiers. In this deliverable we will describe the pipeline of UOXF.DF to create semantic maps automatically using active learning for road mark (e.g parking spot) classification.

In deliverable D2.2 we showed the behaviour of general classification algorithms which offer some probability estimation for further decision-making stages (e.g. LogitBoost, Support Vector Machines (SVM), or Gaussian Process (GP)). In conclusion, we point out that our chosen classification framework must have to provide reliable mechanisms to estimate meaningful uncertainties in order to create reliable maps which can be used without fear of accidents.



(a) Stage 1



(b) Stage 2



(c) Stage 3

Figure 12: The overhead image builder is a semi-supervised process by which the operator can choose which images to use manually, or use all the images for automatic overhead-image generation. Image transformations and positions are calculated using vehicle poses supplied by ETH-CVG.

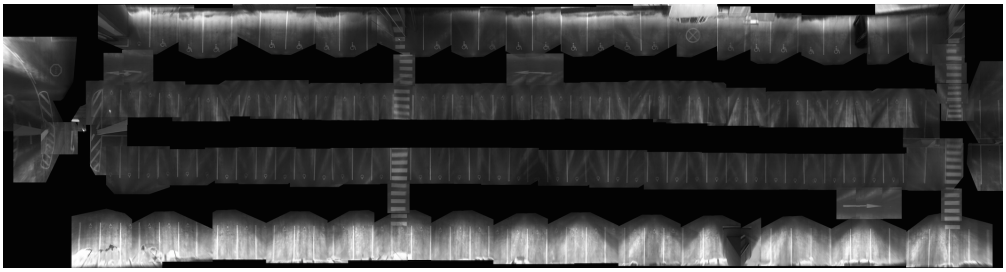


Figure 13: Overhead Image of the parking lot in Stuttgart obtained after stitching 150 images.

In our more recent works (cf. Appendix D and Appendinx E), we proposed an active learning framework for semantic mapping in mobile robotics and demonstrated it in the context of autonomous driving. In this framework data was selected for label disambiguation by a human supervisor using uncertainty sampling. Intuitively, an introspective classification framework – i.e. one which moderates its predictions by an estimate of how well it is placed to make a call in a particular situation – is particularly well suited to this task. Our preliminary evaluation on a publicly available dataset showed that an introspective *active learner* asks more informative questions compared to a more traditional non-introspective approach as simple SVM's in terms of learning rate.

In the work attached as Appendix F we also studied the road mark classification problem and proposed a method which employs a unique set of geometric feature functions within a probabilistic RUSBoost and Conditional Random Field (CRF) classification framework. This allowed us to jointly classify extracted road markings from a set of images. Furthermore, the semantics of road scenes (pedestrian approaches and no drive regions) was inferred based on marking classification results. Preliminary evaluation of the algorithms was carried out on a large real-life ground truth annotated dataset acquired from an autonomous vehicle.

In order to address the V-Charge goals, in this deliverable we report evaluation results on a dataset from experiments conducted in the BOSCH parking house located in Stuttgart. By following a similar idea as in Appendices D, E, the classification back-end employed for this V-Charge task relies on a SVM-based active learning approach. Due to the lack of road marks in the place, our results are focused on the classification of parking spots. Nonetheless, our approach can extrapolate to most general and richer scenarios with multiple labels. In the following subsections we provide details of the semantic map pipeline shown in Fig. 11.

3.1 Global Overhead Map Image

Similar to [20], our semantic mapping approach receives as input a sequence of n images collected during navigation at the parking lot with their corresponding estimated camera poses $\mathbf{T_{WCi}} \in \mathbf{SE3}$, $\mathbf{i} : \mathbf{1} \cdots \mathbf{n}$ delivered by the ETH-ASL group. All the images are rectified using a look-up table considering a fish-eye camera distortion model. For each camera (left, right, front), a top virtual camera $\mathcal{T}$ is created w.r.t the vehicle frame. At each frame k, a local ground plane is assumed to calculate an homography $\mathbf{H_{C_k\mathcal{T}}}$ that projects images from a particular camera on the corresponding image plane of the virtual camera. Given the global estimated poses, we calculate the set of homographies $\mathbf{H_{C_{k-1}k}}$ to sequentially stitching images yielding a global panorama image $\mathcal{P}$. This process is illustrated in Fig. 12. As final refinement, we perform a linear blending on the stitched images. Figure 13 shows the final result from a composition of 150 images.

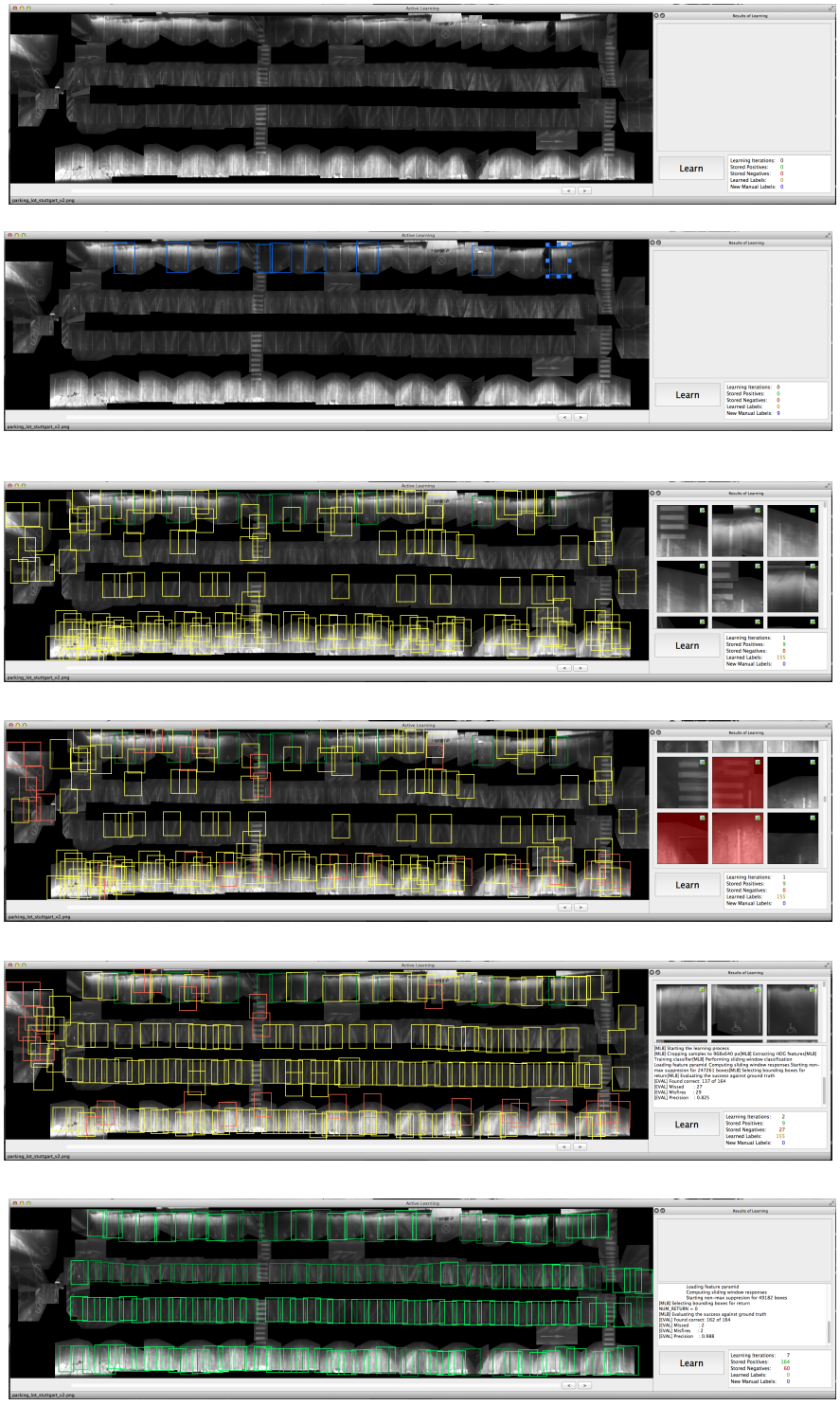


Figure 14: The active learning GUI, starting with a totally unlabeled image and a classifier which knows nothing, and ending with a completely labelled image and an excellent classifier ($> 95\%$ precision), which can then be used on other parking lots. At each round, the classifier learns using the green (parking space) and red (not parking space) boxes, and returns a set of hypothesis boxes in yellow. The user can then label any number of yellow boxes as correct (green) or incorrect (red) and click on "Learn" for re-training. Only 10 parking spaces were hand-drawn by the user over the course of the process.

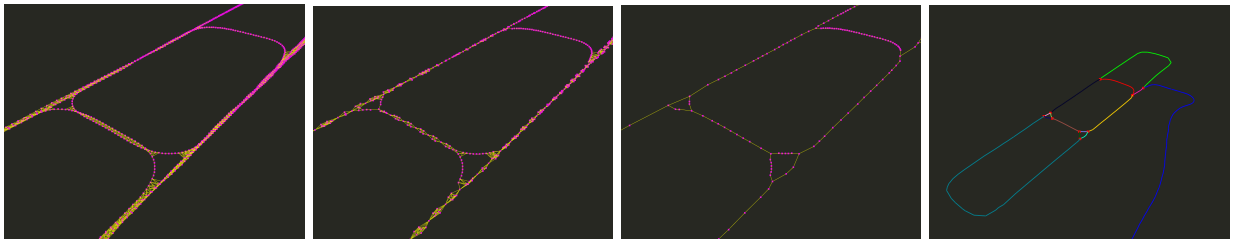


Figure 15: Road Graph obtained after recursive pruning of a graph with high connectivity. First, second and third column, sequential output of the algorithm. Forth column, final solution. Red points correspond to nodes that belong to intersections.

3.2 Active Learning for Label Classification

We have the option of manually labeling all the parking spaces in the overhead image in Figure 13, but that would be time-consuming, and every time we have a new parking lot we would have to re-label it, regardless of how similar it is to previously labelled parking lots. Instead, it is much more efficient to use machine learning. However, traditional classification requires the use of an extensive (> 200 images) repository of ground-truth labelled images for training, which in turn would have to be hand-labelled, so that doesn't solve the problem. That is why we employ active learning, a process which the human seeds with a handful of hand-labelled examples, and then the classifier chooses likely options for parking spaces, which are corrected or accepted by the human (which is easier than drawing them by hand) and then the process is repeated, iteratively improving the classifier.

In Figure 14 we have applied this process to the task of detecting parking spaces in the overhead image, and already by the 6th iteration we have a precision of $> 95\%$, which exceeds the stipulated level of 80% in the description of work. This was done with a total of 10 hand-drawn parking spaces, followed by the human operator accepting (green boxes) or rejecting (red boxes) the hypotheses (yellow boxes) for locations of parking spaces offered by the classifier. While we have used active learning for detecting parking spaces in this example, it can be used for a very wide variety of applications.

3.3 Consistent Road Graph creation

We address the problem of building road graphs from a given trajectory with thousands of poses as an alternative solution to image-based road detection when there is not sufficient visual information available about lane marks. We propose a technique that prunes cliques from a graph with high connectivity. Figure 15 shows the sequential output of our algorithm until convergence. First, we create a fully connected undirected graph $G(E, V)$ by traversing the pose-chain. Since we want to maintain the locality between nodes, all possible connections with nodes which distance is grater than a matched distance radius are rejected. Second, we recursively prune all maximal cliques in the graph, a clique that cannot be extended by including one more adjacent nodes. The solution of this approach is a road graph which consists of hundreds of pose nodes clustered as segments of smooth paths. Notice that this intrinsically corresponds to the definition of lanes. Also, the inner nature of the algorithm allows us to obtain minimum cliques that corresponds to the definition of road intersections. Figure 15 fourth column, shows the final road graph for the BOSCH parking lot.

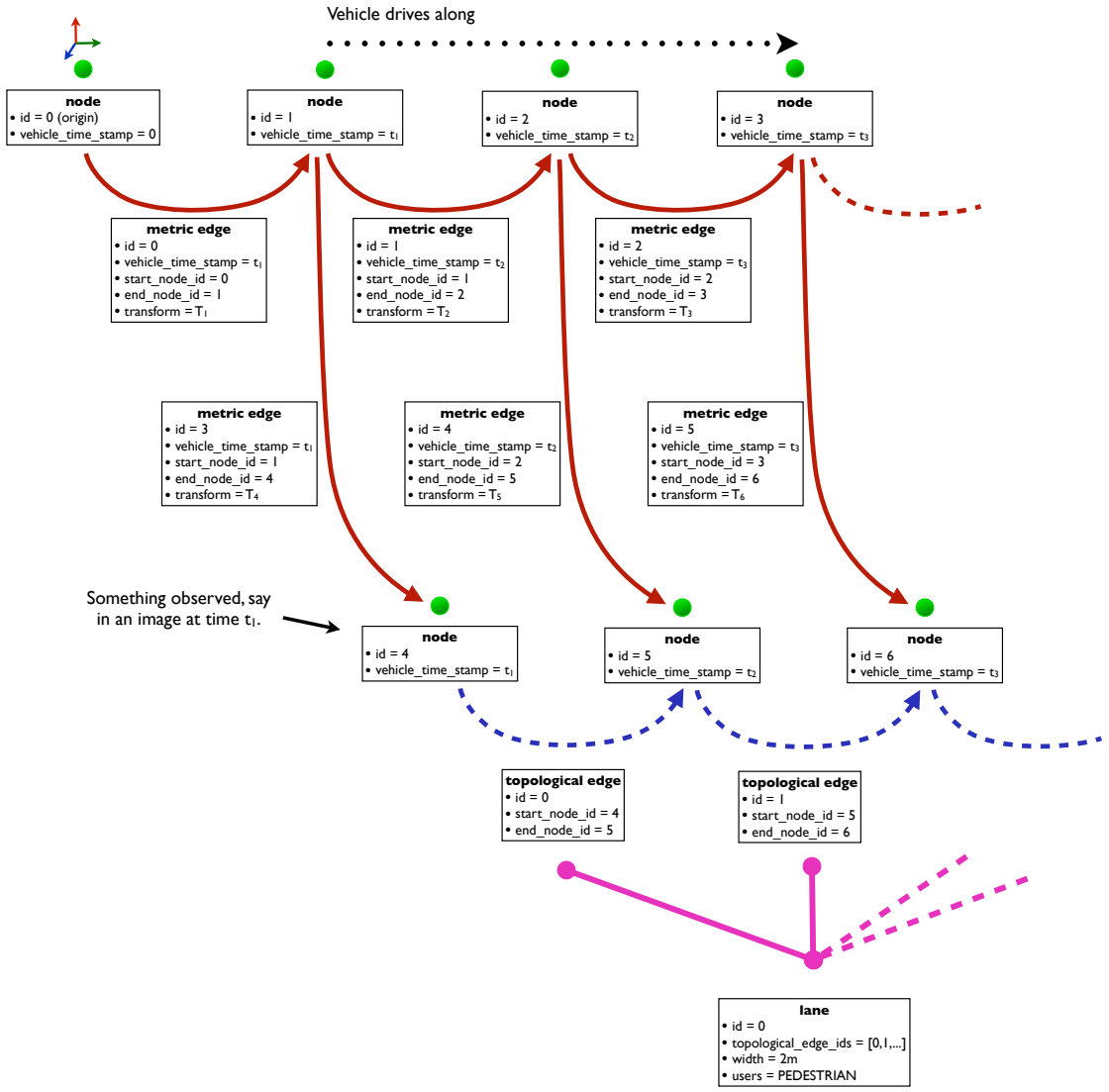


Figure 16: Example of the Semantic Map representation for a road with a single lane. Provided an estimated trajectory and the observations of a road/lane, we first create nodes for each vehicle frame and observation. Our relative map uses metric edges in three cases: to connect two consecutive vehicle pose nodes, roadlane consecutive nodes and between vehicle and road/lane nodes. We create topological edges to connect the nodes defining a lane (this assumption is consistent for any lane detection approach as a lane can be observed in different frames). That way, we can perform further improvements of the accuracy of the original map: the positions of the nodes representing the lane will just move accordingly, and no metric inconsistencies are introduced.

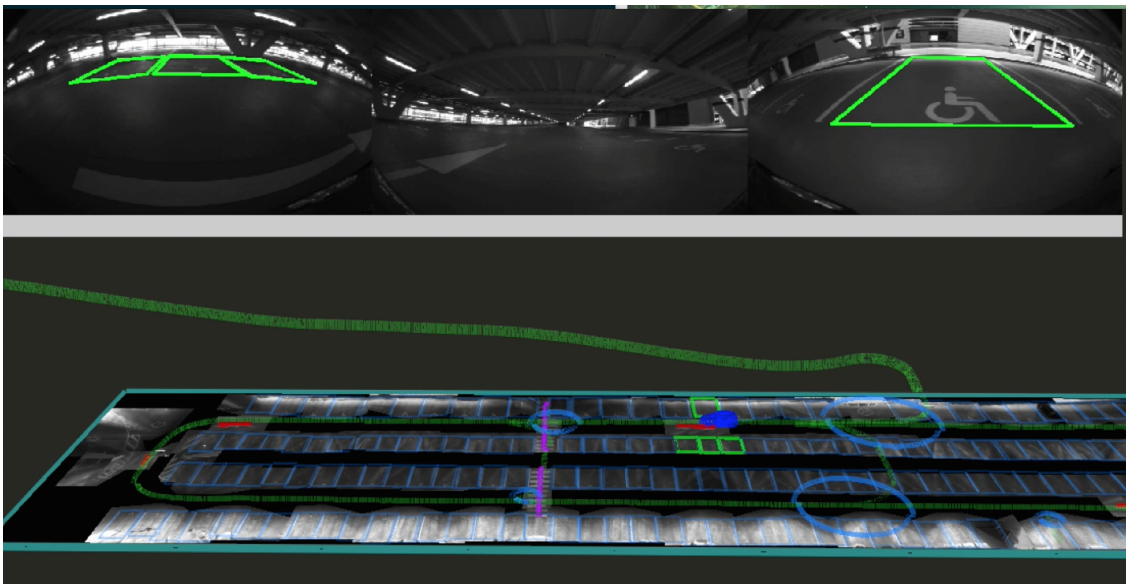


Figure 17: Final Semantic Map of BOSCH parking lot at Stuttgart. Top, projected labels onto the current camera frames. Bottom, representation in global coordinates of semantic map objects: Road graph (green trajectory), parking spots (blue rectangles), pedestrian crossings (magenta lines), road arrows (red lines), intersections (blue circles).

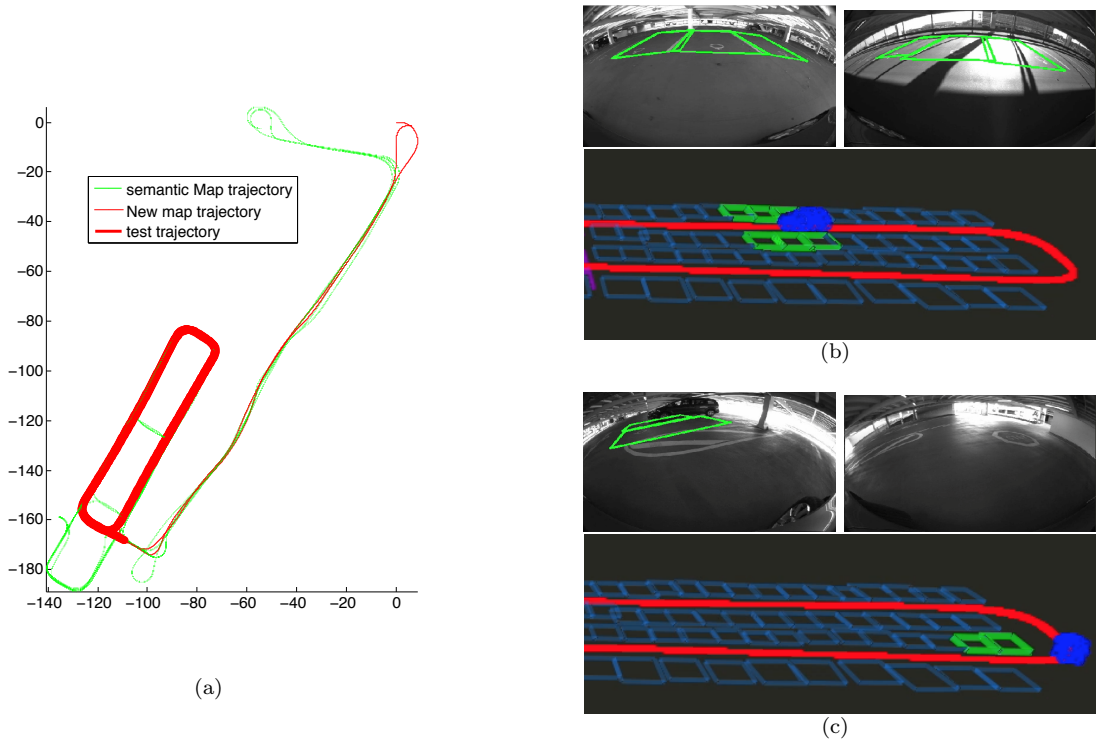
3.4 Semantic Map Structure

The detected labels along with the road graph are represented within a semantic map structure designed to fulfill the V-Charge specifications. Due to the nature of this information, we combine both topological and metric mapping graph concepts. Instead of a "global map" we adopt a "relative map" representation. We believe that relative frameworks are suitable in the sense that they allow adaptive pose relaxation, do not have to be globally consistent, and are stable for large scale maps.

Our structure considers three main elements contained in the semantic map: nodes, metric edges, and topological edges. A metric edge connects two nodes together with a transform $T \in SE3$. These are used for relating two vehicle position nodes, or a parking spot label relative to the nearest pose nodes. In contrast, a topological edge has no metric information but can also connect several nodes to define high level objects (lanes, ways, roads). These are used for example to represent lanes, which comprise a list of topological edges. The diagram in Fig. 16 shows an example of how the three would be used together as the car drives through an area and makes observations.

3.5 Usability of the Semantic Map

Despite tasks as the parking spot assignment and the local parking planner can be carried out during real execution, it requires a consistent mechanism to use labels from the semantic map at the current vehicle pose during a different navigation experience at the parking lot. Also, using the semantic map for motion aid requires the annotation of the map with statistics which can be made possible through a density estimation technique to learn a distribution from exemplar data.



(a)

(b)

(c)

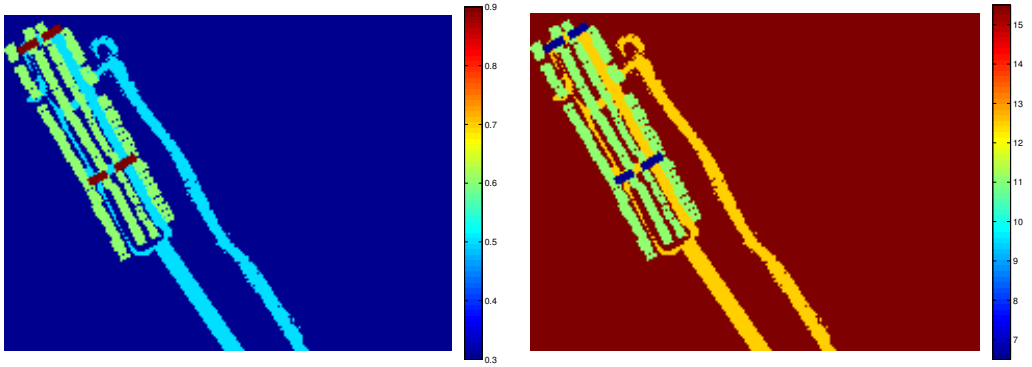
Figure 18: Transfer of parking spots from an apriori semantic map to an independent dataset with 40000 image pairs. (a) 3800 pair images were used to geometrically test the consistency of the transfer. Green line, trajectory of the apriori semantic map. Red line, registered images from the independent dataset (∼20000 poses). Red solid line, positions in which we succeed to transferring the labels. (b) Projected labels on images under illumination changes. (c) Projected labels when the vehicle is turning along the road graph.

3.5.1 Transfer of Semantic map labels to different maps

In collaboration with both the VW and ASL ETH teams, we are currently developing an approach that registers each image frame from an independent dataset of the parking lot to transfer labels from the current semantic map of the parking lot. Each new registered frame is represented also relative to the road graph. For each new pose, we search for the parking spots by projecting them in the corresponding camera image and keep only those that falls into the camera field of view. Figure 18 shows the results of the transfer of labels on a dataset consisting of 20000 left-right image pairs. In order to test the consistency of the transfer, we use 20% of the sequence (a subset of 3800 image pairs) corresponding to part of the trajectory when the car was navigating inside the parking lot.

3.5.2 Annotation of Semantic Map to aid motion planning

The semantic knowledge about various objects in the map, such as pedestrians, can be used to estimate sensible vehicle speeds given a position in the map. We envisage a system by which vehicles drive through the parking lot, and as they detect pedestrians, a semantic map is updated with the positions of those pedestrians. Driving through areas where pedestrians are seen more frequently prompts a change in behavior in the car, most simply a reduction in driving speed.



(a) The prior probabilities of a pedestrian

(b) The resulting speed map

Figure 19: Using the semantic map and some prior probabilities of a pedestrian occupying lanes, pedestrian crossings or parking spaces in order to generate a speed map. 19(a) shows the prior probabilities, and 19(b) shows the resulting suggested vehicle speed profile.

We have provided a speed map based on the prior estimations of the probability of a pedestrian occupying various types of space (see Figure 19(a)), such as pedestrian crossings, parking spaces, driving lanes, etc. Using this we produce a speed map of the parking lot, calculating the recommended speed with the formula

$$\text{recommended speed} = v_c \times p(\text{pedestrian}) + v_n \times \big(1 - p(\text{pedestrian})\big), \tag{4}$$

where v_c is the careful driving speed (we set this to be 5kph) and v_n is the normal driving speed (we set this to be 20kph). This map (see Figure 19(b)) can be used for driving in its present form.

3.6 Conclusion

For an specific task as autonomous parking, the V-Charge car could exploit the availability of our apriori semantic map of the environment since it has useful global and local information of parking spaces. Also we have shown how these tasks can leverage of information of other labels such as zebra crossings where the presence of pedestrians is more likely, such that the vehicle can be advised with recommendable speed limits. Moreover, the map can provide a road graph description useful for global/local trajectory planning. We have presented in this report the semantic mapping approach that uses consistent metric information to generate parking spot labels and a road graph with the v-charge specifications. We have developed the mechanisms that are required to facilitate the integration of the semantic map structure into the aforementioned tasks.

4 Conclusion

As can be seen from the reports attached to this document, Work Package 2 of the V-Charge project is well on track with its scientific, engineering, and with partial integration goals at the time of writing.

A Unsupervised Learning of Threshold for Geometric Verification in Visual-Based Loop-Closure

Authors: Lee, Pollefeys

Link to Table of Contents.

Unsupervised Learning of Threshold for Geometric Verification in Visual-Based Loop-Closure

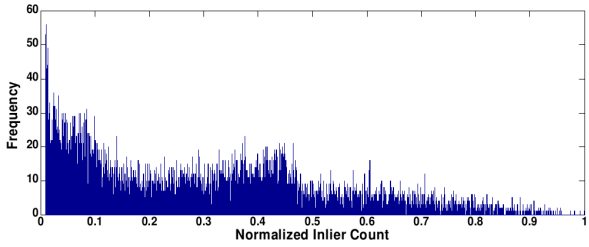
Gim Hee Lee, and Marc Pollefeys
Computer Vision and Geometry Lab, ETH Zürich, Switzerland
{glee@student, marc.pollefeys@inf}.ethz.ch

***Abstract*— A potential loop-closure image pair passes the geometric verification test if the number of inliers from the computation of the geometric constraint with RANSAC exceed a pre-defined threshold. The choice of the threshold is critical to the success of identifying the correct loop-closure image pairs. However, the value for this threshold often varies for different datasets and is chosen empirically. In this paper, we propose an unsupervised method that learns the threshold for geometric verification directly from the observed inlier counts of all the potential loop-closure image pairs. We model the distributions of the inlier counts from all the potential loop-closure image pairs with a two components Log-Normal mixture model - one component represents the state of non loop-closure and the other represents the state of loop-closure, and learn the parameters with the Expectation-Maximization algorithm. The intersection of the Log-Normal mixture distributions is the optimal threshold for geometric verification, i.e. the threshold that gives the minimum false positive and negative loop-closures. Our algorithm degenerates when there are too few or no loop-closures and we propose the χ^2 test to detect this degeneracy. We verify our proposed method with several large-scale datasets collected from both the multi-camera setup and stereo camera.**

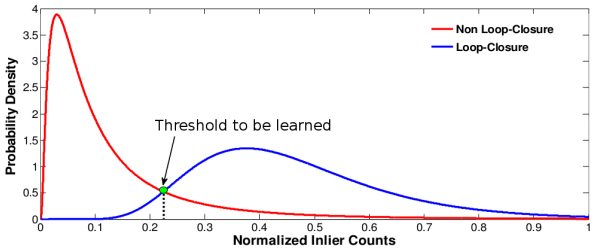
I. Introduction

Loop-closure for Simultaneous Localization and Mapping (SLAM) refers to the problem of detecting whether the robot at its current location sees a previously visited location and the computation of the geometric constraint that relates these two locations. In the recent years, many works [1]–[5] have demonstrated the effectiveness of using a camera to do place recognition for loop-closure detection. These works made used of the vocabulary-tree [6] that consists of the training and query phases. In the training phase, the vocabulary-tree is trained offline with image features such as the SURF [7] extracted from a given set of training images. In the query phase, a database is built from the unique IDs assigned to the query images and the extracted image features according to the pre-trained vocabulary-tree in the form of an inverted file for efficient retrieval. The database is queried with the image features extracted from the current query image and the output is the image ID from the database with the highest similarity score. The current query image and the image from the database with the highest similarity score form a potential loop-closure image pair.

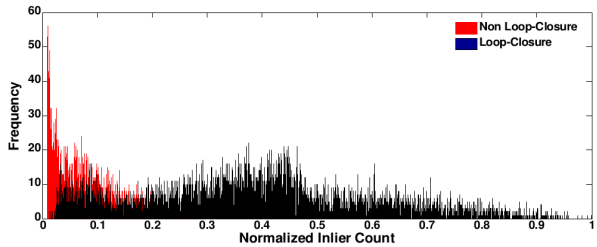
The selection of the potential loop-closure image pair with the vocabulary-tree is purely based on the appearance similarity between the images and does not take the geometric relation between the images into account. As a result, an additional step of geometric verification is taken to determine



(a)



(b)



(c)

Fig. 1. (a) An example distribution of the all the inlier count from the geometric verifications. (b) Two components Log-Normal mixture model and geometric verification threshold learned from the inlier counts. (c) GPS/INS ground truth distribution of the loop-closure and non loop-closure inlier counts.

the correctness of each potential loop-closure image pair. Geometric verification refers to the process of checking the consistency of the feature correspondences between the image pairs. This is achieved by counting the number of inliers from the computation of the geometric constraint between each potential loop-closure image pair with RANSAC [8]. The 5-point [9] and 8-point [10] algorithms are commonly used to compute the geometric constraint between an image pair from a monocular camera. Similarly, the absolute orientation algorithm [11] is used for stereo cameras, and the 3-point [5] and pose estimation [12] algorithms for multi-camera setups. A potential loop-closure image pair with the

number of inliers from the geometric constraint computation exceeding a pre-defined threshold is selected as the loop-closure image pair. The choice for this threshold is critical to the success of identifying the correct loop-closures. A threshold which is too high results in missing out many correct loop-closure image pairs while a threshold which is too low results in many false positives. However, there is no fixed value for this threshold as it often varies with different datasets. The value for the threshold is also unknown and has been chosen empirically in the existing works for visual-based loop-closures [1]–[5].

In this paper, we propose an unsupervised method that learns the threshold for geometric verification directly from the observed geometric verification inlier counts for all the potential loop-closure image pairs. We observed that the inlier counts from all the potential loop-closure image pairs, normalized with the maximum inlier count, can be approximated with a two components Log-Normal mixture distribution. An example of the distribution is shown in Figure 1(a). Each component represents the state of loop-closure or non loop-closure. The normalized inlier counts are the observed variables and we use the Expectation-Maximization (EM) algorithm to learn the hidden variables representing the parameters of the mixture model, and the latent variables representing the state of loop-closure or non loop-closure. The intersection of the Log-Normal mixture distributions is the optimal threshold for geometric verification, i.e. the threshold that gives the minimum false positive and negative loop-closures. Figure 1(b) shows an example of the learned model and Figure 1(c) shows the ground truth from GPS/INS. Our algorithm degenerates when there are too few or no loop-closures and we propose the χ^2 test to detect this degeneracy. We verify our proposed method with several large-scale datasets collected from both the multi-camera setup and stereo camera.

II. Unsupervised Learning of the Threshold

Let us denote the robot poses as $X = [x_1, x_2, ..., x_t]^T$ and the N potential loop-closure pose pairs as $\mathcal{X} = [..., \{x_i, x_j\}_n, ..., \{x_i, x_j\}_N]$. The potential loop-closure pose pairs $\mathcal{X}$ are the pose pairs with the most similar image pairs detected by the vocabulary-tree, i.e. $\{x_i, x_j\}_n$ is the n^{th} pose pair where the query image taken at x_i is most similar to the image at x_j from the vocabulary-tree database. Let $L = [l_1, l_2, ..., l_N]$ where $l_n \in \{0, 1\}$ denote the actual loop-closure state, i.e. $l_n = 1$ if the correspondence loop-closure robot pose pair $\{x_i, x_j\}_n$ is truly a loop-closure pair and vice versa. We further let $V = [v_1, v_2, ..., v_N]$ denote the inlier counts from the geometric verification of all the potential loop-closure image pairs. Each of the value v_n is normalized with the maximum value from V, i.e. $V = \frac{V}{max(V)}$. The normalization step makes it easier to model the distribution of the inlier counts since it is now within the range of 0 to 1.

A. Learning Mixture Model Parameters with EM

Assuming that each v_n is drawn independently, we can write the joint probability distribution of L and V as a two components Log-Normal mixture model given by

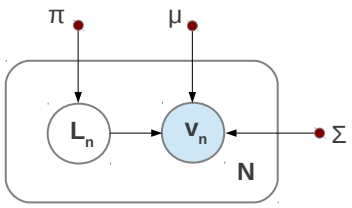


Fig. 2. Graphical model representing the threshold learning problem.

$$p(V, L|\mu, \Sigma, \pi) = \prod_{n=1}^{N} \prod_{k=1}^{2} (\pi_k ln\mathcal{N}(v_n|\mu_k, \Sigma_k))^{L_{n,k}} \quad (1)$$

where $\mu = [\mu_1, \mu_2]$, $\Sigma = [\Sigma_1, \Sigma_2]$ and $\pi = [\pi_1, \pi_2]$ are the means, standard deviations and mixing coefficients of the Log-Normal mixture model. Note that the mixing coefficients must sum to 1, i.e. $\pi_1 + \pi_2 = 1$. Figure 2 show the graphical model that represents the joint probability $p(V, L|\mu, \Sigma, \pi)$ where V is the observed variable, L is the latent variable and μ, Σ and π are the hidden variables. We shall denote the hidden variables jointly as $\phi = [\mu, \Sigma, \pi]$. The task is to find the hidden variables ϕ that maximizes the log likelihood $\ln p(V|\phi)$ given as

$$\underset{\phi}{\operatorname{argmax}} \ln p(V|\phi) = \underset{\phi}{\operatorname{argmax}} \ln\{\sum_{L} p(V, L|\phi)\} \quad (2)$$

which involves marginalizing out the latent variable L. One way to find the maximum likelihood in Equation 2 is by using the EM algorithm which iterates between the Expectation and Maximization steps. In the Expectation step, the posterior distribution $p(L|V, \phi')$ is computed based on the hidden variables ϕ' computed from the previous Maximization step. The posterior $p(L|V, \phi')$ is a $N \times 2$ matrix with each entry given by $\mathcal{T}(L_{n,k})$, where $n = 1, 2, ..., N$ and $k = 1, 2$.

$$\mathcal{T}(L_{n,k}) = p(L_{n,k}|v_n, \phi') = \frac{\pi_k ln\mathcal{N}(v_n|\mu_k, \Sigma_k)}{\sum_{j=1}^{2} \pi_j ln\mathcal{N}(v_n|\mu_j, \Sigma_j)} \quad (3)$$

The posterior probability $\mathcal{T}(L_{n,k})$ is used to evaluate the expectation of the log likelihood of the joint probability $p(V, L|\phi)$ given by

$$\begin{aligned} Q(\phi, \phi') &= \sum_{L} p(L|V, \phi') \ln p(V, L|\phi) \\ &= \sum_{n=1}^{N} \sum_{k=1}^{2} \mathcal{T}(L_{n,k}) \{\ln \pi_k + \ln ln\mathcal{N}(v_n|\mu_k, \Sigma_k)\} \end{aligned} \quad (4)$$

In the Maximization step, we compute the current values of the hidden variable ϕ by maximizing the expectation from Equation 4.

$$\underset{\phi}{\operatorname{argmax}}\ Q(\phi, \phi') \quad (5)$$

The Maximization step given by Equation 5 can be expressed in closed-form by setting the partial differentiation of $Q(\phi, \phi')$ with respect to the individual variables in ϕ to 0 and enforcing the constraint of $\pi_1 + \pi_2 = 1$. This gives

$$\mu_k = \frac{\sum_{n=1}^{N} \mathcal{T}(L_{n,k}) \ln(v_n)}{\sum_{n=1}^{N} \mathcal{T}(L_{n,k})} \tag{6a}$$

$$\Sigma_k = \frac{\sum_{n=1}^{N} \mathcal{T}(L_{n,k}) (\ln(v_n) - \mu_k)^2}{\sum_{n=1}^{N} \mathcal{T}(L_{n,k})} \tag{6b}$$

$$\pi_k = \frac{\sum_{n=1}^{N} \mathcal{T}(L_{n,k})}{N} \tag{6c}$$

The EM steps are iterated until convergence, i.e. there is minimal or no more changes to the hidden variables μ_k, Σ_k and π_k.

B. Solving for the Threshold

The parameters of the two components Log-Normal mixture distribution which we have solved with the EM algorithm models the distributions of the inlier counts from the non loop-closure and loop-closure image pairs. The intersection point v_T of the two Log-Normal distributions defines the point where $v_n < v_T$ is the region with a higher probability of non loop-closure, i.e. $l_n = 0$. Similarly, $v_n > v_T$ defines the region with a higher probability of loop-closure, i.e. $l_n = 1$. We solve for the threshold v_T by finding the root of the difference in the two Log-Normal distributions

$$f(v) = \pi_1 ln\mathcal{N}(v|\mu_1, \Sigma_1) - \pi_2 ln\mathcal{N}(v|\mu_2, \Sigma_2) = 0 \tag{7}$$

Since $f(v)$ is a non-linear function, we solve for the root with the Brent's method [13] which by default uses the less robust but faster Secant method to iteratively search for the root and fall back to the slower but more robust Bisection method if needed. Finally, we compute the threshold for geometric verification as $T_{GV} = max(V)v_T$ after denormalization.

C. Degenerated Case

Our algorithm for unsupervised learning of the threshold for geometric verification degenerates when there is too few or no loop-closures. In this case, there is only one dominant Log-Normal distribution from the non-loop closure inlier counts. The EM algorithm would not be able to detect the missing Log-Normal distribution that represents the loop-closure in the degenerated case. We propose to detect the degeneracy by performing the χ^2 test on the optimization results after SLAM optimization. Intuitively, a degenerated case includes wrong loop-closure constraints which causes wrong convergence in the optimization. As a result, the total Mahalanobis distance D^2 after optimization is high and will fail the χ^2 test.

A popular approach to perform the optimization with the loop-closure constraints is the pose-graph SLAM which represents the SLAM problem as a graph. The pose-graph SLAM optimization minimizes the errors between the predicted poses $X = [x_1, ..., x_i, ...x_j, ...x_t]^T$ represented by the vertices and the $\mathcal{K}$ set of measured constraints $Z = [..., z_{i,j}, ...]^T$ represented by the edges given by

$$\underset{X}{\operatorname{argmin}} \sum_{\mathcal{K} \in \{i,j\}} ||z_{i,j} - h(x_i, x_j)||^2_{Q_{i,j}} \tag{8}$$

where $h(.)$ is the function which computes the relative transformation between two given poses x_i and x_j, and $Q_{i,j}$ is the error covariance of the measurement $z_{i,j}$. The total Mahalanobis distance D^2 of the optimized poses $X^* = [x_1^*, ..., x_i^*, ...x_j^*, ...x_t^*]^T$ which reflects the "goodness-of-fit" is given by

$$D^2 = \sum_{\mathcal{K} \in \{i,j\}} ||z_{i,j} - h(x_i^*, x_j^*)||^2_{Q_{i,j}} \tag{9}$$

We compare the Mahalanobis distance D^2 with the critical value $\chi^2_{\mathcal{K},\alpha}$ of the χ^2 distribution where $\mathcal{K}$ is the degree of freedom which is equal to the number of measured constraints, and α is the significance level, i.e. there is a probability of α that D^2 would be greater than the critical value $\chi^2_{\mathcal{K},\alpha}$ by chance. α is usually set at 0.05 or 0.025. A degenerated case is detected when the total Mahalanobis distance is higher than the critical value, i.e. $D^2 > \chi^2_{\mathcal{K},\alpha}$.

D. Summary

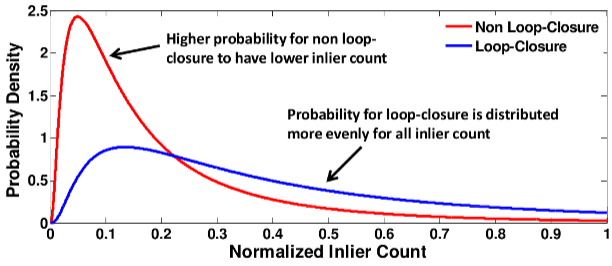


Fig. 3. Initial two components Log-Normal mixture distribution.

Algorithm 1 shows the pseudo-code of our algorithm for unsupervised learning of the threshold for geometric verification. The input is the inlier counts $V = [v_1, v_2, ..., v_N]$ from all the potential loop-closure image pairs generated by the vocabulary-tree. The output is the threshold T_{GV} for geometric verification. Line 2 normalizes V with its maximum value so that the values lie in the range of 0 to 1. Line 4 initializes the parameters ϕ from the two components Log-Normal mixture distribution. The mixing coefficients π are initialized to 0.5 which give the two components of the Log-Normal distribution equal weights. Figure 3 shows the initial Log-Normal mixture distribution with means $\mu = [-2, 1]$ and standard deviations $\Sigma = [1, 1]$. The non loop-closure Log-Normal distribution has a high peak at 0.05 and a long tail with values close to 0. In comparison, the loop-closure Log-Normal distribution is distributed more evenly with lower values near the peak and higher values at the tail region. We chose this initial distribution that is close to the final solutions, where there is always a higher occurrence of low inlier counts which are non loop-closure and higher occurrence of high inlier counts which are loop-closure.

Lines 7 to 31 are the EM loop which terminates when the change in ϕ is small. Lines 10 to 20 are the Expectation

Algorithm 1 Unsupervised Learning of Threshold for Geometric Verification.

Require: Inlier counts $V = [v_1, v_2, ..., v_N]$ from all the potential loop-closure image pairs.

Ensure: Threshold for geometric verification T_{GV}.

1: // Normalize inlier count list
2: $V = \frac{V}{max(V)}$;
3: // Parameters initialization
4: $\mu = [-2.0, 1.0]$; $\Sigma = [1.0, 1.0]$; $\pi = [0.5, 0.5]$;
5: $\phi = [\mu, \Sigma, \pi]$; $\phi' = 0$;
6: // EM algorithm
7: **while** $|\phi' - \phi| > \delta$ **do**
8: $\phi' = \phi$;
9: // Expectation step, Equation 3
10: **for** $n = 1$ to N **do**
11: // Compute denominator of Equation 3
12: $\mathcal{Z}_n = 0$;
13: **for** $j = 1$ to 2 **do**
14: $\mathcal{Z}_n = \mathcal{Z}_n + \frac{\pi'_j}{v_n \Sigma'_j \sqrt{2\pi}} \exp(-\frac{(\ln(v_n) - \mu'_j)^2}{2\Sigma'^2_j})$;
15: **end for**
16: **for** $k = 1$ to 2 **do**
17: // Compute posterior probability $\mathcal{T}(L_{n,k})$
18: $\mathcal{T}(n,k) = \frac{\pi'_k}{\mathcal{Z}_n v_n \Sigma'_k \sqrt{2\pi}} \exp(-\frac{(\ln(v_n) - \mu'_k)^2}{2\Sigma'^2_k})$;
19: **end for**
20: **end for**
21: // Maximization step, Equation 6
22: **for** $k = 1$ to 2 **do**
23: $\mu_k = 0$; $\Sigma_k = 0$; $\pi_k = 0$;
24: **for** $n = 1$ to N **do**
25: $\mu_k = \mu_k + \mathcal{T}(n,k) \ln(v_n)$;
26: $\Sigma_k = \Sigma_k + \mathcal{T}(n,k)(\ln(v_n) - \mu_k)^2$;
27: $\pi_k = \pi_k + \mathcal{T}(n,k)$;
28: **end for**
29: $\mu_k = \frac{\mu_k}{\pi_k}$; $\Sigma_k = \frac{\Sigma_k}{\pi_k}$; $\pi_k = \frac{\pi_k}{N}$;
30: **end for**
31: **end while**
32: Solve for the root v_T in Equation 7.
33: $T_{GV} = max(V) v_T$; // Denormalization
34: **return** T_{GV};

step where the posterior distribution $\mathcal{T}(L_{n,k})$ is computed, and Lines 22 to 30 are the Maximization step where the parameter ϕ is updated. Line 32 solves for the intersection of the two Log-Normal distributions and Line 33 does denormalization to compute the threshold T_{GV}.

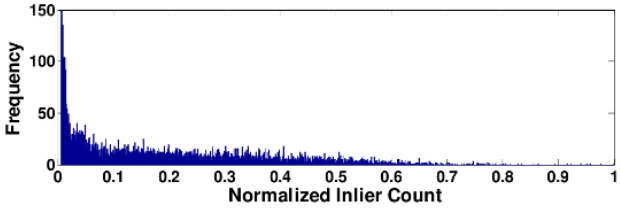
III. Results

We verify our proposed algorithm with several large-scale real-world datasets - two datasets from a multi-camera setup and one dataset from a stereo camera. We chose the multi-camera and stereo setups because these two setups allow us to compute the loop-closure constraint with metric scale [5]. We show the χ^2 test results for all the three datasets. In addition, we show an example of the degenerated case and its detection with the χ^2 test.

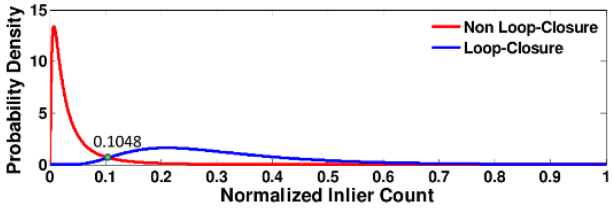
A. Multi-Camera System



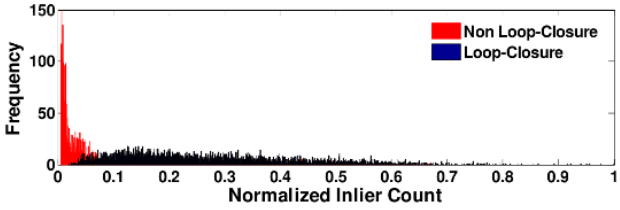
Fig. 4. Our car equipped with a multi-camera system with minimal overlapping field-of-views and GPS/INS system for ground truth.



(a)



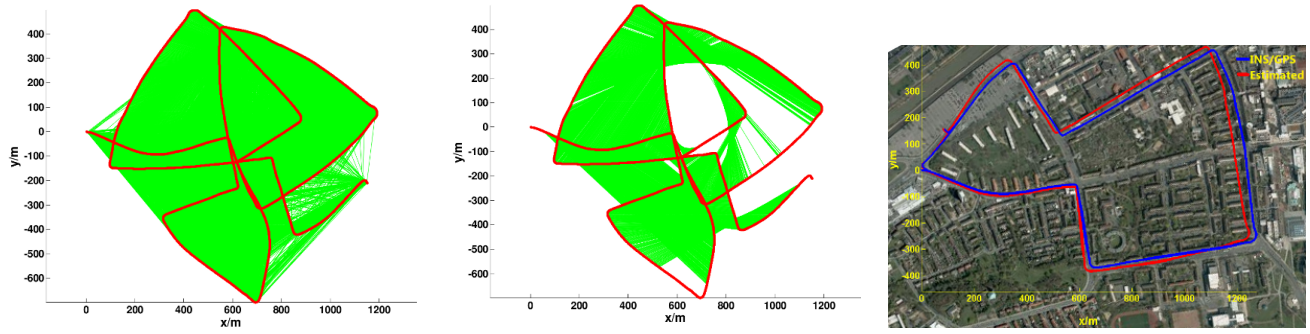
(b)



(c)

Fig. 5. Wolfsburg dataset with a multi-camera system. (a) Distribution of the all the inlier counts from the geometric verifications. (b) Two components Log-Normal mixture model and geometric verification threshold learned from the inlier counts. The threshold is 220 after denormalization. (c) Ground truth distribution from INS/GPS.

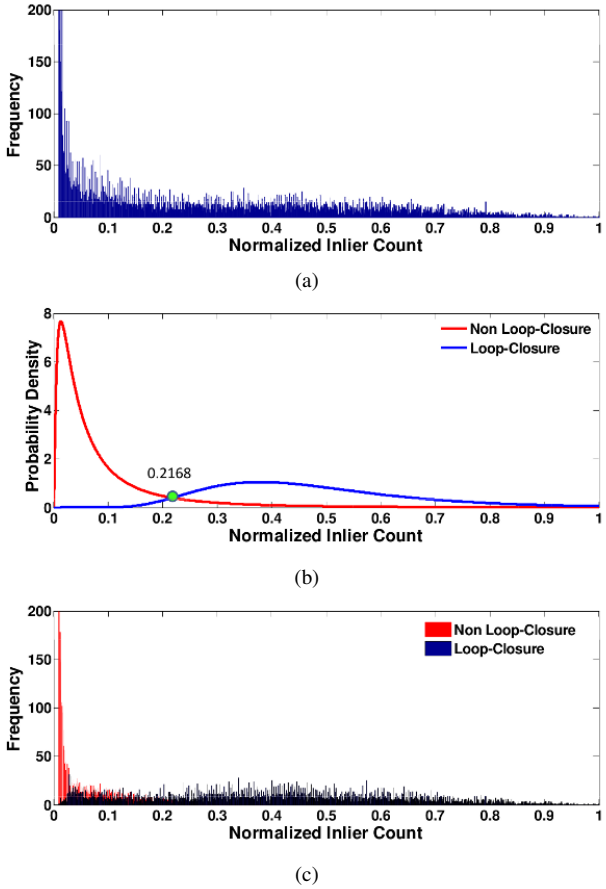
Figure 4 shows the car platform we used to collect the datasets. The car is equipped with four fish-eye cameras looking front, left, rear and right with minimal overlapping field-of-views. The car is also equipped with wheel odometry and INS/GPS system for ground truth. The first dataset - the Wolfsburg dataset was collected by driving the car along public roads in Wolfsburg Germany where it is largely urban scenes. The Wolfsburg dataset consists of a total of 13250×4 images from a trajectory that spans across approximately 9 km forming two large closed loops. We form the pose-graph with the wheel odometry readings. The loop-closure candidates are found from the vocabulary-tree as mentioned in Section I and the loop-closure constraints are computed with the 3-point algorithm described in [5]. Similar to [5], we maintain only one vocabulary-tree for all the cameras



(a) (b) (c)

Fig. 6. Results from the Wolfsburg dataset. (a) Pose-graph from wheel odometry trajectory (red) with all the detected loop-closures (green). (b) Pose-graph from wheel odometry (red) with the loop-closures (green) after applying the geometric verification threshold learned from our algorithm. (c) Final result after pose-graph optimization (red) compared with the INS/GPS ground truth (blue).

from our multi-camera system. We do so by assigning unique image IDs given by imageID = frameID $\times$ n + cameraID, where n is the total number of cameras in the multi-camera system.



(a)

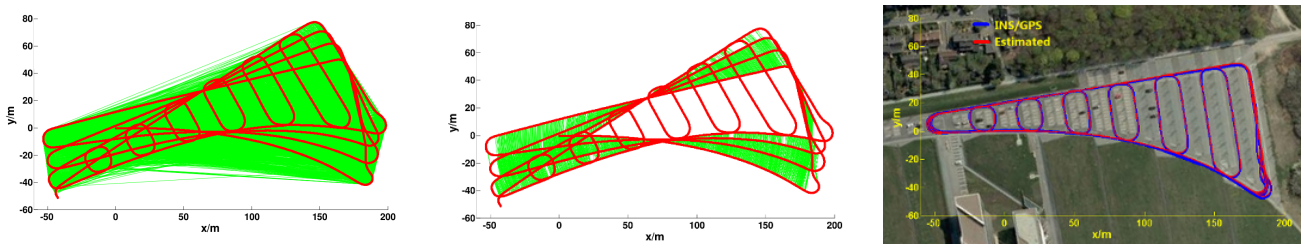
(b)

(c)

Fig. 7. Carpark dataset with a multi-camera system. (a) Distribution of the all the inlier counts from the geometric verifications. (b) Two components Log-Normal mixture model and geometric verification threshold learned from the inlier counts. (c) Ground truth distribution from INS/GPS.

Figure 5(a) shows the distribution of the normalized inlier counts obtained from the computations of the geometric constraints with the 3-point algorithm for the Wolfsburg dataset. Figure 5(b) shows the two components Log-Normal mixture distribution learned from the observations in Figure 5(a) with our proposed algorithm. The intersection of the two Log-Normal distributions is found to be 0.1048, which is 220 after denormalization. Figure 5(c) shows the ground truth distribution from the INS/GPS. We can see that the intersection of the two distributions is very close to the estimate we have obtained from our algorithm. Figure 6(a) shows the pose-graph from wheel odometry (red) with all the detected loop-closures (green) from the vocabulary-tree where there are many wrong loop-closures. Figure 6(b) shows the pose-graph with the loop-closures after applying the geometric verification threshold learned from our algorithm where majority of the wrong loop-closures are removed. Figure 6(c) shows the pose-graph (red) after optimization overlaid on the satellite image. We apply the robust pose-graph optimization proposed in [14] to minimize the effects of the a small number of non-loop closures with higher inlier count than the threshold. We compare the estimated pose-graph with the INS/GPS ground truth (blue) where we can see that our estimated pose-graph is sufficiently close to the ground truth.

We apply our algorithm on another dataset - the Carpark dataset collected from the same car with the multi-camera setup. The dataset was collected by driving the car around a huge car park besides an office building. A total of 12000×4 images are collected over a trajectory that spans across approximately 3.5 km forming 3 large and 6 nested loops. Figure 7(a) shows the distribution of the normalized inlier counts obtained from the computations of the geometric constraints with the 3-point algorithm for the Carpark dataset. Figure 7(b) shows the two components Log-Normal mixture distribution learned from the observations from Figure 7(b) with our algorithm. The intersection of the two Log-Normal distributions is found to be 0.2168, which is 277 after denormalization. Figure 7(c) shows the ground truth distribution from the INS/GPS. We can see that the intersection of the two distributions is very close to the estimate we have obtained from our algorithm. It is important to note that that the threshold for geometric verification for the datasets differs slightly even though they are collected from the same platform. This explains our approach to learn the threshold from each dataset in a batch process for the best results. However, it is also possible to take the most



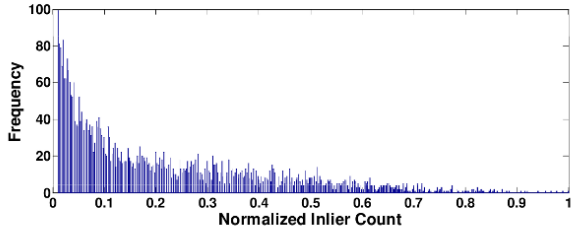
(a) (b) (c)

Fig. 8. Results from the Carpark dataset. (a) Pose-graph from wheel odometry trajectory (red) with all the detected loop-closures (green). (b) Pose-graph from wheel odometry (red) with the loop-closures (green) after applying the geometric verification threshold learned from our algorithm. (c) Final result after pose-graph optimization(red) compared with the INS/GPS ground truth (blue).

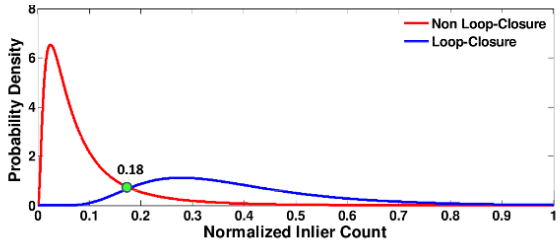
conservative threshold, i.e. highest threshold learned from multiple datasets collected from the same platform and use it for online geometric verifications.

Figure 8(a) shows the pose-graph from wheel odometry (red) with all the detected loop-closures (green) from the vocabulary-tree where there are many wrong loop-closures. Figure 8(b) shows the pose-graph after applying the geometric verification threshold learned from our algorithm where majority of the wrong loop-closures are removed. Figure 8(c) shows the final pose-graph (red) after robust pose-graph optimization. We compare the estimated pose-graph with the INS/GPS ground truth (blue) where we can see that our estimated pose-graph follows the ground truth very closely.

B. Stereo Camera



(a)



(b)

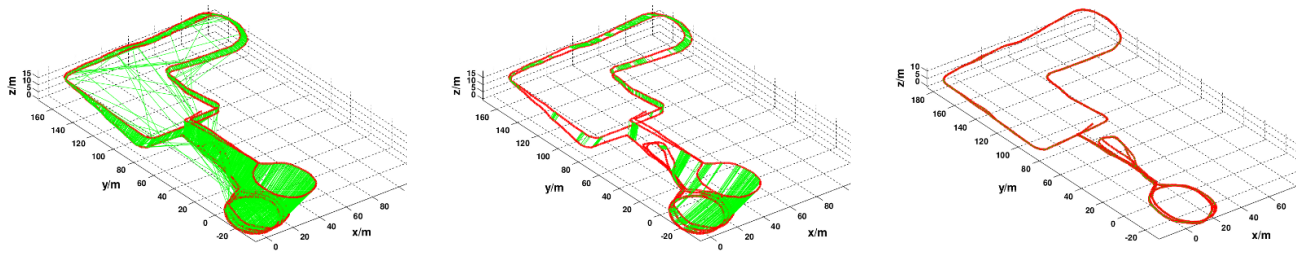
Fig. 9. New College dataset with a stereo camera. (a) Distribution of the all the inlier counts from the geometric verifications. (b) Two components Log-Normal mixture model and geometric verification threshold learned from the inlier counts.

We also test our algorithm on the New College dataset [15] which was collected with a stereo camera mounted on a ground robot. A total of 48241 stereo frames from a trajectory that spans across approximately 1.25 km are used. The vocabulary-tree for visual loop-closure is formed from the left stereo images and the loop-closure constraints are computed from the absolute orientation algorithm [11]. Figure 9(a) shows the distribution of the distribution of the normalized inlier counts obtained from the computations of the geometric constraints with the absolute orientation for the New College dataset. Note that this distribution is similar to Figure 5(a) and 7(a) even though a different camera setup is used. Figure 9(b) shows the two components Log-Normal mixture distributions learned with our algorithm. The intersection of the two Log-Normal distributions is found to be 0.18, which is 76 after denormalization. Figure 10(a) shows the pose-graph from stereo visual odometry [16] (red) with all the detected loop-closures (green) from the vocabulary-tree where there are many wrong loop-closures. Figure 10(b) shows the pose-graph with the loop-closures after applying the geometric verification threshold learned from our algorithm where majority of the wrong loop-closures are removed. Figure 10(c) shows the pose-graph after robust robust pose-graph optimization. Although there is no INS/GPS ground truth for the New College dataset, we can see that the pose-graph after optimization appears reasonable with all the deviations in the z-axis removed.

C. Degenerated Case

We show an example of the degenerated case with a dataset collected from our car platform with the multi-camera setup. The dataset consists of 1600×4 images from a trajectory that spans across approximately 300 m forming one loop with the starting and ending points at approximately the same location. The small number of loop-closures means that there is not enough information for our algorithm to learn the threshold correctly. Figure 11(a) shows the pose-graph from wheel odometry (red) and all the loop-closures from the vocabulary-tree. Figure 11(b) shows the pose-graph with the loop-closures after applying the threshold learned from our algorithm where it is clearly visible that some wrong loop-closures remain. Figure 11(c) shows the pose-graph (red) after robust pose-graph optimization compared with the ground truth (blue). The ground truth is obtained by performing robust pose-graph optimization on the manually chosen correct loop-closures. It can be observed that the estimated pose-graph is slightly distorted by the wrong loop-closures even after applying the robust optimization because the ratio of correct to wrong loop-closures is too low.

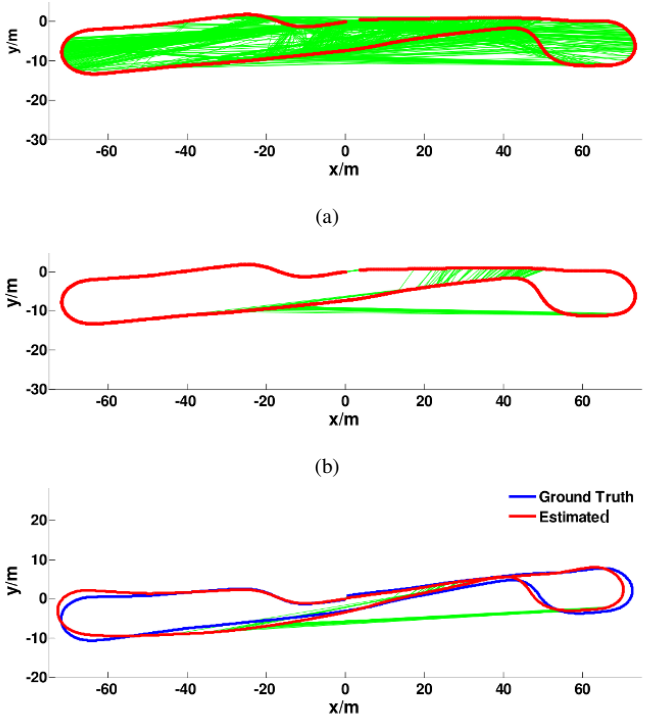
As mentioned in Section II-C, we do the χ^2 test to detect the degenerated case. Table I shows the χ^2 test results for all the datasets at a significance level $\alpha = 0.05$. We can see from the χ^2 test results that the first three datasets -



(a) (b) (c)

Fig. 10. Results from the New College Dataset. (a) Pose-graph from visual odometry trajectory (red) with all the detected loop-closures (green). (b) Pose-graph from visual odometry (red) with the loop-closures (green) after applying the geometric verification threshold learned from our algorithm. (c) Final result after pose-graph optimization.

Wolfsburg, Carpark and New College pass the χ^2 test with $D^2 < \chi^2_{\alpha=0.05}$ while the degenerated dataset fails the χ^2 test with $D^2 > \chi^2_{\alpha=0.05}$.



(a)

(b)

(c)

Fig. 11. Results from the Degenerated dataset. (a) Pose-graph from wheel odometry trajectory (red) with all the detected loop-closures (green). (b) Pose-graph from wheel odometry (red) with the loop-closures (green) after applying the geometric verification threshold learned from our algorithm. (c) Wrong convergence after pose-graph optimization(red) compared with the ground truth (blue).

IV. Conclusion

The threshold for geometric verification is crucial in identifying the correct loop-closures. However, the threshold varies for different datasets and has been chosen empirically in the existing works for visual-based loop-closure. We described a method for unsupervised learning of the threshold for geometric verification in this paper. Our method is based on the EM algorithm which learns the threshold from the inlier counts generated from the RANSAC computation of the geometric constraints between all potential loop-closure image pairs. We verified our method with multiple large-scale datasets from both the multi-camera and stereo setups.

TABLE I

χ^2 Test Results for the Datasets

Dataset	DOF	$\chi^2_{\alpha=0.05}$	D^2
Wolfsburg	12840	1.3105e+04	4.1249e+03
Carpark	17506	1.7815e+04	1.0325e+04
New College	9713	9.9434e+03	644.7097
Degenerated	1589	1.6828e+03	4.8715e+03

V. Acknowledgement

This work is supported in part by the European Community's Seventh Framework Programme (FP7/2007-2013) under grant #269916 (v-charge) and 4DVideo ERC Starting Grant Nr. 210806.

References

[1] M. Cummins and P. Newman, "Appearance-only slam at large scale with fab-map 2.0," *IJRR*, vol. 30, no. 9, pp. 1100–1123, August 2011.
[2] D. Galvez-Lopez and J. D. Tardos, "Real-time loop detection with bags of binary words," in *IROS*, September 2011, pp. 51 –58.
[3] N. Sünderhauf and P. Protzel, "BRIEF-Gist - Closing the loop by simple means," in *IROS*, 2011, pp. 1234–1241.
[4] D. Galvez-Lopez and J. D. Tardos, "Bags of binary words for fast place recognition in image sequences," *IEEE Transactions on Robotics*, vol. 28, no. 5, pp. 1188–1197, October 2012.
[5] G. H. Lee, F. Fraundorfer, and M. Pollefeys, "Structureless pose-graph loop-closure with a multi-camera system on a self-driving car," in *IROS*, 2013.
[6] D. Nistér and H. Stewénius, "Scalable recognition with a vocabulary tree," in *CVPR*, 2006, pp. 2161–2168.
[7] H. Bay, A. Ess, T. Tuytelaars, and L. Van Gool, "Speeded-up robust features (surf)," *CVIU*, vol. 110, no. 3, pp. 346–359, June 2008.
[8] M. A. Fischler and R. C. Bolles, "Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography," *Communications of the ACM*, vol. 24, no. 6, pp. 381–395, June 1981.
[9] D. Nister, "An efficient solution to the five-point relative pose problem," in *PAMI*, vol. 26, no. 6, 2004, pp. 756–770.
[10] R. I. Hartley and A. Zisserman, *Multiple View Geometry in Computer Vision*, 2nd ed. Cambridge University Press, ISBN: 978-0-521-54051-3, 2004.
[11] S. N. Berthold K. P. Horn, Hugh M. Hilden, "Closed-form solution of absolute orientation using unit quaternions," *Journal of the Optical Society of America A*, 1987.
[12] G. H. Lee, B. Li, M. Pollefeys, and F. Fraundorfer, "Minimal solutions for pose estimation of a multi-camera system," in *ISRR*, 2013.
[13] R. P. Brent, *Algorithms for Minimization without Derivatives*. Englewood Cliffs, NJ: Prentice-Hall, ISBN: 0-13-022335-2, 1973.
[14] G. H. Lee, F. Faundorfer, and M. Pollefeys, "Robust pose-graph loop-closures with expectation-maximization," in *IROS*, 2013.
[15] M. Smith, I. Baldwin, W. Churchill, R. Paul, and P. Newman, "The new college vision and laser data set," *IJRR*, vol. 28, no. 5, pp. 595–599, May 2009.
[16] D. Nistér, O. Naroditsky, and J. R. Bergen, "Visual odometry for ground vehicle applications," *JFR*, vol. 23, no. 1, pp. 3–20, 2006.

B Joint 3D Scene Reconstruction and Class Segmentation

Authors: Häne, Zach, Cohen, Angst, Pollefeys

Link to Table of Contents.

Joint 3D Scene Reconstruction and Class Segmentation

Christian Häne[1], Christopher Zach[2], Andrea Cohen[1], Roland Angst[1*], Marc Pollefeys[1]

[1]ETH Zürich, Switzerland
{chaene, acohen, rangst, pomarc}@inf.ethz.ch

[2]Microsoft Research Cambridge, UK
chzach@microsoft.com

Abstract

Both image segmentation and dense 3D modeling from images represent an intrinsically ill-posed problem. Strong regularizers are therefore required to constrain the solutions from being 'too noisy'. Unfortunately, these priors generally yield overly smooth reconstructions and/or segmentations in certain regions whereas they fail in other areas to constrain the solution sufficiently. In this paper we argue that image segmentation and dense 3D reconstruction contribute valuable information to each other's task. As a consequence, we propose a rigorous mathematical framework to formulate and solve a joint segmentation and dense reconstruction problem. Image segmentations provide geometric cues about which surface orientations are more likely to appear at a certain location in space whereas a dense 3D reconstruction yields a suitable regularization for the segmentation problem by lifting the labeling from 2D images to 3D space. We show how appearance-based cues and 3D surface orientation priors can be learned from training data and subsequently used for class-specific regularization. Experimental results on several real data sets highlight the advantages of our joint formulation.

1. Introduction

Even though remarkable progress has been made in recent years, both image segmentation and dense 3D modeling from images remain intrinsically ill-posed problems. The standard approach to address this ill-posedness is to regularize the solutions by introducing a respective prior. Traditionally, the priors enforced in image segmentation approaches are stated entirely in the 2D image domain (e.g. a contrast-sensitive spatial smoothness assumption), whereas priors employed for image-based reconstruction typically yield piece-wise smooth surfaces in 3D as their solutions. In this paper we demonstrate that joint image segmentation and dense 3D reconstruction is beneficial for both tasks. While the advantages of a joint formulation for segmentation and 3D reconstruction have already been observed and utilized in the literature, our main contribution is the introduction of a rigorous mathematical framework to formulate and solve this joint optimization task. We extend volumetric scene reconstruction methods, which segment a volume of interest into occupied and free-space regions, to a multi-label volumetric segmentation framework assigning object classes or a free-space label to voxels. On the one hand, such a joint approach is highly beneficial since the associated appearance (and therefore a likely semantic category) of surface elements can influence the spatial smoothness prior. Thus, a class-specific regularizer guided by image appearances can adaptively enforce spatial smoothness and preferred orientations of 3D surfaces. On the other hand, densely reconstructed models induce image segmentations which are guaranteed to correspond only to geometrically meaningful objects in 3D. Hence, the segmentation results are trivially consistent across multiple images.

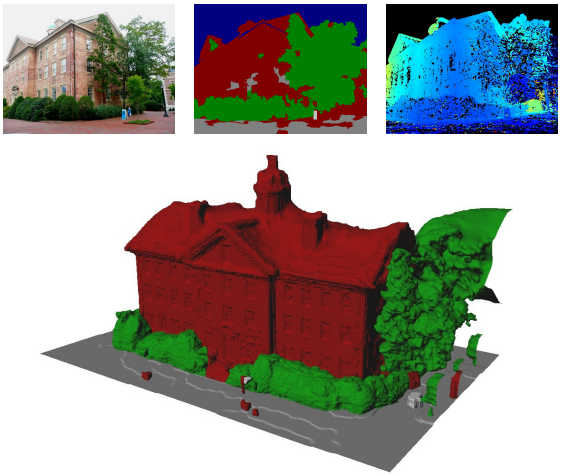


Figure 1: Top: Example of input image, standard image classification result, depthmap. Bottom: Our proposed joint optimization combines class segmentation and geometry resulting in an accurately labeled 3D reconstruction

In a nutshell, we propose to learn appearance likelihoods and class-specific geometry priors for surface orientations

*Now at Stanford University

from training data in an initial step. These data-driven priors can then be used to define unary and pairwise potentials in a volumetric segmentation framework, complementary to the measured evidence acquired from depth maps. While optimizing over the label assignment in this volume, the image-based appearance likelihoods, depth maps from computational stereo, and geometric priors interact with each other yielding an improved dense reconstruction and labeling. The remainder of the paper explains each step in detail, and our mathematical framework is verified on several challenging real-world data sets.

2. Related Work

There is a vast literature on dense 3D modeling from images. Here we sketch only a small subset of related literature, and refer e.g. to the Middlebury MVS evaluation page [20] for a broader survey. Given a collection of depth images (or equivalently densely sampled oriented 3D points) the methods proposed in [13, 27, 23] essentially utilize the surface area as regularization prior, and obtain the final surface representation indirectly via volumetric optimization. One main difference between [13] and [27, 23] is the utilization of a combinatorial graph-cut formulation in the former, whereas [27, 23] employ a continuously inspired numerical scheme. The regularization prior in these works is isotropic, i.e. independent of the surface normal (up to the impact of the underlying discretization), corresponding to a total variation (TV) regularizer in the volumetric representation. The work of [10] utilizes an anisotropic TV prior for 3D modeling in order to enforce the consistency of the surface normals with a given normal field, thus better preserving high frequency details in the final reconstruction. All of the above mentioned work on volumetric 3D modeling from images returns solely a binary decision on the occupancy state of a voxel. Hence, these methods are unaware of typical class-specific geometry, such as the normals of the ground plane pointing upwards. These methods are therefore unable to adjust the utilized smoothness prior in an object- or class-specific way. This observation led to the initial motivation for the present work. More specifically, it is notoriously difficult to faithfully reconstruct weakly or indirectly observed parts of the scene such as the ground, which is usually captured in images at very slanted angles (at least in terrestrial image data). [9] proposes to extend an adaptive volumetric method for surface reconstruction in order not to miss important parts of the scene in the final geometry. The assumption in their method is that surfaces with weak evidence are likely to be real surfaces if adjacent to strongly observed freespace. A key property of our work is that weakly supported scene geometry can be assisted by a class-specific smoothness prior.

If only a single image is considered and direct depth cues from multiple images are not available, assigning object categories to pixels yields crucial information about the 3D scene layout [8, 19], e.g. by exploiting the fact that building facades are usually vertical, and ground is typically horizontal. These relations are generally not manually encoded, but extracted from training data. Such known geometric relations between object categories can also be helpful for 2D image segmentation, e.g. by assuming a particular layout for indoor images [15], a tiered layout [6] or class-specific 2D smoothness priors [22]. Utilizing appearance-based pixel categories and stereo cues in a joint framework was proposed in [11] in order to improve the quality of obtained depth maps and semantic image segmentations. In our work, we also aim on joint estimation of 3D scene geometry and assignment of semantic categories, but use a completely different problem representation—which is intrinsically using multiple images—and solution method. [18, 2] also present joint segmentation and 3D reconstruction methods, but the determined segments correspond to individual objects (in terms of an underlying *smooth* geometry) rather than to semantic categories. Furthermore, a method [1] using semantic information for dense object reconstruction in form of shape priors has been developed concurrently to our work.

3. Joint 3D Reconstruction and Classification

In this section we describe the underlying energy formulation for our proposed joint surface reconstruction and classification framework and its motivation. Similar to previous works on global surface reconstruction we lift the problem from an explicit surface representation to an implicit volumetric one. The increased memory consumption is compensated by the advantages of allowing arbitrary but closed and oriented topology for the resulting surface.

3.1. Continuous Formulation

We cast the ultimate goal of semantically guided shape reconstruction as a volumetric labeling problem, where one out of $L+1$ labels is assigned to each location $z \in \Omega$ in a continuous volumetric domain $\Omega \subset \mathbb{R}^3$. In the following we will use indices i and j for labels. Allowed labels are "free/empty space" (with numeric value 0) and "occupied space" with an associated semantic category (values from $\{1, \dots, L\}$). The label assignments will be encoded with $L+1$ indicator functions $x^i : \Omega \to [0,1]$, $i \in \{0, \dots, L\}$: $x^i(z) = 1$ iff label i is assigned at $z \in \Omega$. Note that in the following, the dependence of all quantities on the 3D location z will be indicated with a subscript to be more consistent with the later discrete formulation, i.e. $x^i(z) = x^i_z$. With this notation in place, the convex relaxation of the labeling problem in a continuous volumetric domain Ω reads as

$$E_{\text{cont}}(\mathbf{x}, \mathbf{y}) = \int_\Omega \sum_i \rho^i_z x^i_z + \sum_{i,j:i<j} \phi^{ij}_z(y^{ij}_z)\, dz, \qquad (1)$$

where $y^{ij} : \Omega \to [-1,1]^3$, $i \in \{0,\dots,L\}$ with $j > i$ are "jump processes" satisfying a modified marginalization constraint

$$\nabla_z x^i = \sum_{j:j>i} y^{ij} - \sum_{j:j<i} y^{ji}. \tag{2}$$

$\rho^i : \Omega \to \mathbb{R}$ encodes the local preference for a particular label. Note that the smoothness term in Eq. 1 is an extension of the standard length/area-based boundary regularizer to Finsler metrics (see e.g. [16]) and the infinitesimal *length functions* $\phi_z^{ij} : \mathbb{R}^3 \to \mathbb{R}_0^+$ are naturally extended from $\mathbb{S}^2$ to $\mathbb{R}^3$, rendering ϕ_z^{ij} a convex and positively 1-homogeneous function. Such choice of ϕ_z^{ij} generalizes the notion of total variation to location and orientation dependent penalization of segmentation boundaries. In addition to the marginalization constraints in Eq. 2, the functions x^i also need to satisfy the normalization constraint, $\sum_i x^i \equiv 1$, and non-negativity constraints. We refer to [25] for a detailed derivation and theoretical interpretation of this energy. A minimizer $(\mathbf{x}, \mathbf{y})$ induces a partition of Ω into free space and respective object categories. The boundaries between the individual regions form the 3D surfaces of interest.

3.2. Discretized Formulation

A disadvantage of this *continuous energy formulation* is that the class of smoothness priors ϕ_s^{ij} is restricted to metrics under reasonable assumptions (see e.g. [12]). Consequently, we focus our attention on discrete lattices (i.e. regular voxel grids) as underlying domain where these restrictions do not apply. Hence, Ω denotes a finite voxel grid with voxels $s \in \Omega$ in the following. A discrete version of the continuous energy in Eq. 1 not requiring a metric prior reads as [24]

$$E_{\text{discr}}(\mathbf{x}) = \sum_{s\in\Omega} \Big(\sum_i \rho_s^i x_s^i + \sum_{i,j:i<j} \phi_s^{ij}(x_s^{ij} - x_s^{ji}) \Big) \tag{3}$$

subject to the following marginalization, normalization and non-negativity constraints,

$$x_s^i = \sum_j (x_s^{ij})_k, \quad x_s^i = \sum_j (x_{s-e_k}^{ji})_k \quad (k \in \{1,2,3\})$$
$$x_s \in \Delta, \qquad x_s^{ij} \geq 0. \tag{4}$$

$e_k \in \mathbb{R}^3$ denotes the k-th canonical basis vector and $(\cdot)_k$ is the k-th component of its argument. The discrete marginalization constraints above follow from Eq. 2 by employing a forward finite difference scheme for the spatial gradient. The probability simplex of appropriate dimension is denoted by Δ. The variables appearing in Eq. 3 have the following interpretation in the context of joint surface reconstruction and segmentation tasks:

- x_s^i encodes whether label i (i.e. free space or one of the solid object categories) is assigned at voxel s,
- $x_s^{ij} - x_s^{ji} \in [-1,1]^3$ represents the local surface orientation if it is non-zero,
- ρ_s^i is the unary data term encoding the measured evidence, i.e. the preference of voxel s for a particular label i. This data term captures the evidence from two sources: firstly, the measurements from a set of depth maps, and secondly, appearance-based classification scores from the input images as obtained from previously trained classifiers. Section 4 describes in detail how this unary term is modeled.
- Finally, ϕ_s^{ij} is the location and direction-dependent smoothness prior indicating the local compatibility of a boundary between label i and j. Hence, these priors encode the previously mentioned class-specific geometric priors. Of highest importance is the directly observable boundary between free space and any of the object categories. Modeling ϕ_s^{ij} from training data is explained in Section 5.

We will restrict ourselves to homogeneous priors in the following, i.e. the local smoothness contribution $\phi_s^{ij}(x_s^{ij} - x_s^{ji})$ does not depend on s, and the objective in Eq. 3 slightly simplifies to

$$E_{\text{discr}}(\mathbf{x}) = \sum_{s\in\Omega} \Big(\sum_i \rho_s^i x_s^i + \sum_{i,j:i<j} \phi^{ij}(x_s^{ij} - x_s^{ji}) \Big). \tag{5}$$

The rationale behind the spatial homogeneity assumption is that only the orientation of a boundary surface and the affected labels are of importance, but not the precise location.

Once the values of ρ_s^i are determined and the smoothness priors ϕ^{ij} are known, the task of inference is to return an optimal volumetric labeling. Since we employ a convex problem stated in Eq. 3, any convex optimization suitable for non-smooth programs can be utilized. After introducing Lagrange multipliers for the constraints and after biconjugation of the smoothness term we are able to directly apply the primal-dual algorithm proposed in [4]. We briefly outline the numerical scheme used in our experiments in the supplementary material.

4. The Ray Likelihood and Its Approximation

In this section we describe how available depth maps (with potentially missing depth values) and appearance-based class likelihoods are converted into respective unaries ρ for joint volumetric reconstruction and classification as described in the previous section. A completely sound graphical model relating image observations with occupancy states of 3D voxels requires observation likelihoods corresponding to clique potentials with entire rays in 3D forming cliques (e.g. [14]). In the following we argue that—under suitable smoothness assumptions on the solution—we can approximate the higher-order clique potentials by suitable unary

ones. We aim on factorizing the clique potential into only unary terms such that the induced (higher-order) cost of a particular boundary surface is approximated by the unaries. Additionally we employ the usual assumption of independence of observations across images. This means, that the unary potentials described below based on color images (and associated depth maps) are accumulated over all images to obtain the effective unary potentials.

In the following we consider a particular pixel p in one of the input images (respectively depth maps, since we assume that depth images use color images as their reference views). The pixel p induces a ray in 3D space, which leads to a set of traversed voxels $s \in \text{ray}(p)$ and the corresponding latent variables x_s^i and their associated unary potentials ρ_s^i. Recall that i indexes one of the $L+1$ semantic categories $\{0, 1, \ldots, L\}$ with 0 corresponding to sky (i.e. free space) and i indicating object category i, respectively. Our task is to (approximately) model the likelihoods

$$P\left(\hat{d}(p), \hat{A}(p) \mid \{x_s^i\}_{s \in \text{ray}(p)}\right), \tag{6}$$

where $\hat{d}(p)$ is the observed depth at pixel p (which may be missing), and $\hat{A}(p)$ encapsulates the local image appearance in the neighborhood of p. Note that in terms of a graphical model the respective potential, $-\log P\big(\hat{d}, \hat{A} \mid \{x_s^i\}_{s \in \text{ray}(p)}\big)$, depends on the entire clique $\{s : s \in \text{ray}(p)\}$. Clearly, for a particular ray the likelihoods of observing $\hat{d}(p)$ and $\hat{A}(p)$ only depend on the first crossing from freespace to occupied space. Nevertheless, proper handling of voxel visibility links all voxels along the ray to form a clique.

For notational convenience we will drop the dependence on the pixel p, and also index voxels along ray(p) by their depth with respect to the current view. We will substantially simplify the potentials (and therefore the inference task) by considering the following cases:

Observed depth: This is the case when $\hat{d}$ in the depth map is valid (i.e. not missing). In this case we assume that Eq. 6 factorizes into

$$P\big(\hat{d}, \hat{A} \mid \text{voxel } d \text{ is first crossing to } i\big) = P(\hat{d} \mid d) P(\hat{A} \mid i),$$

where $P(\hat{d} \mid d)$ captures the noise for inliers in the depth sensing process and is usually a monotonically decreasing function of $|d - \hat{d}|$. $P(\hat{A} \mid i)$ is induced by the confidence of an appearance-based classifier for object category i.

We only define non-zero unaries for voxels along the ray near the observed depth. Assume that the inlier noise of depth estimation is bounded by δ, and we denote by $\hat{d} \pm \delta$ the voxels along the ray with distance δ and $-\delta$, respectively. We set the unary potentials

$$\rho^i_{\hat{d}+\delta} = \sigma_{\text{class } i} \qquad \rho^i_d = \begin{cases} 0 & \text{for } i = 0 \\ \eta(\hat{d} - d) & \text{for } i > 0. \end{cases} \tag{7}$$

for voxels d near the observed depth, i.e. voxels d closer to $\hat{d}$ than δ. Here $\sigma_{\text{class } i} = -\log P(\hat{A} \mid i)$ (with class 0 corresponding to sky). The function $\eta : [-\delta, \delta] \to \mathbb{R}$ is independent of the object category i and reflects the noise assumptions of $\hat{d}$. We choose $\eta(\hat{d} - d) = \beta \operatorname{sgn}(\hat{d} - d)$ for $\beta > 0$, corresponding to an exponentially distributed noise for depth inliers. Inserting unaries only near the observed depth corresponds to truncating the cost function, hence we assume exponentially distributed inliers and uniformly distributed outlier depth values. See Fig. 2 for an illustration of unaries along the ray.

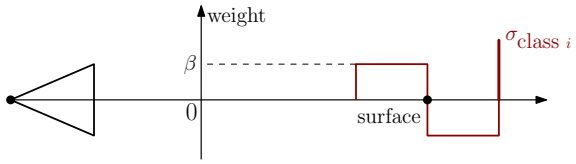


Figure 2: Unaries assigned to voxels along a particular line-of-sight.

Since we enforce spatial smoothness of the labeling (i.e. multiple crossings within the narrow band near $\hat{d}$ are very unlikely), we expect three possible configurations for voxels in $[\hat{d} - \delta, \hat{d} + \delta]$ described below. For each configuration we state the contribution of unary terms for the particular ray to the complete energy.

1. In the labeling of interest we have that free-space transitions to a particular object class i at depth d. Hence, $x_s^0 = 1$ for $s \in [\hat{d} - \delta, d)$ and $x_s^i = 1$ for $s \in [d, \hat{d} + \delta]$. Summing the unaries according to Eq.7 over the voxels in $[\hat{d} - \delta, \hat{d} + \delta]$ yields

$$\sigma_{\text{class } i} + \sum_{d' \in [d, \hat{d}+\delta]} \eta(\hat{d} - d'),$$

   i.e. the negative log-likelihood of observing the appearance category i in the image and the one corresponding to the depth noise assumption. Note that the second term, $\sum_{d'} \eta(\hat{d} - d')$ will be non-positive and therefore lower the overall energy. This beneficial term is not appearing in the other cases below.

2. If all voxels in the particular range $[\hat{d} - \delta, \hat{d} + \delta]$ are freespace ($x_s^0 = 1$ for all the voxels in this range), then the contribution to the total energy is just σ_{sky}. Since a potential transition to a solid object class outside the near band is not taken into account, this choice of unary potentials implicitly encodes the assumption that that freespace near the observed depth implies freespace along the whole ray.

3. All voxels in the range are assigned to object label i (i.e. $x_s^i = 1$ for $s \in [\hat{d} - \delta, \hat{d} + \delta]$. This means that there

was a transition from freespace to object type i earlier along the ray. Thus, the contribution to the energy is $\sigma_{\text{class } i}$ in this case.

Overall, our choice of unaries will faithfully approximate the desired true data costs in most cases. Since camera centers are in free-space by definition, we add a slight bias towards free-space along the line-of-sight from the respective camera center to the observed depth (i.e. voxels in the range $[0, \hat{d} - \delta]$). This has also a positive influence on the convergence speed.

Missing depth: If no depth was observed at a particular pixel p, we cannot assign unaries along the corresponding ray. Since missing depth values mostly occur in the sky regions of images, we found the following modification helpful to avoid "bleeding" of buildings etc. beyond their respective silhouettes in the image: in case of missing depth we set the unary potentials to

$$\rho_s^0 = \min\left\{0, \sigma_{\text{sky}} - \min_{i \neq \text{sky}} \sigma_i\right\} \tag{8}$$

and $\rho_s^i = 0$ for $i > 0$ for all voxels s along ray(p). This choice of unaries favors freespace along the whole ray whenever depth is missing and sky is the most likely class label in the image.

5. Training the Priors

In this section, we will explain how the appearance likelihoods used in the unary potentials ρ_s^i and the class-specific geometric priors ϕ^{ij} are learned from training data. While the appearance terms are based on classification scores of a standard classifier, training of geometric priors from labeled data is more involved. We first start describing the training of the appearance likelihoods before discussing the training procedure for smoothness priors.

5.1. Appearance Likelihoods

In order to get classification scores for the labels in the input images we train a boosted decision tree classifier [7] on manually labeled training data. In a first step, the training images are segmented into super-pixels using the mean shift segmentation algorithm[1]. Features are extracted for each super-pixel. We use the default parameters as implemented by [7], resulting in 225 dimensional feature vectors based on color, intensity, geometry, texture and location. It should be noted that the geometry and location features are extracted by using 2-D information on the images (superpixel size, shape, and relative position in the image) and they are not related to the 3-D geometry of the scene. The extracted features and ground truth annotations are fed into the boosted decision tree. The classifier is trained over 5 classes: sky, building, ground, vegetation, and clutter. We use 2 splits per decision tree and 200 boosting rounds. We designed a training dataset by taking 76 images from the CamVid dataset [3] and 101 images from the MSRC dataset. We also added 34 images taken at street level of different buildings. These buildings are not part of the evaluation data set.

Once the classifier is trained, it can be used to obtain scores for each region of the input images. These scores represent the log-likelihoods of each class for each region of the image.

[1] OpenCV implementation

5.2. Class-Specific Geometric Priors

We use a parametric model for the functions ϕ_s^{ij} appearing in the smoothness term of Eq. 5. As already mentioned we restrict ourselves to spatially homogeneous functions $\phi_s^{ij} = \phi^{ij}$, and thus there is no dependency on the location s. Note that the energy formulation in Eq. 5 naturally corresponds to a negative log-probability. Hence, the functions ϕ^{ij} will be also interpreted as negative log-probabilities. Let $s_{i \leftrightarrow j}$ denote a transition event between labels i and j at some voxel s, and let n_s^{ij} be the (unit-length) boundary normal at this voxel. Instead of modeling ϕ^{ij} directly, we use

$$P(n_s^{ij}) = P(n_s^{ij} \mid s_{i \leftrightarrow j}) P(i \leftrightarrow j), \tag{9}$$

where we applied the homogeneity assumption, i.e. $P(s_{i \leftrightarrow j}) = P(i \leftrightarrow j)$. The conditional probability, $P(n_s^{ij} \mid s_{i \leftrightarrow j})$ is now modeled as a Gibbs probability measure

$$P(n_s^{ij} \mid s_{i \leftrightarrow j}) = \exp\left(-\psi^{ij}(n_s^{ij})\right) / Z^{ij}, \tag{10}$$

for a pos. 1-homogeneous function ψ^{ij}. Z^{ij} is the respective partition function, $Z^{ij} \stackrel{\text{def}}{=} \int_{n \in \mathbb{S}^2} \exp\left(-\psi^{ij}(n^{ij})\right) dn$, and $\mathbb{S}^2$ is the 3-dimensional unit sphere. Consequently, ϕ^{ij} in Eq. 5 is now given by

$$\phi^{ij}(n) = \psi^{ij}(n) + \log Z^{ij} - \log P(i \leftrightarrow j) \tag{11}$$

for a unit vector $n \in \mathbb{S}^2$. Maximum-likelihood estimation is used to fit the parameters to available training data, formally

$$\theta = \arg\max_{\theta} \prod_s \prod_{i,j} P(n_s^{ij} \mid s_{i \leftrightarrow j}) P(i \leftrightarrow j), \tag{12}$$

where the product goes over all training samples s and ψ^{ij} and Z^{ij} are functions of the parameters θ^{ij} which are gathered in $\theta = \{\theta^{ij} \mid i, j \in \{0, \ldots, L\}\}$. In our implementation, we estimate the discrete probabilities $P(i \leftrightarrow j)$ of observing a transition $i \leftrightarrow j$ upfront by counting the relative frequencies $N^{ij} / \sum_{i,j} N^{ij}$ of the respective type of boundaries from training data. Estimating first $P(i \leftrightarrow j)$ has the advantage that the ML-estimation in Eq. 12 decouples into independent estimation problems of the form

$$\theta^{ij} = \arg\min_{\theta^{ij}} \sum_{k=1}^{N^{ij}} \psi^{ij}(n_k^{ij}; \theta^{ij}) + N^{ij} Z^{ij}(\theta^{ij}), \tag{13}$$

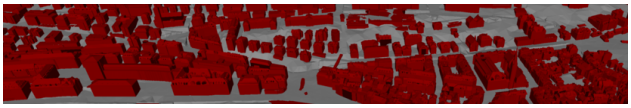


Figure 3: A section of the cadastral city model used to train the geometric priors.

where the summation goes over all the N^{ij} transition samples n_k^{ij} between labels i and j. Since for many choices of ψ^{ij} the partition function cannot be solved analytically, we use Monte Carlo integration to obtain an estimate for Z^{ij}. Given the low dimensionality of θ^{ij} (up to 4 components, see Section 5.3 below) and the necessity of Monte Carlo integration for the partition function, we use a simple grid search to find an approximate minimizer θ^{ij}. As training data we use a three dimensional cadastral city model (see Fig. 3) which enables us to train ψ^{ij} for the transitions ground $\leftrightarrow$ free space, ground $\leftrightarrow$ building and building $\leftrightarrow$ free space. Label transitions unobserved in the training data are defined manually. At this point we need to address two small technical issues:

Remark 1. ϕ^{ij} is only specified for unit vectors $n \in \mathbb{S}^2$, but the argument in the energy model Eq. 5 are usually non-normalized gradient directions $y_s^{ij} \stackrel{\text{def}}{=} x_s^{ij} - x_s^{ji} \in [-1,1]^3$. However, remember that ψ^{ij} is a convex and positively 1-homogeneous function. Together with the fact that the area of the surface element in finite difference discretizations is captured exactly by $\|y_s^{ij}\|_2$, we derive the contribution of y_s^{ij} to the regularizer as

$$\|y_s^{ij}\|_2 \phi^{ij}\left(y_s^{ij}/\|y_s^{ij}\|_2\right) = \phi^{ij}(y_s^{ij})$$

by the 1-homogeneity of ϕ^{ij}. Therefore, the extension of ϕ^{ij} as given in Eq. 11 to arbitrary arguments $y \in \mathbb{R}^3$ is

$$\phi^{ij}(y) = \psi^{ij}(y) + \|y\|_2 \underbrace{\left(\log Z^{ij} - \log P(i \leftrightarrow j)\right)}_{\stackrel{\text{def}}{=} C^{ij}}. \quad (14)$$

Consequently, our smoothness prior ϕ^{ij} will always be composed of an anisotropic, direction-dependent component ψ^{ij} and an isotropic contribution proportional to $C^{ij} = \log Z^{ij} - \log P(i \leftrightarrow j)$. This also implies that there is no need to explicitly model any isotropic component in ψ^{ij}.

Remark 2. The function ϕ^{ij} given in Eq. 14 above is positively 1-homogeneous if ψ^{ij} is, but convexity can only be guaranteed whenever $C^{ij} = \log Z^{ij} - \log P(i \leftrightarrow j) \geq 0$ or $P(i \leftrightarrow j) \leq Z^{ij}$. This is in practice not a severe restriction, since for a sufficiently fine discretization of the domain the occurrence of a boundary surface is a very rare event and therefore $P(i \leftrightarrow j) \ll 1$.

5.3. Choices for ψ^{ij}

We need to restrict ψ^{ij} to be convex and positively 1-homogeneous. One option is to parametrize $\psi^{ij}(n) = \psi^{ij}(n;\theta)$ in the primal and to limit θ such that the resulting ψ^{ij} has these properties, but this may be difficult in general. We choose a slightly different route and parametrize the convex conjugate of ψ^{ij}, $\left(\psi^{ij}\right)^*$,

$$\left(\psi^{ij}\right)^*(p) = \max_n \left\{p^T n - \psi^{ij}(n)\right\} = \imath_{W_{\psi^{ij}}}(p),$$

i.e. the indicator function for a (convex) shape $W_{\psi^{ij}}$ (which will be called a *Wulff shape* [17] in the following). We find it easier to model parametric convex Wulff shapes $W_{\psi^{ij}}$ rather than ψ^{ij} directly. Below we describe the utilized Wulff shapes and its parametrizations. Which Wulff shape is picked for ψ^{ij} (in addition to its continuous parameters) is part of the ML estimation. The description below is for Wulff shapes in a canonical position, since any ψ induced by a rotated shape can be expressed using a canonical one,

$$\psi(n;R) = \max_{p \in R \cdot W_\psi} p^T n = \max_{p \in W_\psi} (Rp)^T n = \psi(R^T n; I).$$

Given remark 1 above there is no need to model the Wulff shape with an isotropic and an anisotropic component (i.e. as Minkowski sum of a sphere and some other convex shape).

The Wulff shapes described below are designed to model two frequent surface priors encountered in urban environments: one prior favors surface normals that are in alignment with a specific direction (e.g. ground surface normals prefer to be aligned with the vertical direction), and the second Wulff shape favors surface normals orthogonal to a given direction (such as facade surfaces having generally normals perpendicular to the vertical direction). In order to obtain a discriminative prior we assume that an approximate vertical direction is provided. We refer to the supplementary material for graphical illustrations of the Wulff shapes and induced smoothness costs.

Line Segment This Wulff shape has only one parameter l and is a line segment in z-direction centered at the origin with length $2l$ (i.e. its endpoints are $(0,0,l)^T$ and $(0,0,-l)^T$). This shape translates to a function $\psi(n) = l|n_3|$, which is convex as long $l \geq 0$.

Half-sphere plus spherical cap This Wulff shape W_ψ consists of a half-sphere with radius r centered at the origin in opposition to a spherical cap with height h. The corresponding function ψ favors directions pointing upwards and isotropically penalizes downward pointing normals. ψ can be computed in closed form (with $n = (n_1, n_2, n_3)^T$),

$$\psi(n) = \begin{cases} r\|n\| & \text{if } n_3 \leq 0 \\ \|n\|\left(\frac{r^2}{2h} + \frac{h}{2}\right) - n_3\left(\frac{r^2}{2h} - \frac{h}{2}\right) & \text{if } (*) \\ r\left\|\binom{n_1}{n_2}\right\| & \text{otherwise,} \end{cases}$$

where $(*)$ is $n_3 > 0$ and $n_3(h^2 + r^2) > (r^2 - h^2)/\|n\|$. By construction W_ψ is convex (and therefore also ψ) as long as $r \geq 0$ and $h \in [0, r]$.

6. Experiments

In this section we present the results obtained on four challenging real world datasets. We compare our geometry to a standard volumetric fusion (in particular "TV-Flux" [23]) and also illustrate the improvement of the class segmentation compared to a single image best-cost segmentation.

We use the dataset castle P-30 from [21] and three additional urban datasets ranging from 127 to 195 images in the dataset size. Camera poses where obtained with the publicly available structure from motion pipeline [26]. The depth maps are computed using plane sweep stereo matching for each of the images with zero mean normalized cross correlation (ZNCC) matching costs. Up to nine images are matched to the reference view simultaneously with best K occlusion handling. To get rid of the noise the raw depth maps are filtered by discarding depth values with a ZNCC matching score above 0.4. The class scores are obtained by using the boosted decision tree classifier explained in Section 5.1. To align the voxel grid with the scene we use the approach described in [5]. We use a multi-threaded C++ implementation to find a minimizer of Eq. 5 (running on a 48 cores).

Fig. 4 illustrates the results for all 4 datasets. As expected, computational stereo in particular struggles with faithfully capturing the ground, which is represented by relatively few depth samples. Consequently, depth integration methods with a generic surface prior such as TV-Flux easily remove the ground and other weakly observed surfaces (due to the well-known shrinking bias of the employed boundary regularizer). In contrast, our proposed joint optimization leads to more accurate geometry, and at the same time image segmentation is clearly improved over a greedy best-cost class assignment.

The third column in Fig. 4 illustrates that the most probable class labels according to the trained appearance likelihoods especially confuses ground, building, and clutter categories. Fusing appearance likelihood over multiple images and incorporating the surface geometry almost perfectly disambiguates the assigned object classes. The joint determination of the right smoothness prior also enables our approach to fully reconstruct ground and all the facades as seen in Fig. 4, 4th column. The ground is consistently missing in the TV-Flux results, and partially the facades and roof structure suffer from the generic smoothness assumption Fig. 4, 5th column). We selected a weighting between data fidelity and smoothness in the TV-Flux method such that successfully reconstructed surfaces have a (visually) similar level of smoothness than the results of our proposed method.

7. Conclusion

We present an approach for dense 3D scene reconstruction from multiple images and simultaneous image segmentation. This challenging problem is formulated as joint volumetric inference task over multiple labels, which enables us to utilize class-specific smoothness assumptions in order to improve the quality of the obtained reconstruction. We use a parametric representation for the respective smoothness priors, which yields a compact representation for the priors and—at the same time—allows to adjust the underlying parameters from training data. We demonstrate the benefits of our approach over standard smoothness assumptions for volumetric scene reconstruction on several challenging data sets.

Future work needs in particular to address the scalability of the method. As a volumetric approach operating in a regular voxel grid, our method shares the limitations in terms of spatial resolution with most other volumetric approaches. Adaptive representations for volumetric data can be a potential solution. We also plan to extend the number of object categories to obtain a finer-grained segmentation. Note that not all pairwise transitions between labels in 3D are equally important or even occur in practice. This fact can be utilized to improve the computational efficiency of our proposed formulation.

Acknowledgements: We thank Carsten Rother and Pushmeet Kohli for initial discussions, and Arnold Irschara for providing dataset 3. Furthermore we gratefully acknowledge the support of the 4DVideo ERC starting grant #210806 and V-Charge grant #269916 both under the EC's FP7/2007-2013. Roland Angst was a recipient of the Google Europe Fellowship.

References

[1] Y. Bao, M. Chandraker, Y. Lin, and S. Savarese. Dense object reconstruction with semantic priors. In *Proc. CVPR*, 2013.

[2] M. Bleyer, C. Rother, P. Kohli, D. Scharstein, and S. Sinha. Object stereo-joint stereo matching and object segmentation. In *Proc. CVPR*, pages 3081–3088, 2011.

[3] G. Brostow, J. Fauqueur, and R. Cipolla. Semantic object classes in video: A high-definition ground truth database. *Pattern Recognition Letters*, 30(2):88–97, 2009.

[4] A. Chambolle and T. Pock. A First-Order Primal-Dual Algorithm for Convex Problems with Applications to Imaging. *J. Math. Imag. Vision*, pages 1–26, 2010.

[5] A. Cohen, C. Zach, S. Sinha, and M. Pollefeys. Discovering and exploiting 3d symmetries in structure from motion. In *Proc. CVPR*, 2012.

[6] P. F. Felzenszwalb and O. Veksler. Tiered scene labeling with dynamic programming. In *Proc. CVPR*, pages 3097–3104, 2010.

[7] S. Gould, O. Russakovsky, I. Goodfellow, P. Baumstarck, A. Y. Ng, and D. Koller. The STAIR Vision Library. http://ai.stanford.edu/~sgould/svl, 2010.

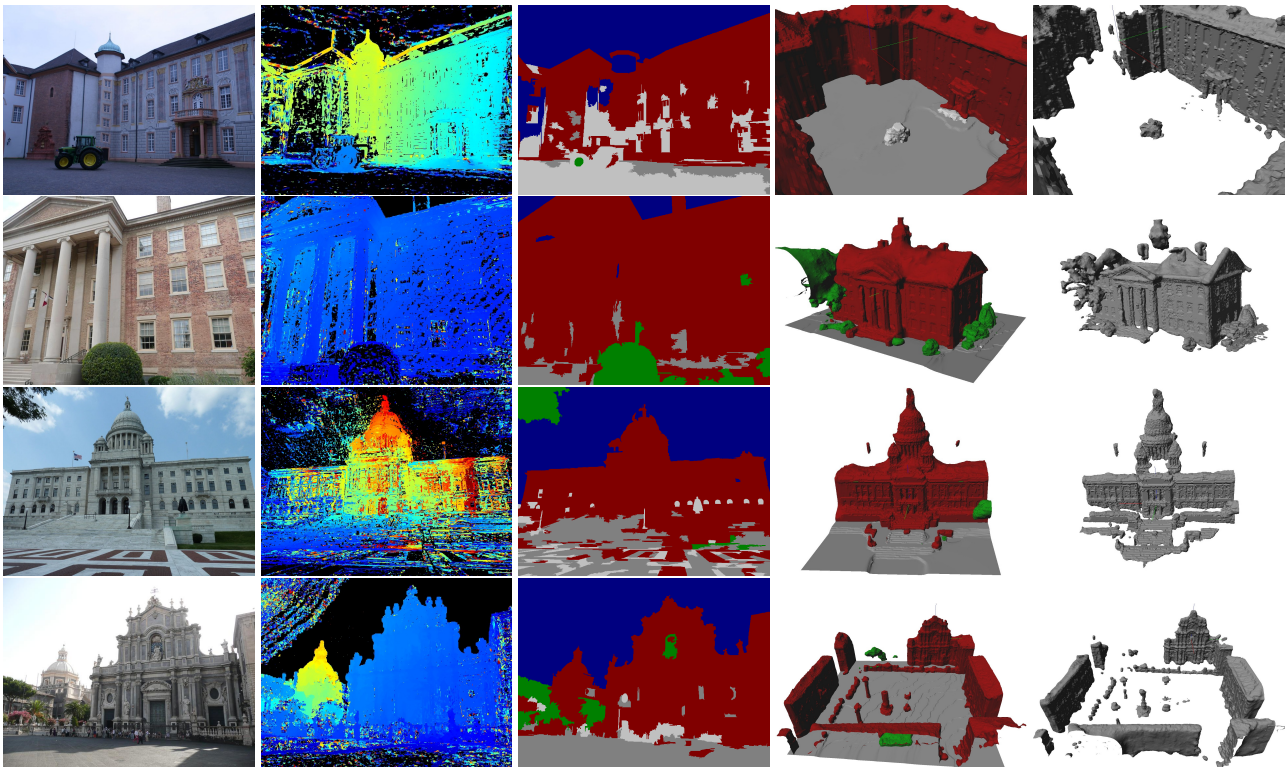


Figure 4: Results for 4 datasets. First row: castle-P30 [21]. Second row to last: Datasets 1 to 3. From left to right: Example input images, example depth map, raw image labeling, our result, tv-flux fusion result; The different class labels are depicted using the following color scheme: building → red, ground → dark gray, vegetation → green, clutter → light gray.

[8] D. Hoiem, A. Efros, and M. Hebert. Recovering surface layout from an image. *IJCV*, 75(1):151–172, 2007.

[9] M. Jancosek and T. Pajdla. Multi-view reconstruction preserving weakly-supported surfaces. In *Proc. CVPR*, 2011.

[10] K. Kolev, T. Pock, and D. Cremers. Anisotropic minimal surfaces integrating photoconsistency and normal information for multiview stereo. In *Proc. ECCV*, pages 538–551, 2010.

[11] L. Ladický, P. Sturgess, C. Russell, S. Sengupta, Y. Bastanlar, W. Clocksin, and P. Torr. Joint optimisation for object class segmentation and dense stereo reconstruction. In *Proc. BMVC*, pages 104.1–11, 2010.

[12] J. Lellmann and C. Schnörr. Continuous multiclass labeling approaches and algorithms. *SIAM Journal on Imaging Sciences*, 4(4):1049–1096, 2011.

[13] V. Lempitsky and Y. Boykov. Global optimization for shape fitting. In *Proc. CVPR*, 2007.

[14] S. Liu and D. Cooper. A complete statistical inverse ray tracing approach to multi-view stereo. In *Proc. CVPR*, pages 913–920, 2011.

[15] X. Liu, O. Veksler, and J. Samarabandu. Order-preserving moves for graph-cut-based optimization. *IEEE Trans. Pattern Anal. Mach. Intell.*, 32:1182—-1196, 2010.

[16] J. Melonakos, E. Pichon, S. Angenent, and A. Tannenbaum. Finsler active contours. *IEEE Trans. Pattern Anal. Mach. Intell.*, 30(3):412–423, 2008.

[17] S. Osher and S. Esedoglu. Decomposition of images by the anisotropic Rudin-Osher-Fatemi model. *Comm. Pure Appl. Math.*, 57:1609–1626, 2004.

[18] L. Quan, J. Wang, P. Tan, and L. Yuan. Image-based modeling by joint segmentation. *IJCV*, 75(1):135–150, 2007.

[19] A. Saxena, S. Chung, and A. Ng. 3-d depth reconstruction from a single still image. *IJCV*, 76(1):53–69, 2008.

[20] S. Seitz, B. Curless, J. Diebel, D. Scharstein, and R. Szeliski. A comparison and evaluation of multi-view stereo reconstruction algorithms. In *Proc. CVPR*, pages 519–526, 2006.

[21] C. Strecha, W. von Hansen, L. V. Gool, P. Fua, and U. Thoennessen. On benchmarking camera calibration and multi-view stereo for high resolution imagery. In *Proc. CVPR*, 2008.

[22] E. Strekalovskiy and D. Cremers. Generalized ordering constraints for multilabel optimization. In *Proc. ICCV*, 2011.

[23] C. Zach. Fast and high quality fusion of depth maps. In *Proc. 3DPVT*, 2008.

[24] C. Zach, C. Häne, and M. Pollefeys. What is optimized in convex relaxations for multi-label problems: Connecting discrete and continuously-inspired MAP inference. Technical report, MSR Cambridge, 2012.

[25] C. Zach, C. Häne, and M. Pollefeys. What is optimized in tight convex relaxations for multi-label problems? In *Proc. CVPR*, 2012.

[26] C. Zach, M. Klopschitz, and M. Pollefeys. Disambiguating visual relations using loop constraints. In *Proc. CVPR*, 2010.

[27] C. Zach, T. Pock, and H. Bischof. A globally optimal algorithm for robust TV-L^1 range image integration. In *Proc. ICCV*, 2007.

C Class Specific 3D Object Shape Priors Using Surface Normals

Authors: Häne, Savinov, Pollefeys

Link to Table of Contents.

Class Specific 3D Object Shape Priors Using Surface Normals

Christian Häne, Nikolay Savinov, Marc Pollefeys
ETH Zürich, Switzerland
{chaene, nsavinov, pomarc}@inf.ethz.ch

Abstract

Dense 3D reconstruction of real world objects containing textureless, reflective and specular parts is a challenging task. Using general smoothness priors such as surface area regularization can lead to defects in the form of disconnected parts or unwanted indentations. We argue that this problem can be solved by exploiting the object class specific local surface orientations, e.g. *a car is always close to horizontal in the roof area. Therefore, we formulate an object class specific shape prior in the form of spatially varying anisotropic smoothness terms. The parameters of the shape prior are extracted from training data. We detail how our shape prior formulation directly fits into recently proposed volumetric multi-label reconstruction approaches. This allows a segmentation between the object and its supporting ground. In our experimental evaluation we show reconstructions using our trained shape prior on several challenging datasets.*

1. Introduction

In the last few years a lot of progress has been made in the area of dense 3D reconstruction from images. The main difficulty is imperfect input such as noisy, missing or ambiguous data. Most approaches have in common that they try to find a solution which follows the input data while being sufficiently regular with respect to a given smoothness prior. These priors often penalize surface properties such as *e.g.* area or curvature. Such approaches work well for data which is mostly affected by noise. If it comes to defects such as specularities, translucency or even reflections these methods are prone to produce suboptimal solutions.

To overcome this problem strong domain specific priors are used. These can be priors on semantic classes, *e.g.* building facades have to be vertical, or priors on the structure, *e.g.* the reconstruction has to form a tree. Recently also 3D object shape priors have been proposed.

One powerful approach to dense 3D reconstruction is volumetric depth map fusion, where a set of depth maps is combined into a volumetric representation. The volume is segmented into the exterior (free space) and interior (occupied space) of the object. In this work we propose to exploit anisotropic regularization to embed object shape priors into convex regularization formulations for volumetric depth map fusion. The reasoning behind this is that object classes have locally only a restricted distribution of surface normals over different examples. The outline of a bottle is always close to vertical but on the other hand there can be regions such as the side mirrors of a car where a large set of normals is likely. We capture these normal variations and utilize them to formulate our shape prior.

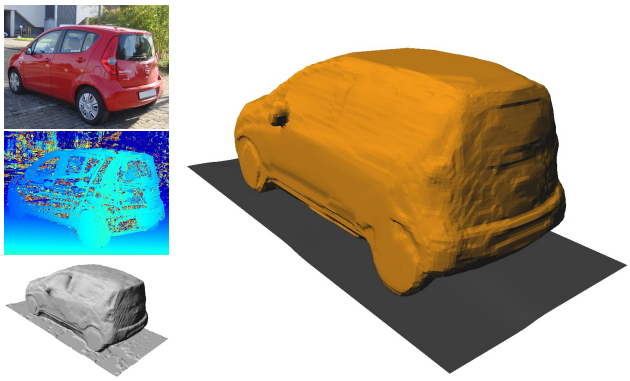


Figure 1. Left: Cropped input image, depth map and baseline result. Right: Result with our shape prior formulation. Object and ground are segmented by our formulation.

1.1. Related Work

There is a broad literature on dense 3D reconstruction from images. We focus on the most related ones to our approach and for a more detailed overview we refer the reader to [15]. The main idea of volumetric depth map fusion is to segment a volume into occupied and free space and extract the surface as boundary between the two regions. The main difference between the various methods is how they regularize the data. Approaches penalizing the surface area have been proposed in the discrete optimization literature [12] and also as continuous convex optimization [22]. The main advantage of such continuously inspired formulations is that they have an underlying continuous representation, which leads to more isotropic behavior compared to discrete approaches using graph-cuts.

Given input images an oriented point cloud can be extracted using structure from motion approaches. Utilizing anisotropic smoothness can then be used to align the final reconstruction with the extracted normal directions leading to a more faithful reconstruction of surface details [11].

While such general smoothness priors lead to faithful reconstructions in the presence of well behaved data they can be too general for very ill-conditioned cases. Therefore other priors have been proposed.

Segmenting elongated structures, such as blood vessels, are prone to be disconnected. To overcome this problem connectivity priors have been introduced [18, 17].

Object shape priors have recently attained much attention in the field of dense 3D reconstruction [3, 5]. Both works try to fit a parametric shape model into input data. This allows the reconstruction of objects which are not well represented in the input data such as highly reflective cars.

Multi-label segmentation methods have only recently emerged. The work of [4] proposes a spatially continuous formulation of multi-label segmentation with a discrete label space. While anisotropic smoothness can be directly applied to this formulation, the main restriction is that the smoothness needs to form a metric over the label space. A comparison of discrete formulations and the discretized version of the continuous formulation revealed that the discretized formulation does not suffer from this problem [20]. The most closely related work to our shape prior formulation is [9]. They propose to introduce semantic information into volumetric 3D reconstruction by utilizing convex multi-label segmentation. The main idea is that the volume is no longer segmented into free and occupied space but rather into multiple semantic classes. This allows the semantic classes to be penalized differently, *e.g*. ground should be less penalized when in a horizontal direction, as this is a more likely direction for ground.

Our approach to shape priors is driven by the observation that, for many real life objects, the local distribution of normals is limited. For example the doors of a car cannot be horizontal but this is a very likely direction for the hood. Therefore, we propose to incorporate a shape prior by locally penalizing the surface directions anisotropically dependent on the distribution of normals given in training data. A more direct approach for volumetric shape priors might be to just derive a location specific prior on the occupancy state of a voxel. However, this does not allow the capturing of that *e.g*. the roof of the car needs to be close to horizontal, which is nicely captured by our direction based prior.

Often the reconstructed object is standing on a supporting ground. To be able to handle the data costs induced by the ground, which do not fit to the shape prior, we propose to use multi-label segmentation to treat the ground differently. This directly enables us to also segment the object and the ground.

1.2. Our Contributions

- We propose a novel way to incorporate shape priors based on surface normal distributions into convex multi-label optimization. The driving force of our method are spatially varying anisotropic smoothness priors (*c.f*. Sec. 2.3). The formulation is based on more general anisotropic smoothness priors as the ones that have been used in recent convex multi-label segmentation approaches.

- An efficient way to derive the shape priors from training data is detailed in Sec. 2.4.

- Due to the multi-label formulation we can segment the object from its supporting ground in a natural way (see Fig. 1).

2. Shape Priors

Our approach is based on recently published convex multi-label segmentation methods [16, 4, 20]. Before we explain how we can incorporate shape priors, we need to summarize the optimization framework.

2.1. Convex Multi-Label Segmentation

The ultimate goal of surface reconstruction is to find the most probable surface which fits the given noisy data. In convex approaches for surface reconstruction, the surface is often implicitly represented in a volumetric form. Traditionally, the volume is separated into two areas, free space and occupied space [12, 19] and the resulting surface is described by the boundary between free and occupied space. An extension of this formulation to multiple semantic classes was proposed in [9]. We follow a similar approach by dividing the reconstruction into the object itself, on which the shape prior should be applied, and the ground, if there is any. This leads to a segmentation of the volume into three regions: free space, object and supporting ground. This allows us to handle the often strongly observed ground which does not fit to the object shape prior. Additionally, this directly gives the segmentation between the reconstructed object and the supporting ground.

The main ingredients of the formulation are: a unary and a smoothness term plus a set of constraints linking them together. The unary potential is computed from the observed data and will be called data term. A smoothness term is required to tackle the noise in the observed data. It can be seen as a prior on the expected surface and is therefore named smoothness prior.

The convex multi-label segmentation formulation presented in the following originates from continuous optimization [4]. The main drawback of the purely continuous formulation is that it does not allow for non-metric smoothness priors. It has been pointed out that this can be overcome by only considering discrete domains in [20]. As

we are inherently bound to the discrete domain and the underlying continuous formulation is not required for the understanding of this manuscript, we directly present the discretized version of the formulation. However some mechanisms need to be understood as a discretization of the underlying continuous formulation.

More formally, a discretized volumetric domain $\Omega \subset \mathbb{R}^3$ is given. The task of the label assignment is to assign one out of the three labels free space, object or supporting ground $\mathcal{L} = \{0, 1, 2\}$ to each location (voxel) $s \in \Omega$. It is represented as soft indicator vectors $x_s \in [0, 1]^{|\mathcal{L}|}$ with $x_s^i = 1$ if and only if label i is assigned at s. In the following, we will use subscripts to denote the spatial position of a variable and superscripts to denote label indices. e_k denotes the k-th canonical basis vector and $(\cdot)_k$ is used to extract the k-th component of a vector. Having this notation in place, we can now state the objective of the utilized convex minimization problem.

$$E(x) = \sum_{\Omega \in s} \left(\sum_i \rho_s^i x_s^i + \sum_{i,j:i<j} \phi_s^{ij}(x_s^{ij} - x_s^{ji}) \right) \quad (1)$$

In addition to the per voxel soft-indicator vectors x_s^i the objective also contains vectors $x_s^{ij} \in [0, 1]^3$. They are understood as follows: the $k-$th component of the vector x_s^{ij} equals 1 if and only if there is a transition from label i to label j along the k-th canonical direction of the grid. The functions $\phi_s^{ij} : \mathbb{R}^3 \to \mathbb{R}_0^+$ define the smoothness prior and need to be convex and positively 1-homogeneous. Their arguments $x_s^{ij} - x_s^{ji}$ are aligned with the surface normal if they are on the segmentation boundary. They are an anisotropic extension of standard surface area based penalization used in continuous segmentation formulations (c.f. [7, 23]). Note that originating from a discretized continuous formulation, the arguments of the objective are generally non-binary around the optimal surface. This is a necessary behavior to allow the vectors $x_s^{ij} - x_s^{ji}$ to point in arbitrary directions, and the main reason why these formulations are less prone to metrication artifacts than purely graph based approaches. The ρ_s^i are the unary data costs giving a local preference at voxel s for the label i. To ensure that the edge indicator variables x_s^{ij} and the node indicator variables x_s^i are consistent and in a valid configuration, the objective in Eq. 1 is subject to the following marginalization, normalization and non-negativity constraints.

$$\begin{aligned} & x_s^i = \sum_j (x_s^{ij})_k, \quad x_s^i = \sum_j (x_{s-e_k}^{ji})_k \quad k \in \{1, 2, 3\} \\ & x_s \in \Delta, \quad x_s^{ji} \geq 0 \end{aligned} \quad (2)$$

The marginalization constraints ensure that the per node and per edge variables agree on the assigned labels. Constraining the per-node variables to lie in the unit probability simplex $\Delta = \{\sum_i x_s^i = 1, x_s^i \geq 0\}$ ensures that exactly one label is assigned at each position s.

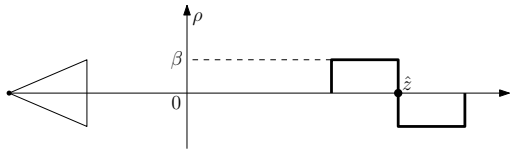


Figure 2. Data term entered into the grid along a viewing ray.

2.2. Data Term

The unary data term gives a local preference for a certain label. In our formulation, we have two different labels denoting occupied space: object and ground. It has been shown that such multi-label segmentation can be done based on per image class scores in [9]. We choose a slightly different approach and do not compute any class specific data term. The segmentation of object and ground is solely based on the shape prior detailed in Sec. 2.3. The only input for the data term is a set of potentially noisy depth maps. This leads to only one data term describing a local preference for occupied space $\rho_s := \rho_s^1 = \rho_s^2$. The local preference for free space is implicitly encoded as penalty for occupied space, therefore $\rho_s^0 = 0$.

In the following we consider a particular voxel s and its associated data cost ρ_s. The projective depth of s with respect to one of the input depth maps is denoted as z_s and its observed depth in the depth map is denoted as $\hat{z}_s$. We then define the data term as

$$\rho_s = \begin{cases} \beta & \text{if } z_s < \hat{z}_s, \hat{z}_s - z_s < \delta \\ -\beta & \text{if } z_s > \hat{z}_s, z_s - \hat{z}_s < \delta \\ 0 & \text{otherwise .} \end{cases} \quad (3)$$

This is based on the assumption that in front of an observed depth $\hat{z}$ a voxel s should be assigned free space and in a neighborhood behind occupied space. The weights ρ along a viewing ray are depicted in Fig. 2. Note that we are minimizing our energy and therefore a negative ρ_s means preference for occupied space. This specific choice for ρ_s corresponds to an exponentially distributed noise assumption for depth inliers (see also [9, 19]). Also note that we have only specified our data term for a single observed depth $\hat{z}_s$. It remains to be mentioned that the final data term is formed by summing up all the weights from all the input depth maps.

2.3. Novel Shape Prior Formulation

Unlike recent works on shape priors that fit a parametric shape model to given data [3, 5] we are taking a somewhat different approach. The main drawback in the aforementioned works is that using a parametric shape model brings the additional difficulty that details not represented by the prior need to be added in an additional refinement step. In this section we will detail our shape prior formulation.

For many real world objects, the distributions of normals in a set of given examples is locally restricted. For example, the outline of a bottle is always close to vertical. We exploit this observation to define our shape priors in terms of surface normal distributions. The main idea is to use a

spatially varying anisotropic penalization of the surface.

This can be included in the convex volumetric multi-label segmentation formulation by utilizing smoothness priors $\phi_s^{ij}(\cdot)$ that penalize the surface according to the expected surface orientation at each location s in our volume Ω. For the remainder of the paper, we assume that the domain Ω is aligned with the object's bounding box. At this point, we would like to mention that the smoothness priors $\phi_s^{ij}(\cdot)$ need to be defined for all the possible transitions between the three labels (free space, object and ground). However, the shape prior is only applied to transitions involving the *object* label, meaning the transitions *object* $\leftrightarrow$ *free space* and *object* $\leftrightarrow$ *ground*. In the remainder of this section we therefore omit the label indices and implicitly refer to the relevant transitions.

Any positively 1-homogeneous convex function can be used as a smoothness prior ϕ_s. Choosing functions fulfilling these properties can be difficult, therefore we do not define them directly but use the primal-dual formulation (c.f. [7])

$$\phi_s(x) = \max_{p \in \mathcal{W}_s} (p^T x). \tag{4}$$

$\mathcal{W}_s$ denotes the so-called Wulff shape and can be any closed, bounded and convex set which contains the origin. Using this formulation, we transformed the problem of specifying a suitable function to specifying a convex shape which guarantees the definition of a valid smoothness prior.

This idea has been used before, but only a restricted set of possible shapes was taken into account [23, 9]. Namely, one that prefers a certain direction and one that prefers all the directions that are orthogonal to a given direction. This may be sufficient in these works, but in our case, in order to capture the distribution of normals present on the object's surface faithfully, we need to allow for more general smoothness priors. Therefore, we propose to use a more general parameterization of Wulff shapes.

We start with the following fact: any convex shape can be described as an intersection of a set of half spaces. We choose a discrete set of half spaces $\mathcal{H}_s$ and call the induced shape $\mathcal{W}_{\mathcal{H}_s}$ the discrete Wulff shape. The half spaces $h_s \in \mathcal{H}_s$ are defined by a normal direction $n \in \mathcal{S} \subset \mathbb{S}^2$ and distances to the origin d_s^n. $\mathbb{S}^2$ denotes the three dimensional unit sphere and $\mathcal{S}$ a discrete subset thereof. The distances d_s^n are indexed by the normal direction n. This is not a restriction as only one half space per direction can be active (having a joint boundary with $\mathcal{W}_s$) in the Wulff shape defined by the intersection of all $h_s \in \mathcal{H}_s$. The set of normal directions $\mathcal{S}$ is chosen to be spatially homogeneous and therefore not indexed by the position s. By using a sufficiently fine grained discretization for the normal directions in $\mathcal{S}$, any valid Wulff shape can be represented with high accuracy. Fig. 3 shows visualizations of some examples of possible shapes.

In our experiments, we use a geodesic sphere, which is generated as a iterative subdivision of an icosahedron, to define the set of normal directions $\mathcal{S}$. The number of directions used is $|\mathcal{S}| = 162$. The actual distances d_s^n used for the individual directions n are determined by training data. This step is detailed in the Sec. 2.4.



Figure 3. Discrete Wulff shapes, from left to right: isotropic, preferring one main direction, more complex shape that penalizes a segment of normals much more strongly than the rest.

For the transitions between ground and free space, which are not covered by the shape prior we use a prior preferring the ground to be horizontal as done in [9].

For objects that only touch the ground at specific places *e.g.* the four wheels of a car, the position of the support points are known from the training data. This allows us to utilize the shape prior only in a neighborhood around the area where the interaction is expected to happen. Outside that region we use a strong isotropic penalization. In Sec. 4 it is shown that this enables the optimization to properly segment cars from the ground.

2.4. Training of the Discrete Wulff Shapes

The aim of training the Wulff shapes is to determine the parameters of the shape prior based on training data. Our training data are mesh models that we downloaded from the Internet. Often realistic models also contain the geometry in the interior of the object. In our case we are only interested in the outline shape of the models, more specifically only in the normals of the outline shapes at all positions $s \in \Omega$. To separate the outline shape from the rest we first render depth maps from the given mesh and run volumetric depth map fusion [19] using the rendered depth maps in order to finally extract the outline shape's normals.

The parameters that need to be determined during training are the parameters of the discrete Wulff shapes, namely all the distances d_s^n for all the normals $n \in \mathcal{S}$ at all the positions $s \in \Omega$. Naturally, the terms ϕ_s^{ij} in the energy function in Eq. 1 correspond to negative log-probabilities. The standard approach for the training would be to find a maximum likelihood estimation of the distances d_s^n given a set of training normals. As we need to estimate a set of parameters at each position s a gradient descent based maximum likelihood estimation would be too slow.

To make the problem tractable we discretize the space of directions $x_s^{ij} - x_s^{ji}$ to the ones that are aligned with the discrete set of normals $\mathcal{S}$. Therefore we have for each unit length normal direction $n \in \mathcal{S}$ at position s a direct mapping between the probability $P_s(n)$ and the corresponding d_s^n by

$$P_s(n) = e^{-\phi_s^{ij}(n)} = e^{-\max_{p \in \mathcal{W}_{\mathcal{H}_s}} (p^T n)} = e^{-d_s^n} \tag{5}$$

and it follows that

$$d_s^n = -\log\left(P_s(n)\right). \tag{6}$$

To get an estimate for $P_s(n)$ we generate histograms over the input normal directions weighted by their respective surface area in the training data. The bin centers are set to be $n \in \mathcal{S}$. Now the empirical probabilities can be directly extracted from the histograms and define the d_s^n. For robustness against non observed directions in the training data we impose a maximum on the values d_s^n. At this point we need to make a few remarks.

- In Eq. 5 the arguments to $\phi_s^{ij}(\cdot)$ are unit length normal vectors. However, the arguments $y_s^{ij} := x_s^{ij} - x_s^{ji}$ are in general non-normalized gradient directions from $[-1,1]^3$. From 1-homogeneity it follows that $\phi_s^{ij}(y_s^{ij}) = \|y_s^{ij}\|_2 \phi_s^{ij}(y_s^{ij}/\|y_s^{ij}\|_2)$. And $\|y_s^{ij}\|_2$ corresponds to the area of the surface element in the underlying finite-difference discretization (c.f. [9]). Therefore the surfaces are naturally penalized by their surface area.

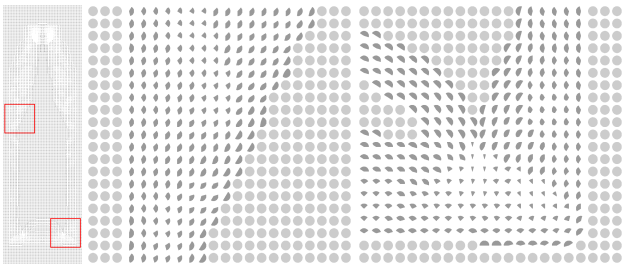
- It can happen that an actual half space h_s is not active (does not share a boundary with $\mathcal{W}_{\mathcal{H}_s}$) in the trained discrete Wulff shape. This means that the corresponding direction n will be penalized less then the training data suggests. This is not an issue as it does not happen for the most probable directions. This behavior can actually be beneficial if a single empty histogram bin is surrounded by much more probable ones.

- In the training process we restricted the directions to the normals $n \in S$. We can still allow for arbitrary directions during the optimization as we generate a valid Wulff shape which assigns a cost to all directions $x_s^{ij} - x_s^{ji}$.

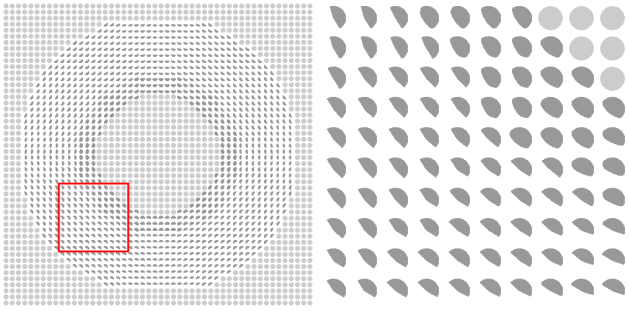
Overall, this way of training the parameters leads to an estimate that penalizes the most probable directions $n \in S$ with the corresponding empirical probability and smoothly blends for directions in between. The training is very efficient as it only involves the computation of a normal histogram while getting an accurate penalization for the most likely directions. In Fig. 4 the shape prior trained on a set of bottles is visualized.

3. Optimization

For the optimization we use the preconditioned first order primal-dual algorithm presented in [14]. In order to apply the algorithm we need to rewrite the optimization problem in its primal-dual saddle point form. This is done by inserting the primal-dual formulation of the smoothness prior from Eq. 4 into Eq. 1 and introducing Lagrange multipliers for the constraints (Eq. 2). The optimization problem can then be stated as



(a) Full slice on the left, closeups in the middle (left side of the bottle neck) and on the right (bottom corner of the bottle).



(b) Full slice on the left, closeup of the lower left part on the right.

Figure 4. Slices through the bottle shape prior in vertical direction (a) and horizontal direction (b). Isotropic Wluff shapes for positions s where there was no training data available are shown in light gray. Trained discrete Wulff shapes are shown in dark gray.

$$\begin{aligned} E_{\mathrm{pd}}(x,p,\lambda,\gamma,\mu) = &\sum_s \left(\sum_i \rho_s^i x_s^i + \mu_s \left(\sum_i x_s^i - 1 \right) \right) \\ &+ \sum_{s,i,k} (\lambda_s^i)_k \left(\sum_j (x_s^{ij})_k - x_s^i \right) + \sum_{s,i,k} (\gamma_s^i)_k \left(\sum_j (x_{s-e_k}^{ji})_k - x_s^i \right) \\ &+ \sum_s \sum_{i,j:i<j} ((p_s^{ij})^T (x_s^{ij} - x_s^{ji})) \end{aligned} \tag{7}$$

subject to $p_s^{ij} \in \mathcal{W}_s^{ij}$, $\mu_s \geq 0$, $x_s^{ij} \geq 0, x_s^i \geq 0$. The objective is minimized with respect to x and maximized with respect to p, λ, γ and μ. The Wulff shapes $\mathcal{W}_s^{ij}$ define the smoothness prior and describe depending on the labels i and j; either our shape prior, or an isotropic or horizontal direction preferring prior.

The updates of the primal-dual algorithm [14] are gradient steps followed by projections to the feasible area. The main difficulty is the projection to the discrete Wulff shapes $\mathcal{W}_{\mathcal{H}_s}$ which define our shape prior. We detail this step in the following section.

3.1. Projection onto Discrete Wulff Shapes

At each iteration of the algorithm the dual variables p_s need to be projected back to the feasible area. In case of the discrete Wulff shapes $\mathcal{W}_{\mathcal{H}_s}$ this has to be done whenever p_s lies outside one or multiple of the half spaces $h_s \in \mathcal{H}_s$. In the following we denote such half spaces h_s as violating

half spaces. In our procedure we exploit that in practice only a few half spaces per Wulff shape are violated in each iteration of the optimization.

As a preprocessing step that only has to be done once for each shape prior we compute the intersection of all the half spaces $h_s \in \mathcal{H}_s$ and store for each half space, in case it is forming a facet of the intersection, the neighboring half spaces along the boundary of the facet. Computing the intersection of the half spaces can be done by utilizing point plane duality (c.f. [6]). For each half space we take the plane defining its boundary, $\pi = [n \quad -d]$ and map it to a dual point $\bar{\pi} = (1/d)n$. A subsequent computation of the convex hull [1] of the dual points reveals the intersection of the planes. The convex shape can then be extracted by transforming back to the primal representation. Note that this way of computing the intersection of half spaces is only valid because the origin is always contained within the shape.

For the projection

$$p'_s = \arg \min_{p'_s \in \mathcal{W}_{\mathcal{H}_s}} \|p_s - p'_s\|_2 \tag{8}$$

of a dual point p_s to the Wulff shape $\mathcal{W}_{\mathcal{H}_s}$ we use the following observation.

If a point p_s lies outside $\mathcal{W}_{\mathcal{H}_s}$, its projection p'_s lies on the boundary of one of the violating half spaces. This can be seen by case distinction. The projection either lies on a facet, edge or vertex of $\mathcal{W}_{\mathcal{H}_s}$, and for all three cases the area where the point p_s needs to be located to project to p'_s is at least outside of one of the half spaces incident to p'_s.

In our projection we first find the set of violating half spaces. Due to the above observation the projection can then be reduced to an exhaustive check of all the violated half spaces (remember in practice only very few half spaces are violated). At this point we would like to make a few remarks:

- Finding the set of all violating half spaces can be expensive. By storing the closest distance $d_s^{\min} := \min_{n \in \mathcal{S}} d_s^n$ for each $\mathcal{W}_{\mathcal{H}_s}$ we can first check if p_s lies withing the $d_s^{\min}$-ball and only do the expensive check if it does not. This considerably speeds up the optimization.
- At many positions s in the volume there is no training data available because the surface never passes through s. In this case an isotropic prior with a high smoothness cost is utilized. To reduce the memory usage we do not store a discrete Wulff shape but a perfect isotropic prior with the corresponding weight.

4. Experiments

Before presenting the results we obtained on challenging real-world datasets we show a synthetic experiment that shows how our shape prior handles shape variations. Often shape variations within a certain class are still constrained to a set of normals which can be captured by our shape prior. For example looking at a door with respect to the frame, or the door of a car with respect to the car, the different opening angles induce big shape variations that cannot be captured by simply capturing the statistics of the occupancy state of a voxel. To demonstrate that our prior is able to capture such variations we trained a shape prior based on synthetic data.

We use a box which is centered at the origin and rotate it around the z-axis. For the training data we use 32 equally spaced angles. Our shape prior can now reconstruct this box with an fully random rotation angle around the z-axis. This is demonstrated by generating a box using a random rotation angle as test data. We rendered eight depth maps where we only keep 4% of the depth values and add Gaussian distributed noise to the remaining depth values.

For all our experiments we use standard volumetric fusion [19] as a baseline method. The data costs for the result with and without shape prior are identical up to a scaling factor which is necessary to account for the different regularization.

The results of the experiments are shown in Fig. 5. Not using a shape prior we cannot reconstruct the box but using our shape prior the box is nicely reconstructed for an arbitrary rotation angle.

In our experiments using real-world data we computed a shape prior for two classes of objects, bottles and cars. Both of the classes share the difficulty that objects are highly reflective and contain transparent elements such as the windows of cars. We trained our priors on mesh models downloaded from the Internet; for the bottle prior the training set size was 48, and for the car prior we used 32 instances.

To underline that our shape prior can be used on arbitrary shapes within the trained object class, we took several real world datasets of bottles and cars of varying shapes. For cars we used real cars as well as toy cars. As a first step the camera poses are computed using the publicly available structure from motion software [21]. We then use our plane sweep stereo matching implementation to compute a depth map for each of the images as a reference view. The matching is done on 5 images at once. Zero mean normalized cross correlation (ZNCC) matching scores are computed between each of the images and the reference view and occlusions are handled using only the best $K = 3$ matching scores.

In order to be able to apply the shape prior, the domain Ω needs to be aligned with the bounding box of the object. This can be done based on image based object detectors *e.g.* [8] and optimization over all the input images [2, 10, 13]. As our work focuses on the actual shape prior formulation we omit this step and annotate the bounding box manually.

In Fig. 6 we present the results for the bottle datasets. The same shape prior was used for all the results. Due to the specular, reflective and translucent areas, an isotopic prior, as used in the baseline, is not faithfully modeling the shapes of the bottles. Our shape prior is general enough to recover

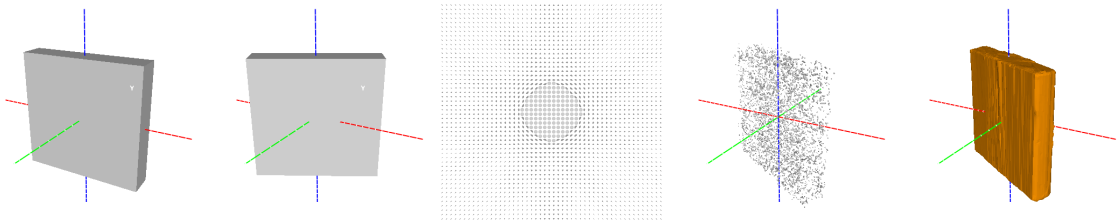


Figure 5. Synthetic experiment. First two images, two out of 32 training shapes. Third image, center part of a horizontal cut through the shape prior. Last two images the reconstruction without and with shape prior, respectively. In the case of not using a shape prior stronger regularization removes the reconstruction entirely.



Figure 6. Reconstructions of the bottle datasets. From left to right: cropped input image, cropped depth map, baseline result [19] and our result using the shape prior with the object and ground segmented. Number of images used, from left to right and top to bottom: 28, 21, 27 and 26.

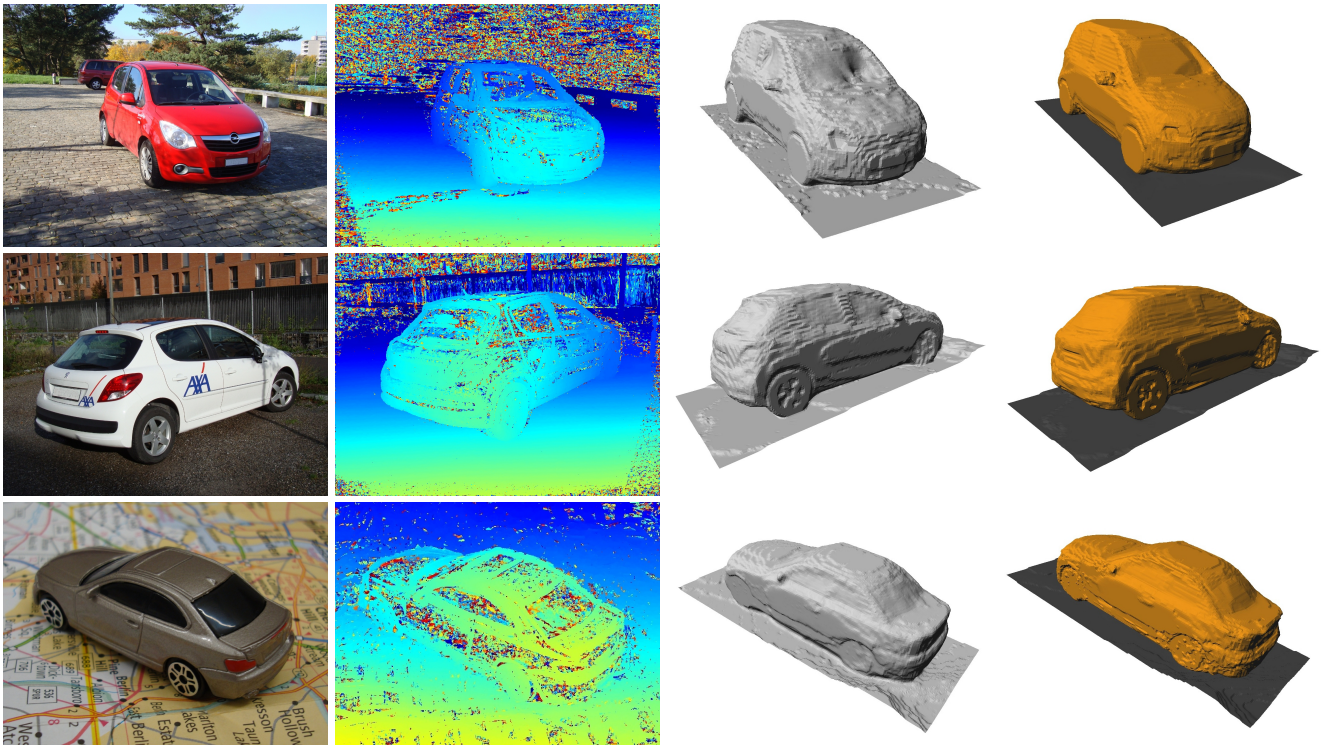


Figure 7. Reconstructions of the car datasets. On the first and second row real cars reconstructed using 80 and 62 images, respectively and on the third row a toy car reconstructed using 27 images. From left to right, cropped input image, cropped depth map, baseline result [19] and our result using the shape prior with the object and ground segmented.

the different shapes present in the input data. Also note that the actual bottle and the ground are segmented through our three label formulation.

Our results on the the car datasets are depicted in Figs. 1 and 7. While a toy car already gives a faithful result in the baseline, applying our shape prior does not change much on the actual shape of the car. Note that details such as the side mirrors and the door handles are present even when utilizing the shape prior. This illustrates that our shape prior is able to allow for a higher variation of normals where they are expected. Additionally, applying the shape prior allows us to segment the ground from the car. Real cars are more challenging then toy cars. Through the transparent windows details in the interior are present in the depth maps, leading to defects on the front wind-shield and the side windows. These issues are fixed by applying our shape prior.

5. Conclusion

We present an approach to include shape priors in convex multi-label segmentation for volumetric 3D reconstruction. It is driven by using spatially varying anisotropic smoothness priors to guide the optimization to the right shape. In order to have a general enough parameterization for anisotropic smoothness we define a discrete Wulff shape, which is formed by the intersection of a discrete number of half spaces. An efficient approach to train the parameters of the discrete Wulff shapes is used to derive the prior from given training data.

On synthetic data we show that our shape prior is capable of capturing big shape variations. Furthermore, we demonstrate the applicability of our formulation on two challenging real world object classes. The prior is able to generalize to different shapes present within a class of objects. We underline this, by showing reconstructions for several examples of each category. Compared to a state-of-the-art baseline not using a shape prior we are able to improve the reconstructions considerably.

Future work needs to examine the possibility of applying such shape priors within reconstructions of larger scenes also containing the surrounding of the object, or scenes with multiple objects of possibly different classes. One step in this direction has been made already in this work by segmenting the ground apart. Also the semantic properties could be investigated further, *e.g*. in the case of a reconstructed car the assigned semantic class to the ground below it is expected to be road.

Acknowledgements: We acknowledge the support of the 4DVideo ERC starting grant #210806 and the V-Charge grant #269916 both under the EC's FP7/2007-2013.

References

[1] CGAL, Computational Geometry Algorithms Library. http://www.cgal.org.

[2] S. Y. Bao and S. Savarese. Semantic structure from motion. In *Proc. CVPR*, 2011.

[3] Y. Bao, M. Chandraker, Y. Lin, and S. Savarese. Dense object reconstruction with semantic priors. In *Proc. CVPR*, 2013.

[4] A. Chambolle, D. Cremers, and T. Pock. A convex approach to minimal partitions. *SIIMS*, 5(4), 2012.

[5] A. Dame, V. A. Prisacariu, C. Y. Ren, and I. Reid. Dense reconstruction using 3d object shape priors. In *Proc. CVPR*, 2013.

[6] M. De Berg, M. Van Kreveld, M. Overmars, and O. C. Schwarzkopf. *Computational geometry*. 2000.

[7] S. Esedoḡlu and S. J. Osher. Decomposition of images by the anisotropic rudin-osher-fatemi model. *Comm. PAM*, 57(12), 2004.

[8] P. F. Felzenszwalb, R. B. Girshick, D. McAllester, and D. Ramanan. Object detection with discriminatively trained part-based models. *Trans. PAMI*, 32(9), 2010.

[9] C. Häne, C. Zach, A. Cohen, R. Angst, and M. Pollefeys. Joint 3d scene reconstruction and class segmentation. In *Proc. CVPR*, 2013.

[10] M. Hödlmoser, B. Micusik, M.-Y. Liu, M. Pollefeys, and M. Kampel. Classification and pose estimation of vehicles in videos by 3d modeling within discrete-continuous optimization. In *Proc. 3DIMPVT*, 2012.

[11] K. Kolev, T. Pock, and D. Cremers. Anisotropic minimal surfaces integrating photoconsistency and normal information for multiview stereo. In *Proc. ECCV*, 2010.

[12] V. Lempitsky and Y. Boykov. Global optimization for shape fitting. In *Proc. CVPR*, 2007.

[13] J. Liebelt and C. Schmid. Multi-view object class detection with a 3d geometric model. In *Proc. CVPR*, 2010.

[14] T. Pock and A. Chambolle. Diagonal preconditioning for first order primal-dual algorithms in convex optimization. In *Proc. ICCV*, 2011.

[15] S. M. Seitz, B. Curless, J. Diebel, D. Scharstein, and R. Szeliski. A comparison and evaluation of multi-view stereo reconstruction algorithms. In *Proc. CVPR*, 2006.

[16] E. Strekalovskiy and D. Cremers. Generalized ordering constraints for multilabel optimization. In *Proc. ICCV*, 2011.

[17] J. Stühmer, P. Schröder, and D. Cremers. Tree shape priors with connectivity constraints using convex relaxation on general graphs. In *Proc. ICCV*, 2013.

[18] S. Vicente, V. Kolmogorov, and C. Rother. Graph cut based image segmentation with connectivity priors. In *Proc. CVPR*, 2008.

[19] C. Zach. Fast and high quality fusion of depth maps. In *Proc. 3DPVT*, 2008.

[20] C. Zach, C. Häne, and M. Pollefeys. What is optimized in convex relaxations for multi-label problems: Connecting discrete and continuously-inspired map inference. *Trans. PAMI*, 2014.

[21] C. Zach, M. Klopschitz, and M. Pollefeys. Disambiguating visual relations using loop constraints. In *Proc. CVPR*, 2010.

[22] C. Zach, T. Pock, and H. Bischof. A globally optimal algorithm for robust tv-l^1 range image integration. In *Proc. ICCV*, 2007.

[23] C. Zach, L. Shan, and M. Niethammer. Globally optimal finsler active contours. In *Proc. DAGM*, 2009.

D Driven Learning for Driving: How Introspection Improves Semantic Mapping

Authors: Hugo Grimmett, Rudolph Triebel, Rohan Paul and Ingmar Posner

Link to Table of Contents.

Driven Learning for Driving: How Introspection Improves Semantic Mapping

Rudolph Triebel, Hugo Grimmett, Rohan Paul, and Ingmar Posner

Abstract This paper explores the suitability of commonly employed classification methods to action selection tasks in robotics and argues that a classifier's *introspective* capacity is a vital but as yet largely under-appreciated attribute. As illustration we propose an active learning framework for semantic mapping in mobile robotics and demonstrate it in the context of autonomous driving. In this framework data are selected for label disambiguation by a human supervisor using uncertainty sampling. Intuitively, an introspective classification framework – i.e. one which moderates its predictions by an estimate of how well it is placed to make a call in a particular situation – is particularly well suited to this task. To achieve an efficient implementation we extend the notion of introspection to a particular sparse Gaussian Process Classifier, the Informative Vector Machine (IVM). Furthermore, we leverage the information-theoretic nature of the IVM to formulate a principled mechanism for forgetting stale data, thereby bounding memory use and resulting in a truly life-long learning system. Our evaluation on a publicly available dataset shows that an introspective active learner asks more informative questions compared to a more traditional non-introspective approach like a Support Vector Machine (SVM) and in so doing, outperforms the SVM in terms of learning rate while retaining efficiency for practical use.

1 Introduction

In answering the question 'where am I?' roboticists have gone to great lengths to model, manage, cope with and, indeed, exploit uncertainty. This, however, is not

Rudolph Triebel
Mobile Robotics Group, Oxford University, UK, e-mail: rudolph.triebel@in.tum.de,
who has since moved to Computer Vision Group, Technical University of Munich, Germany

Hugo Grimmett, Rohan Paul, and Ingmar Posner
Mobile Robotics Group, Oxford University, UK, e-mail: {hugo, rohanp, hip}@robots.ox.ac.uk

as yet the case when it comes to asking 'what is this?'. As we aspire to robust, long-term autonomous operation our systems have to contend with vast amounts of continually evolving, non-i.i.d. data from which information needs to be assimilated. This presents a challenge and an opportunity particularly to the robotics community as here the real cost of failure can be significant. We believe that *realistic* estimates of uncertainty are pivotal to achieving robust and efficient decision making in robotics. However, in particular in the context of classification as a precursor to action selection this aspect seems to be largely disregarded by the community.

We frame our argument in the context of offline semantic mapping. Significant progress in autonomous driving in recent years has inspired a view that successful autonomous operation in complex, dynamic environments critically depends on *a-priori* available semantic maps representing ostensibly permanent aspects of the environment such as lane markings, traffic light positions and road sign information (see, for example, [2, 21]). Owing to their safety-critical nature, these maps are typically created manually for particular routes [4]. This is, of course, an expensive process which scales badly with the number of routes for which autonomous operation is to be provided. Much, therefore, can be gained by reducing human involvement in this process and thus providing a robust and scalable solution.

A prominent approach to tackling such a challenge is that of active learning, where classification results are iteratively refined by asking a human supervisor for ground-truth labels in ambiguous cases and incorporating the added information into classifier training. To the best of our knowledge this paper is the first in robotics to present an efficient and scalable active learning framework for the task of offline semantic mapping. Crucially, however, our work is also set apart from the vast majority of the related works in active learning by the unusual stance we take with regards to uncertainty estimates in the system. Commonly, active learning relies on selecting data for human labelling using a variant of *uncertainty sampling*, by which data are selected according to how confident a classifier is in individual predictions (see, for example, [16]).

However, Grimmett *et al.* [6] show that several of the classification frameworks commonly used in robotics are unrealistically overconfident in their assessment of class membership. To characterise this attribute, the authors introduce the notion of an *introspective capacity* of a classification framework: the ability to estimate a classification confidence which realistically reflects how qualified the classifier is to make a particular class decision in each individual test instance. In this paper we show that *introspective classification* harbours significant benefits for active learning as compared to more traditional, non-introspective approaches. In particular, our contributions are

- the application of an active learning framework to semantic mapping in robotics,
- the application of the notion of introspection to the Informative Vector Machine (IVM) [9] as an efficient extension to [6],
- the application of the IVM specifically to achieve *introspective* active learning, which is demonstrated to lead to more effective information extraction over more traditional approaches, and

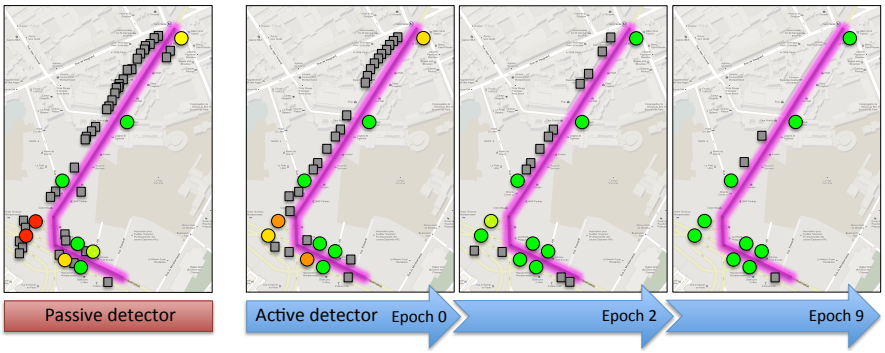


Fig. 1: Active learning in a semantic mapping context. This figure shows semantic maps indicating the positions of traffic lights along a street in Paris. Circles denote the locations of ground-truth traffic lights. The shading encodes the correctness of the classification output as provided by a probabilistic classifier: red denotes no detections, and green denotes all possible traffic light detections being found. False positives are shown as grey squares. From left to right, we first see a typical passive detector, followed by our active-learning framework at epochs 0, 2, and 9 respectively. Note that in the active learning setting the shading of the circles progresses from red to green as a greater proportion traffic lights are correctly detected with increasing confidence. Similarly the number of false positives reduces dramatically. By epoch 2 the active learning framework already outperforms the passive detector. In this paper we show that our formulation of an *introspective* active learning approach provides for more efficient information extraction – and thus a higher learning rate – over conventional active learning approaches. (This figure is best viewed in colour.)

- the introduction of a principled mechanism for the IVM to *forget* less important data to provide for scalable, life-long active learning on a mobile robot.

The work presented here first appeared as a workshop paper by the same authors [20]. However, here we offer a more detailed treatment of that work as well as the following significant extensions:

- the introspective capacity of the IVM is established, including the effects of varying the sparsity factor,
- qualitative results are included of when the IVM is confident (correctly and incorrectly) in its classifications, and
- timing information is provided regarding the training of an IVM.

We apply our framework to the detection of traffic lights in a real, third-party vision dataset and demonstrate iteratively improved semantic mapping, which makes efficient use of available label information. A typical qualitative example of our system output is shown in Fig. 1.

2 Related Works

Active learning is an established and vibrant field of research spanning a significant number of application domains. Consequently, a variety of methods have been proposed for selecting informative measurements for labelling and/or for incrementally training a learning algorithm. For example, Freund *et al.* [5] propose disagreement among a committee of classifiers as a criterion for active data selection. McCullum and Nigam [11] apply this to text classification using high label inconsistency as a query criterion coupled with expectation maximisation (EM) for online learning. More recently, Joshi *et al.* [7] address multi-class image classification using SVMs and propose criteria based on entropy and best-versus-second-best (BvSB) measures based on the hyperplane-margin for determining uncertain points. Tong and Koller [18] pick unlabelled data for query based on minimising the version space within a margin-based SVM formulation. Kapoor *et al.* [8] propose an active learning system for object categorization using a GP classifier where data points possessing large uncertainty (using posterior mean and variance) are queried for labels and used to improve classification.

Within the robotics community, active learning and directed information acquisition has received attention in recognition, planning and mapping tasks. For example, Dima *et al.* [3] present unlabelled data filtering for outdoor terrain classification tasks with the aim of reducing the amount of training data to be human-labelled. The approach relies on kernel density estimation over unlabelled data and estimating a "surprise" score for image patches, hence only querying the least likely samples given the density estimate for human labelling. In [13] the authors present a learning approach for continually improving place recognition perfomance by actively learning an appearance model of a robot's operating environment. The method uses probabilistic topic models and a measure of perplexity to identify least explained images which further drives retrieval of thematically linked samples leading to an improved workspace representation. Recent work by Tellex *et al.* [17] explores active information gathering for human-robot dialog. The authors formulate an information-theoretic strategy for asking clarifying questions to disambiguate the robot's belief over the mapping between phrases and aspects of the workspace.

While, to the best of our knowledge, this is the first work in robotics applying active learning to a semantic mapping task, our work is also set apart significantly from prior art in active learning in that we introduce and demonstrate the benefits of efficient *introspective* active learning. In this respect, the work most closely related to ours is that of [8] above, in which an inherently introspective classifier is used but its use is not motivated by its introspective qualities.

3 Introspective Classification

The introspective capacity of a classifier characterises its ability to *realistically* estimate the uncertainty in its predictions. Grimmett *et al.* [6] define the introspective capacity as a classifier's ability to moderate its output by an appropriate measure as to how 'qualified' it is to make a call given its own prior experience, usually in the form of training data. The intuition is that test data, which are in some form 'similar' to that seen in training, are classified with higher certainty than data which are more dissimilar. This points towards non-parametric approaches potentially being more introspective than parametric ones, as all the training data are available for inference in the former, whereas inference in the latter is based on parametric models learned from the data. Grimmett *et al.* [6] investigated several commonly used classification frameworks providing probabilistic output and found that a Gaussian Process classifier (GPC) [15] indeed is significantly more introspective than, for example, the more commonly used Support Vector Machine (see, for example, [1]) with a probabilistic calibration (such as, for example, provided by Platt *et al.* [14]).

In [6], this quality is attributed to a GPC's Bayesian treatment of predictive variance. Consider a set of training data $\{X, \mathbf{y}\}$, where $X = \{\mathbf{x}_1, \ldots, \mathbf{x}_{|X|}\}$ denotes the set of feature vectors and $\mathbf{y}$ denotes the set of corresponding class labels. Probabilistic predictions for a test point, $\mathbf{x}_*$, are obtained in two steps. First, the distribution over the latent variable corresponding to the test input is obtained by

$$p(f_* \mid X, \mathbf{y}, \mathbf{x}_*) = \int p(f_* \mid X, \mathbf{x}_*, f) p(f \mid X, \mathbf{y}) df, \tag{1}$$

where $p(f \mid X, \mathbf{y})$ is the posterior distribution over latent variables. This is followed by applying a sigmoid function $\sigma(\cdot)$, which in our implementation is the cumulative Gaussian, and *marginalising* over the latent f_* to yield the class likelihood $p(y_* \mid X, \mathbf{y}, \mathbf{x}_*)$ as

$$p(y_* \mid X, \mathbf{y}, \mathbf{x}_*) = \int \sigma(f_*) p(f_* \mid X, \mathbf{y}, \mathbf{x}_*) df_*. \tag{2}$$

It is this *marginalisation* over all models induced by the training set, as opposed to relying on a single *minimisation*-based estimate, which accounts for an accurate estimate of the inherent uncertainty in class distribution, and therefore endows GP classification with a high introspective capacity.

3.1 Efficiency by Sparsification

A key drawback of a GPC is its significant computational demand in terms of memory and run time. This is due to the fact that the GP maintains a mean $\boldsymbol{\mu}$, as well as a covariance matrix Σ, which is computed from a kernel function and has size $|\mathbf{y}|^2$. A number of sparsification methods have been proposed in order to mitigate this computational burden. For efficiency, in this work we adopt one such sparsification

method: the Informative Vector Machine (IVM) [9]. The main idea of this algorithm is to only use a subset of the training points denoted the *active set*, $\mathcal{I}$, from which an approximation $q(f \mid X, \mathbf{y}) = \mathcal{N}(f \mid \boldsymbol{\mu}, \Sigma)$ of the posterior distribution $p(f \mid X, \mathbf{y})$ is computed. The IVM algorithm computes $\boldsymbol{\mu}$ and Σ incrementally, and at every iteration j selects the training point $(\mathbf{x}_k, y_k)$ which maximizes the entropy difference ΔH_{jk} between q_{j-1} and q_j for inclusion into the active set. As q is Gaussian, ΔH_{jk} can be computed by

$$\Delta H_{jk} = -\frac{1}{2}\log|\Sigma_{jk}| + \frac{1}{2}\log|\Sigma_{j-1}|. \tag{3}$$

We implement an efficient form of this, the details of which can be found in Lawrence *et al.* [10]. The algorithm stops when the active set has reached a desired size. In our implementation, we choose this size to be a fixed fraction of the training set q.

To find the kernel hyper-parameters $\boldsymbol{\theta}$ of an IVM, two steps are processed in a loop for a given number of times: estimation of $\mathcal{I}$ from $\boldsymbol{\theta}$ and minimising the marginal likelihood $q(\mathbf{y} \mid X)$, thereby keeping $\mathcal{I}$ fixed. Although there are no convergence guarantees, in practice already a small number of iterations are sufficient to find good kernel hyper-parameters.

Importantly for our work, since inference with the IVM is similar to that with a GPC, the IVM retains the model averaging described in Eq. (2). We argue, therefore, that the IVM provides a significant and well-established improvement in processing speed over a GPC while maintaining its introspective properties (see Sec. 5 and 5.4 for details).

4 Scalable Active Learning: Drive, Ask, Improve

The power of an active learning framework lies in its ability to select a suitable training set in an application-oriented way. It thus inherently allows the system to naturally adapt to the non-stationarity of the data often encountered in long-term robotics applications. The active learning framework considered here provides for supervised learning where a human operator furnishes class labels for selected test data that are then fed back into classifier training to improve the classification result of the next round. We therefore consider progress in terms of consecutive *epochs*, which each consist of (re-)training, classification, and user-feedback. The implementation of a scalable active learning framework requires two problems to be addressed: firstly, a subset of test data has to be selected for re-training such that classification performance increases in the next epoch. Secondly, measures have to be taken that guarantee that the training set is bounded in size, since otherwise the algorithm will sooner or later exhaust the resources of a finite-memory, real robotic system. We compare this active learning approach with what we call a "passive" alternative, that is, training a classifier once without any subsequent human-feedback improvement.

We now outline the specific active learning algorithm employed in this work, before providing details of both our data selection strategy and our approach to forgetting (bounding the training set size).

4.1 The Active Learning Algorithm

Algorithm 1 describes our active learning framework which, for reasons given in Sec. 3, uses an IVM as the underlying classifier. It requires five different input parameters: the initial hyper-parameters $\boldsymbol{\theta}_0$ used for training the IVM, the fraction γ of training points that are used for sparsification, the batch size b, the normalised entropy threshold ϑ that a test point needs to exceed to be considered for retraining, and the maximum number of questions r that the algorithm may ask. The last is intended to minimise nuisance to a human operator due to being asked too many questions. The sub-routines in the algorithm are explained as follows.

Algorithm 1: Active Learning with an IVM

Data: training data $\mathcal{D} = (X, \mathbf{y})$, stream of test data X^*
Input: initial kernel parameters $\boldsymbol{\theta}_0$, batch size b, active set size fraction γ, minimal retraining score ϑ, maximum number of questions r
Output: stream of output labels $\mathbf{y}^*$

$i \leftarrow 0$
while $X^* \neq \emptyset$ **do**
  $(\boldsymbol{\theta}_{i+1}, \mathcal{I}_{i+1}) \leftarrow$ `TrainIVM`$(X, \mathbf{y}, \gamma, \theta_0)$
  move next b test points from X^* into X_i^*
  $\mathcal{P} \leftarrow \emptyset$
  forall the $\mathbf{x}^* \in X_i^*$ **do**
    $z \leftarrow$ `IVMPrediction`$(\mathcal{I}_{i+1}, \boldsymbol{\theta}_{i+1}, \mathbf{x}^*)$
    $s \leftarrow$ `ComputeRetrainingScore`(z)
    if $s > \vartheta$ **then** $\mathcal{P} \leftarrow \mathcal{P} \cup \{(\mathbf{x}^*, s)\}$
  sort $\mathcal{P}$ by decreasing values of s
  $\mathcal{D}^+ \leftarrow \emptyset$
  for $j \leftarrow 1$ **to** `MIN`$(r, |\mathcal{P}|)$ **do**
    $(\mathbf{x}_j^+, s_j) \leftarrow$ element j of $\mathcal{P}$
    $y_j^+ \leftarrow$`AskLabelFromUser` $(\mathbf{x}_j^+)$
    $\mathcal{D}^+ \leftarrow \mathcal{D}^+ \cup (\mathbf{x}_j^+, y_j^+)$
  $\mathcal{D} \leftarrow \mathcal{D} \cup \mathcal{D}^+, \quad i \leftarrow i+1$

`TrainIVM` uses the current training set, the active set fraction γ, and the initial kernel parameters to find optimal kernel parameters $\boldsymbol{\theta}_{i+1}$ and an active set $\mathcal{I}_{i+1}$ as described in Sec. 3.1. Throughout this work we employ a squared exponential kernel with additive white noise, such that

$$k(\mathbf{x}_i, \mathbf{x}_j) = \sigma_f^2 e^{-\frac{(\mathbf{x}_i - \mathbf{x}_j)^2}{2l^2}} + \sigma_n^2 \delta_{ij}, \tag{4}$$

where δ_{ij} is the Kronecker delta, and $\boldsymbol{\theta} = \{\sigma_f^2, l, \sigma_n^2\}$ are the signal variance, the length scale, and the noise variance.

`IVMPrediction` returns an estimate of the probability z that the next test datum $\mathbf{x}^*$ has a particular class label, as given in Eq. (2). Based on this probability, the normalised entropy measure is then computed. The top ranked r test data exceeding the retraining threshold ϑ are labelled by the user and added to the training set for the next epoch.

4.2 Data Selection Strategy: What Questions to Ask?

The key element of an active learning algorithm is the strategy by which a new test point $\mathbf{x}^*$ is considered for re-training. In Algorithm 1, this is done in the subroutine `ComputeRetrainingScore`. An intuitive and well-explored indicator of which data might be suitable for inclusion is the classification *uncertainty* associated with $\mathbf{x}^*$. To characterise the uncertainty of the classification from the given class prediction $z = p(y_* \mid X, \mathbf{y}, \mathbf{x}_*)$, we adopt the measure of *normalised entropy* $H(z)$, such that for the binary case,

$$H(z) = -z \cdot \log_2(z) - (1 - z) \cdot \log_2(1 - z), \tag{5}$$

where $H(z) \in [0, 1]$, with high values representing high uncertainty.

This, indeed is central to our work. While, in principle, any classification framework which provides a distribution over class labels as output can be used in our active learning framework, intuitively we expect those with more realistic estimates of these probabilities to be more effective for active learning. Thus, we expect more introspective classifiers to perform better in the sense that they will ask more informative questions, leading to a higher learning rate. In Sec. 5, we will show that this is indeed the case when comparing the proposed framework based on an IVM with one based on a more commonly used, probabilistically calibrated SVM.

4.3 Forgetting Uninformative Data to Bound Memory Use

The main problem with the active learning framework as we presented it so far is that in theory the training set can grow indefinitely, because there are no guarantees that the algorithm will stop asking new questions. This makes the algorithm less flexible, especially if the input data can not be guaranteed to be within certain locality bounds, for example in a life-long learning application. Therefore, and for run time efficiency, we bound the size of the training set by removing points from it when it exceeds a given target size t. To decide which points to remove, we leverage the

information-theoretic instruments that the IVM already provides. After each training round, we keep the entropy differences given in Eq. (3) for all training points and sort them in increasing order. Those training data which correspond to the first $n_i - t$ values, where n_i is the current training set size, are then removed. Intuitively, this method discards the data that were least informative during the last training round, those which influence the classification performance the least. One caveat with this method is that it assumes independence between the training data, which is not generally given. For example, two data may both have small individual ΔH values, but when removing both of them the entropy could change significantly. In this work we acknowledge but do not explore this phenomenon. Instead, we note that in our experiments we did not observe a deterioration in classification performance when we applied our method for forgetting.

5 Experimental Results

In this section we investigate the performance of our introspective active learning approach in terms of learning rate, data selection strategy, classification performance and tractability. We compare and contrast our approach with one based on the much more commonly used SVM classifier (calibrated to provide probabilistic output). The task we set both learners is to detect traffic lights in a third-party image dataset. Specifically, we use the publicly available Traffic Lights Recognition (TLR) data set [12], which comprises 11,179 colour images taken at 25 Hz from a car driven through central Paris at speeds under 31 mph. It has ground-truth labels for traffic light positions and subtype labels 'green', 'orange', 'red', 'ambiguous' (though here we are only concerned with the detection of traffic lights, irrespective of their state). As recommended by the authors of the dataset, we disregard labels of type 'ambiguous' and exclude sections where the vehicle was stationary for long periods of time. We use data from the first 7,200 frames for training and the remainder for testing. We compute a template-based feature set inspired by Torralba *et al.* [19] which has a successful track record in the detection of traffic lights [6]. From each of the training partition and the test partition we extract 1,000 positive windows and 10,000 negative windows, giving rise to 22,000 feature vectors of dimension 200.

As described in Sec. 4, our active learning algorithm is retrained after having seen a batch of test points, as opposed to running the training algorithm after every new datum encountered. During our experiments, the classifiers each go through 10 epochs. Every epoch consists of a training phase, a classification phase, and a feedback phase. At the very start of epoch 0, the classifiers are trained on a randomly chosen 5% of the training set. During each classification phase, the classifiers are then tested on a batch of 1,000 points randomly drawn from the test set. Then the 10 points with the highest normalised entropy (providing they are over a threshold empirically set to be $\vartheta = 0.97$) are greedily added to the training set, ready for retraining at the start of the next epoch. Note that each classifier (IVM and SVM) makes its own choices regarding which points to add for the next epoch.

For these experiments, we used an active set fraction γ of 0.2 (see Sec. 5.2 for more details), which means that informative points will be added to the active set until its size is 20% of the training set size. Training such a classifier takes approximately 10 minutes (on a single 800MHz core), and this time is almost entirely spent optimising the kernel parameters. It does not impact either the learning rate or the classification results in a significant manner if we only train the kernel parameters during epoch 0 and keep those same values throughout. However, it does make training much faster, in fact it takes less than a second to decide on a new active set, so the total training time stays constant at about 10 minutes.

5.1 Does Introspection Improve Active Learning?

One of the central claims of this paper is that the use of an introspective classifier will lead to more informative questions being asked of the human expert. In order to test this claim both an IVM and an SVM are initially trained on the same data. Then, 1,000 new data are shown to both classifiers for testing. Each chooses 10 data points to add to their own training set for the next round, resulting in *two* new and different training sets: the 'IVM set' and the 'SVM set'. A new IVM and SVM are now trained on *each* of the two new sets and evaluated on a further 1,000 new data points. This process thus gives rise to four classifiers: two IVMs trained on data selected by an IVM and a SVM respectively, and two equivalent SVMs. We compute precision and recall for all four classifiers. The results after 100 repetitions of this experiment are shown in Fig. 2. As expected, both the IVM and the SVM perform better when trained on the dataset chosen by the initial IVM, suggesting that the questions asked by the IVM tend to be significantly more informative. An unpaired t-test shows this result to be significant to a level of over 99.9%. The overall effect of introspection in an active learning setting is therefore an increased learning rate, as shown in Fig. 3, where the IVM learner based on a normalised entropy selection policy outperforms the equivalent SVM based method both in absolute terms per epoch as well as in terms of relative increase (information gained) per epoch.

Fig. 3 further serves to justify empirically our choice of normalised entropy as a valid criterion for data selection by comparing it to randomly selecting new training data. Intuitively, both methods should improve classification by virtue of the fact that they increase the training set size. However, the results indicate that for both the IVM and the SVM, using normalised entropy leads to more rapidly improving classification performance.

5.2 Does Forgetting Affect the Performance?

Our work aims to contribute an introspective active learning algorithm that is efficient in terms of computational effort and scalable with respect to its memory

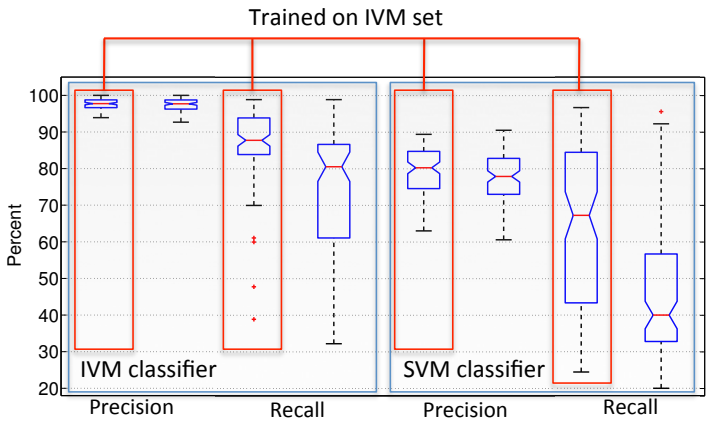


Fig. 2: The benefit of introspection. Data selected by an introspective classifier lead to an improved learning rate in terms of precision and recall for both and IVM and and SVM over that selected by a non-introspective classifier. Results are shown for 100 experimental runs. See text for more details.

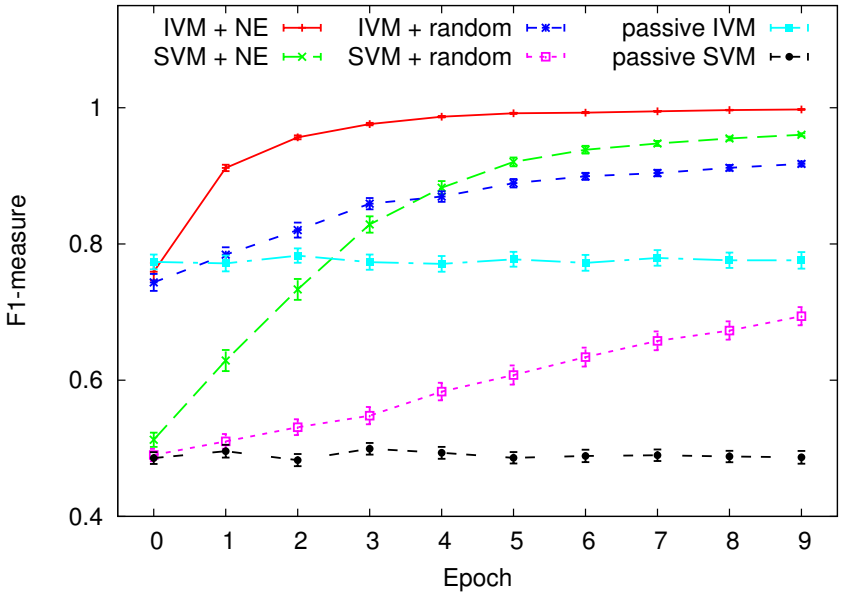


Fig. 3: Classification performance and (learning rates) for both IVM and SVM variants as indicated by the f_1-measure after each epoch. Measurements are averaged over 100 runs. Error bars indicate one standard error of the mean. The IVM using a normalised entropy-based data selection strategy (IVM+NE) consistently outperforms all other active learning variants in terms of overall performance and learning rate.

requirements. In this section we investigate the efficacy of the mechanism we have put in place to provide this tractability: forgetting. In experiments thus far new training data were added in each epoch. The IVM active set size is a fixed proportion of the training set size, $\gamma = 0.2$. This has the benefit of increasing classification performance, but to the detriment of processing time. In the context of a life-long-learner, this is not a scalable solution.

We therefore elect to cap the size of the training set at 550 data, which in turn makes the computational effort constant at every epoch. This *IVM+forgetting* learner can add new data, but only by simultaneously discarding a similar number. Fig. 4 (left) shows the level of sparsity for three different active learners: IVM, SVM, and IVM+forgetting. The SVM sparsity is unbounded, and so grows rapidly with training set size. The IVM active set size grows at 20% of the training set, but the IVM+forgetting is constant past epoch 2. At epoch 2, the specified maximum training set size t has been reached and the IVM+forgetting learner starts to discard uninformative points. Fig. 4 (right) shows the corresponding classification performance as characterised by the f_1 measure. It indicates that, in this case, the IVM+forgetting mechanism performs no worse than the original IVM.

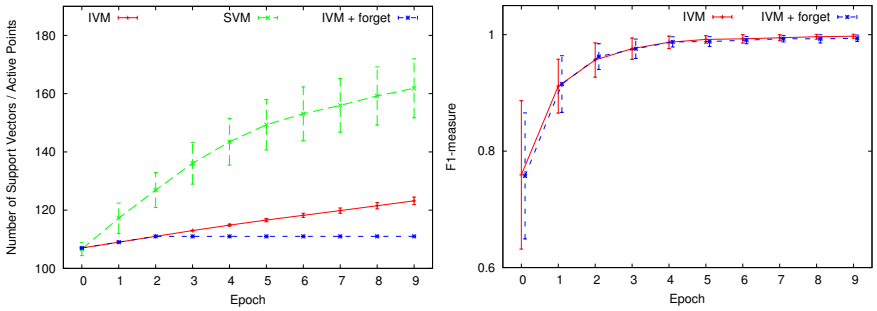


Fig. 4: *Forgetting* results in commensurate classification performance while successfully bounding the active set size of the classifier. Each datum represents the mean (and associated standard deviation) over 100 experimental runs. **Left:** The evolution of sparsity (number of support vectors for the SVM, active set size for the IVM) over several epochs. **Right:** Classification performance with and without *forgetting*. For corresponding SVM results, see Fig. 3.

5.3 What Does the Active Learner Ask?

In Figure 5 we show the 49 most certain and 49 least certain test cases at each epoch, and whether they were correctly classified or not. Firstly, it is reassuring to confirm that the certain classifications are always correct. The uncertain classifications start off being very incorrect, and then the number of false instances decreases suddenly, reaching zero by epoch 10. Secondly, the class fractions are of great interest: at

epoch 1 the uncertain classifications are almost all instances of traffic lights, and then as the learner asks questions to the human supervisor and improves, fewer and fewer traffic lights make up the most uncertain images. In fact, traffic lights even start to enter the most certain images by epoch 10. This shows how well the active learner adapts to differentiating between classes after only a few rounds of retraining.

It is worth noting that many of the images are very uniform in colour, usually a solid shade of dark grey (from the road) or white (from the sky), and that these could be classed as not containing a traffic light immediately using a pre-processing classification step. Although the IVM is fast at the testing phase, this could further speed up the system.

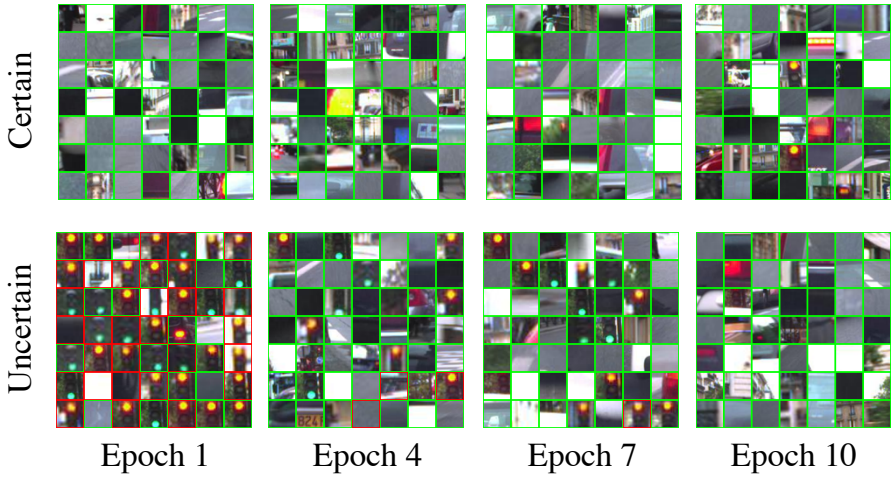


Fig. 5: The 49 most certain and 49 least uncertain test classifications of an IVM at epochs 1, 4, 7, and 10. A green border indicates a correct classification, and red for incorrect.

5.4 The Effects of Sparsity

In [6] we showed that the GPC is more introspective than other more commonly used classification frameworks. In this paper we have argued the necessity of using a sparse formulation for the sake of computational complexity, but first it is necessary to make sure that the IVM is introspective in its own right. The useful characteristic of an introspective classifier is that it tends to be confident when it is making true predictions, and uncertain when it may be making false predictions. In addition, we would like to see whether the introspective quality changes with the active set size; intuitively, a truly introspective classifier will be more confident if it is exposed to more data, and vice versa.

Similarly to in [6] we have plotted the cumulative true and false classifications against uncertainty in Figure 6. In the legend, "IVM-0.4" indicates an IVM with an active set fraction γ of 0.4, such that the active set contains 40% of the training set. There are several things of note here. Firstly, we can see that looking at the curves for the IVM-$\{0.2, 0.4, 0.6, 0.8, 1.0\}$, indeed as we would hope, having a larger active set results in a more confident classifier; however it is interesting to see that there are diminishing returns: very little is gained between an active set fraction of 0.6 and 1.0. Secondly and most importantly, *the IVM is introspective*: if we threshold at, say, NE = 0.5, we get no false classifications at all, and most of the true classifications. Thirdly, even with an active set fraction of 1.0 the IVM is less confident than the GPCs. We propose that this is due to the method by which the IVM optimises the kernel parameters during each round of learning. The full GPC implementation uses expectation propagation (EP), whereas the IVM essentially uses a 1-round equivalent called Assumed Density Filtering (ADF). This ADF round is much faster than EP but does not produce as good estimates for the kernel parameters.

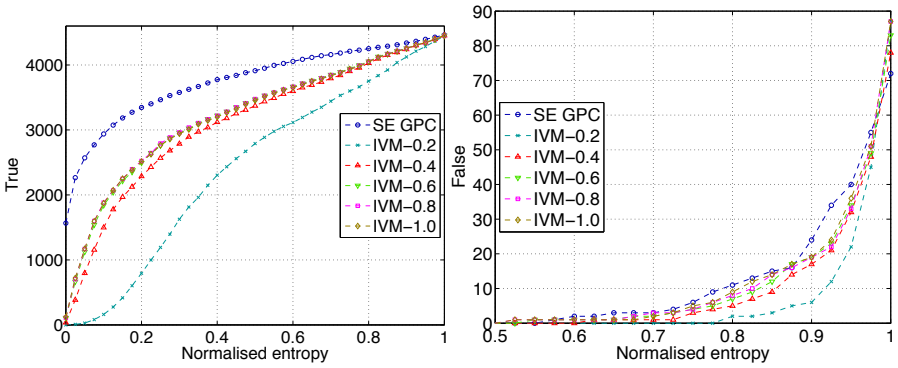


Fig. 6: We show the number of true (**Left**) and false (**Right**) classifications (positive and negative classes together) which are made with a normalised entropy lower than a chosen value. For instance, if we were to threshold at NE = 0.6, we would have 88% of the positives with the SE GPC, 80% with the the IVM-$\{0.4, 0.6, 0.8, 1.0\}$, and 70% with the IVM-0.2. However, we would also have fewer than 5 misclassifications. Here, IVM-0.4 represents am IVM with an active set fraction γ of 0.4, i.e. 40% of the training data in the active set.

6 Conclusion

The contributions of this paper are three-fold: firstly, the notion of introspective classification introduced earlier is shown to be particularly important in the context of active learning, where a reliable estimate of the classification uncertainty is required. We do this by showing an improvement in both classification performance and learning rate over a non-introspective classifier (Sec. 5.1). Secondly, an efficient

version of the Gaussian Process Classifier, namely the Informative Vector Machine is used, which makes the approach particularly useful for robotics applications with large amounts of data. We show visual examples of where it is confused and where it is confident (Sec. 5.3), and use it to create the first offline semantic mapping algorithm via active learning. And finally, we present an elegant, information-theoretic solution to the problem of increasing memory requirements by forgetting the least informative data, which further increases the significance of the approach for mobile robotics applications (Sec. 5.2).

7 Acknowledgements

This work is funded under the European Community's Seventh Framework Programme (FP7/2007-2013) under Grant Agreement Number 269916 (V-CHARGE) and by the UK EPSRC Grant Number EP/J012017/1.

References

[1] C. J. C. Burges. A Tutorial on Support Vector Machines for Pattern Recognition. *Data Mining and Knowledge Discovery*, 2(2):121–167, 1998.

[2] DARPA. Urban Challenge Route Network Definition File (RNDF) and Mission Data File (MDF) Formats, March 2007.

[3] C. Dima, M. Hebert, and A. Stentz. Enabling learning from large datasets: Applying active learning to mobile robotics. In *IEEE Intern. Conf. on Robotics and Automation (ICRA)*, volume 1, pages 108–114. IEEE, 2004.

[4] N. Fairfield and C. Urmson. Traffic light mapping and detection. In *IEEE International Conference on Robotics and Automation (ICRA)*, 2011.

[5] Y. Freund, H.S. Seung, E. Shamir, and N. Tishby. Selective sampling using the query by committee algorithm. *Machine learning*, 28(2):133–168, 1997.

[6] H. Grimmett, R. Paul, R. Triebel, and I. Posner. Knowing when we dont know: Introspective classification for mission-critical decision making. In *Proc. IEEE Intern. Conf. on Robotics and Automation (ICRA)*, 2013.

[7] A.J. Joshi, F. Porikli, and N. Papanikolopoulos. Multi-class active learning for image classification. In *Computer Vision and Pattern Recognition (CVPR)*, pages 2372–2379, 2009.

[8] A. Kapoor, K. Grauman, R. Urtasun, and T. Darrell. Gaussian processes for object categorization. *International Journal of Computer Vision*, 88(2):169–188, 2010.

[9] N. D. Lawrence, M. Seeger, and R. Herbrich. Fast sparse gaussian process methods: The informative vector machine. *Advances in neural information processing systems*, 15:609–616, 2002.

[10] N. D. Lawrence, J. C. Platt, and M. I. Jordan. Extensions of the informative vector machine. In *Proceedings of the First international conference on Deterministic and Statistical Methods in Machine Learning*, pages 56–87. Springer-Verlag, 2005.

[11] A. McCallum and K. Nigam. Employing em in pool-based active learning for text classification. In *Proceedings of ICML-98, 15th International Conference on Machine Learning*, pages 350–358, 1998.

[12] Robotics Centre of Mines ParisTech. Traffic lights recognition (TLR) data set.

[13] R. Paul and P. Newman. Self help: Seeking out perplexing images for ever improving navigation. In *Robotics and Automation (ICRA), 2011 IEEE International Conference on*, pages 445–451. IEEE, 2011.

[14] J. Platt. Probabilistic outputs for support vector machines and comparison to regularize likelihood methods. In A.J. Smola, P. Bartlett, B. Schoelkopf, and D. Schuurmans, editors, *Advances in Large Margin Classifiers*, pages 61–74, 2000.

[15] C. E. Rasmussen and C. K. I. Williams. *Gaussian processes for machine learning*. MIT Press, 2006.

[16] B. Settles. Active learning literature survey. Computer Sciences Technical Report 1648, University of Wisconsin–Madison, 2010.

[17] S. Tellex, P. Thaker, R. Deits, T. Kollar, and N. Roy. Toward information theoretic human-robot dialog. In *Robotics: Science and Systems*, 2012.

[18] S. Tong and D. Koller. Support vector machine active learning with applications to text classification. *The Journal of Machine Learning Research*, 2: 45–66, 2002.

[19] A. Torralba, K. P. Murphy, and W. T. Freeman. Sharing visual features for multiclass and multiview object detection. *IEEE Trans. Pattern Anal. Mach. Intell.*, 29(5):854–869, 2007. ISSN 0162-8828.

[20] R. Triebel, H. Grimmett, R. Paul, and I. Posner. Introspective active learning for scalable semantic mapping. In *Workshop. Robotics Science and Systems (RSS)*, June 2013.

[21] C. Urmson et al. Autonomous driving in urban environments: Boss and the Urban Challenge. *J. of Field Robotics*, 25(8):425–466, 2008. ISSN 1556-4967.

E Knowing When We Don't Know: Introspective Classification for Mission-Critical Decision Making

Authors: Hugo Grimmett, Rohan Paul, Rudolph Triebel and Ingmar Posner

Link to Table of Contents.

Knowing When We Don't Know: Introspective Classification for Mission-Critical Decision Making

Hugo Grimmett Rohan Paul Rudolph Triebel Ingmar Posner
Mobile Robotics Group, University of Oxford, UK
{hugo, rohanp, rudi, ingmar}@robots.ox.ac.uk

***Abstract*— Classification *precision* and *recall* have been widely adopted by roboticists as canonical metrics to quantify the performance of learning algorithms. This paper advocates that for robotics applications, which often involve mission-critical decision making, good performance according to these standard metrics is desirable but *insufficient* to appropriately characterise system performance. We introduce and motivate the importance of a classifier's *introspective* capacity: the ability to mitigate potentially overconfident classifications by an appropriate assessment of how qualified the system is to make a judgement on the current test datum. We provide an intuition as to how this introspective capacity can be achieved and systematically investigate it in a selection of classification frameworks commonly used in robotics: support vector machines, LogitBoost classifiers and Gaussian Process classifiers (GPCs). Our experiments demonstrate that for common robotics tasks a framework such as a GPC exhibits a superior introspective capacity while maintaining commensurate classification performance to more popular, alternative approaches.**

I. Introduction

The semantic mapping of a robot's workspace has become a popular line of research in recent years. A rich body of work now exists in which semantic labels are generated based on a variety of sensor modalities and classification frameworks (see, for example, [1]–[7]). Often, this is done with an implicit understanding that the application is agnostic to the classification method used: after all, for a number of classification frameworks the resulting *precision* and *recall* — quantities commonly used to characterise performance — are often commensurate across a wide variety of applications.

Contrary to this now established status-quo, we advocate that high precision and recall are desirable but do not suffice to fully characterise classification performance in robotics. The dimension missed is that spawned by a robot's ability to take action in ambiguous situations. For example, the robot may query a human operator or seek additional data for disambiguation rather than committing to a potentially incorrect class decision. Crucially, and central to this paper, this requires the classifier output to reflect an amount of ambiguity appropriate to a given situation. Even when hard class assignments are avoided by optimising an expected cost or reward, as is the case for most mission-critical decision making, a *realistic* estimate of uncertainty when modelling the state of the world is crucial; an autonomous car that misses a single traffic light with high confidence can suffer disastrous consequences (see, for example, Fig. 1).

We argue that a classifier which is uncertain when it makes mistakes but certain when classification is correct, is more desirable than a classifier which makes correct and incorrect decisions with similarly high confidence. We are therefore looking for a classifier's capacity to mitigate its assessment by an appropriate measure as to how 'qualified' it is to make a call given its own prior experience, usually in the form of training data. Following classical decision theory (e.g. [8]), mistakes are penalised by means of a loss function. However, if the underlying classification framework leads to an overconfident estimate of the class label, then it will often be ineffective regardless of the high costs imposed. Our work investigates this *introspective* capacity in a number of classification frameworks commonly used in robotics: support vector machines (SVMs), LogitBoosting and Gaussian Process classifiers (GPCs). Our treatment and findings apply to any aspect of robotics where action is required based on inference driven by raw sensor data. Here we choose to frame our exposition in the domain of autonomous driving, where mission-critical decisions equate to *safety*-critical decisions. To the best of our knowledge this is the first work in robotics characterising the introspective properties of commonly used classification frameworks.

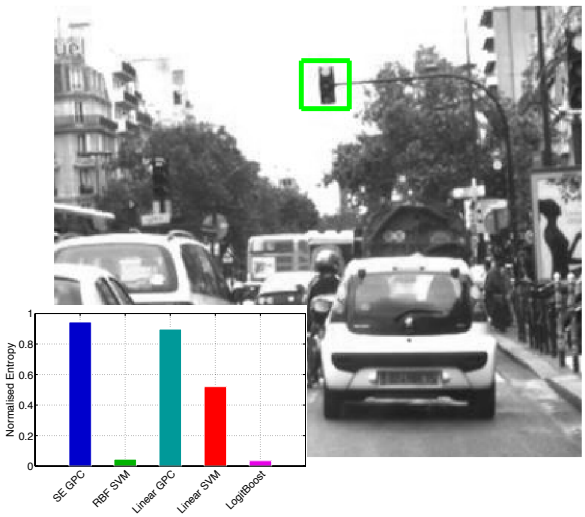


Fig. 1: Uncertainty in classification output as measured using normalised entropy for traffic light detectors based on five different classification frameworks applied to the window shown in green. Note that *all classifiers incorrectly* label this window as *background* (class decisions are not shown). However, the GPC variants do so with a significant amount of uncertainty while the others are inappropriately overconfident. Mission-critical decisions based on overconfident output will lead to catastrophic failure while an appropriately high amount of uncertainty when committing a mistake allows for remedial action to be taken. Providing this more germane output is the introspective quality we seek.

II. Related Works

For a number of years now robots have routinely generated and consumed higher-order abstractions from raw sensor data. Successful applications are as diverse as the detection of ground traversability (e.g. [9]), the detection of lanes for autonomous driving (e.g. [10]), the consideration of classifier output to guide trajectory planning and exploration (see, for example, [11], [12]) or the active disambiguation of human-robot dialogue [13]. These works commonly exploit classification output on a model-trust basis: systems are optimised with respect to precision and recall and egregious misclassifications — including vastly over-confident marginal distributions obtained from some classification frameworks — are accepted as par for the course. However, the suitability of the classification framework employed with respect to its introspective capacity has not previously been considered in robotics. Thus, we consider motivating, defining and investigating introspection in a robotics context to be the primary contribution of our work.

The concept of introspection as introduced here is closely related to considerations in active learning, where uncertainty estimates and model selection steps are often employed to guide data selection and gathering for an incremental learning algorithm. Kapoor *et al.* [14], for example, present an active learning framework for object categorization using a GPC where classifications of large uncertainty (as judged by posterior variance) lead to a query for a ground-truth label and are subsequently used to improve classification performance. Joshi *et al.* [15] address multi-class image classification using SVMs and propose criteria based on entropy and best-versus-second-best measures (see Section III) for disambiguating uncertain classifications. Holub *et al.* [16] propose an information-theoretic criterion that maximises expected information gain with respect to the entire pool of unlabeled data available. Hospedales *et al.* [17] discuss optimising rare class discovery and classification using a combination of generative and discriminative classifiers.

Our treatment of introspection is further informed by an ongoing discussion in the machine learning community regarding how to best account for variance in the space of feasible classifier models when training on, essentially, an incomplete set of data. For example, Tong and Koller [18] present an incremental algorithm for text classification using SVMs and the notion of a *version space*, the set of consistent hyperplanes separating the data in a feature space induced by the kernel function. Zhang et al. [19] introduce a max-margin classifier achieving better generalisation to unseen test data given a limited training corpus. Here, distinctiveness of training instances is assessed using the local classification uncertainty. A global classifier then incorporates these uncertainties as margin constraints, yielding a classifier that places less confusing instances farther away from the global decision boundary. We share the intuition that accounting for variance in version space when selecting a model leads to an increased introspective capacity. As a secondary contribution, therefore, our results serve to further corroborate this intuition.

III. Introspection and Uncertainty

Introspection concerns not the final class decision but rather the confidence with which this decision is made. The concept is motivated by the desire to take appropriate action when a classifier indicates high uncertainty. Our approach to introspection is grounded in the fact that the often cited assumption of independent and identically distributed (*iid*) training and test data is routinely violated in robotics: in the limit of continuous operation in the real world, one-shot classifier training is unlikely to be performed on a complete (or even fully representative) set of data.

Let a classifier map an input $\mathbf{x} \in \mathbb{R}^d$ to one of a set of classes $\mathcal{C} = \{C_1, \ldots, C_{|\mathcal{C}|}\}$ via an associated label $y \in \mathcal{C}$. Prior to training, domain specific knowledge is often used to constrain the family of classification models employed (for example in the form of a kernel, a covariance function or a type of base classifier). Classifier training then involves learning a set of (hyper-) parameters given a training dataset $\{\mathcal{X}, \mathcal{Y}\}$, where $\mathcal{X} = \{\mathbf{x}_1, \ldots, \mathbf{x}_{|\mathcal{X}|}\}$ denotes the set of feature vectors and $\mathcal{Y}$ denotes the set of corresponding class labels. The training data implicitly give rise to a probability distribution over the set of all possible models within the chosen family, $\mathcal{M}$, such that

$$\{\mathcal{X}, \mathcal{Y}\} \rightarrow p(m \mid \mathcal{X}, \mathcal{Y}), \quad m \in \mathcal{M}. \tag{1}$$

With a slight abuse of notation, m here denotes any member of the family of possible models, $\mathcal{M}$. In reality it is a function of the datum evaluated. In the following we make this relationship explicit by conditioning on both a model (or family of models) as well as on a test datum $\mathbf{x}_*$. Typically, training leads to the selection of a *single* model, $\tilde{m}$ from $\mathcal{M}$ such that a prediction y_* for a new, unseen feature vector $\mathbf{x}_*$ is obtained by approximating

$$p(y_* \mid \mathcal{X}, \mathcal{Y}, \mathbf{x}_*) \approx p(y_* \mid \tilde{m}, \mathbf{x}_*), \quad \tilde{m} \in \mathcal{M}. \tag{2}$$

This is illustrated in Figure 2(a). Common examples of this type of classification framework include SVMs and Boosting classifiers, where an optimisation is performed to select the best model given the training data (see Section IV). The *iid* assumption here is inherent since it is assumed that $\tilde{m}$ remains the best model for all predictions of unseen data. Breaking this assumption therefore often renders the chosen model suboptimal.

An alternative to the single model approach are classification frameworks which take into account the *entire set* of possible models in the specified family, such that

$$p(y_* \mid \mathcal{X}, \mathcal{Y}, \mathbf{x}_*) \approx p(y_* \mid \mathcal{M}, \mathbf{x}_*). \tag{3}$$

This case is illustrated in Figure 2(b). Here the shading indicates the distribution $p(m \mid \mathcal{X}, \mathcal{Y})$ with darker shades indicating increased probability. To aid intuition, predictions of four randomly selected members of $\mathcal{M}$ are also illustrated. Final predictions are made by taking into account opinions from all members of $\mathcal{M}$, often via the computation of an expectation such as for a GPC (see Section IV). Crucially, when considering an expectation over all of $\mathcal{M}$, the increased variance in feasible (and therefore likely) models at a distance from the training data leads to a moderation of the class predictions. This is the introspective quality we seek.

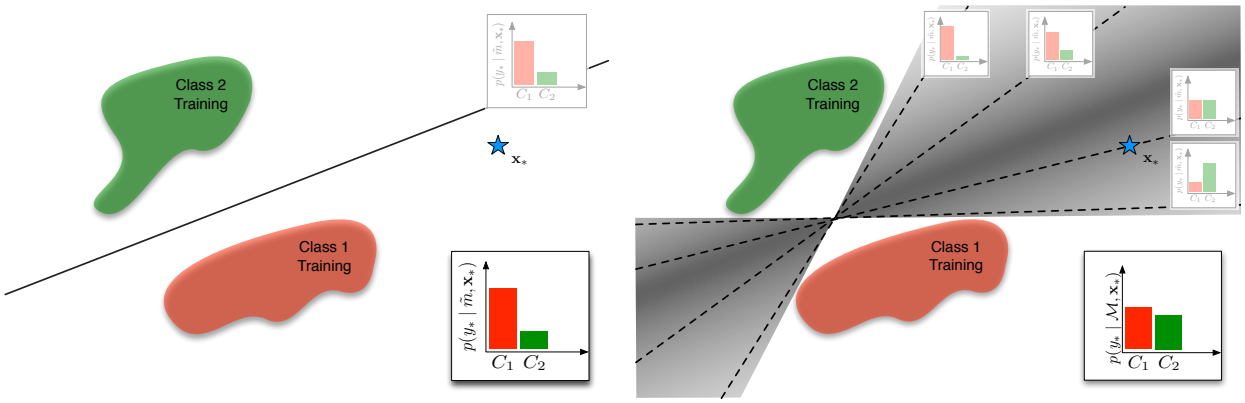


Fig. 2: An illustration of the two types of classification frameworks considered: (a) during training a *single* model is selected to classify an unknown datum $\mathbf{x}_*$ some way removed from the training data; (b) training leads to a distribution over models which is considered entirely to arrive at the final prediction. This illustration is for the family of linear models (indicated by solid (a) and dashed (b) lines). Each predictor is further annotated with its individual prediction. The overall predictive distribution is shown in the bottom right of each subplot. The shading in part (b) indicates the probability weights associated with individual models. Darker regions contain more weight. Note that the overall predictive distribution in (a) stems from the single model used and is, in this case, inappropriately confident. In part (b), however, the overall predictive distribution is moderated by computing the expectation over all models. This gives rise to a much more appropriate uncertainty estimate — the introspective quality we seek. (Best viewed in colour.)

A. *Quantifying Introspection*

In order to characterise the introspective capacity of a classification framework a transferable measure of the inherent uncertainty in the classification output is required. For this purpose, we use an information-theoretic quantity known as normalised entropy, H_N, defined as

$$H_N \quad = \quad -\sum_{C_i \in C} p(y = C_i \mid \mathbf{x}) \log_{|C|} \left[p(y = C_i \mid \mathbf{x}) \right]. \qquad (4)$$

This is equivalent to the Shannon entropy measure normalised by its maximum, which is the entropy of the $|C|$-dimensional uniform distribution, $\log(|C|)$. The result is a measure ranging between 0 and 1 where a *higher* value indicates *greater* uncertainty in the classifier's belief.

An alternative uncertainty measure proposed in the active learning literature is the best-versus-second-best (BvSB) heuristic [15] calculated as the difference between the largest and the second largest class likelihood estimates. This measure attempts to characterise the reliability of the maximum likelihood estimate rather than encoding the shape of the full distribution over class labels. The BvSB and normalised entropy measures are closely related in the binary-classification setting which is the case in this paper. We use normalised entropy throughout the remainder of this work due to its appealing information-theoretic interpretation.

IV. Classification Frameworks

We now present a brief overview of the specific classification frameworks considered in this work: SVMs, LogitBoost classifiers and GPCs. We focus on properties pertinent to introspection. Specifically, we describe the mechanism by which parameters are learned and how probabilistic output is obtained. For simplicity, but without loss of generality, this work considers predominantly binary classification such that $C = \{C_1, C_2\}$. For consistency we adhere to notation commonly found in the literature where a discriminant function is often denoted as $f(\cdot)$. We note that this is equivalent to a particular model m as described in the previous section.

A. *Support Vector Classification*

SVM classification is well established in robotics so that we provide here only an overview[1]. SVMs are based on a linear discriminant framework which aims to maximise the margin between two classes. The model parameters are found by solving a convex optimisation problem, thereby guaranteeing the final classifier to be the best feasible discriminant given the training data. Once training is complete, predictions on future observations are made based on the signed distance of the observed feature vector from the optimal hyperplane, such that

$$f(\mathbf{x}_*) = \sum_{i=1}^{N} \alpha_i y_i k(\mathbf{x}_i, \mathbf{x}_*) + b, \qquad (5)$$

where N is the size of the training set, α_i refers to a Lagrange multiplier associated with datum i, b denotes a bias parameter and $k(\mathbf{x}_i, \mathbf{x}_j)$ denotes the kernel function. Both α_i and b are obtained by training, and α_i is then non-zero only for *support vectors* $\mathbf{x}_i$. The kernel function amounts to a scalar product between two data, which have been transformed from d-dimensional feature space into some higher dimensional space. The nature of this mapping between spaces is inherent in the choice of kernel and need not be specified explicitly (the kernel trick). The regularization and kernel parameters are learnt using cross-validation. We discuss our choices of kernel functions in Section IV-D.

In its original form, the SVM classifier output is an uncalibrated real value. A common means of obtaining a probabilistic interpretation is Platt's method [21]. Here, using a hold-out set not used for classifier training, a parametric sigmoid model is fit directly to the class posterior

[1]For a detailed account the reader is referred to, for example, [20].

$p(y_* = C_1 \mid f(\mathbf{x}_*))$, such that

$$p(y_* = C_1 \mid f(\mathbf{x}_*)) \quad = \quad \frac{1}{1 + \exp(af(\mathbf{x}_*) + b)}. \qquad (6)$$

The sigmoid parameters a and b are chosen via cross-validation using a model-trust optimisation procedure. Note that class likelihoods are derived here using only a *single* estimate of the discriminative boundary obtained from the classifier training procedure. No other feasible solutions are considered. Hence, the predictive variance of the discriminant $f(\mathbf{x})$ is not taken into account while determining probabilistic output [22]. Further, no guarantees exist that the optimisation itself is well-behaved[2].

B. LogitBoosting Classifiers

Boosting is a widely used classification framework which involves training an ensemble of weak learners in sequence. The error function used to train a particular weak learner depends on the performance of the previous models [8]. Each weak learner, $h(\mathbf{x})$ is trained using a weighted form of the dataset in which the data weights depend on the performance of the previous classifiers. Predictions from a boosted classifier are obtained using a weighted combination of the individual weak learner outputs such that

$$\mathrm{sgn}(f(\mathbf{x}_*)) \quad = \quad \mathrm{sgn}\left(\sum_{i=1}^{M} w_i h_i(\mathbf{x}_*)\right), \qquad (7)$$

where M is the number of weak learners used.

LogitBoost [24] is a popular choice for a boosting-based classifier as it directly outputs class probability estimates. Weak learners are often chosen to be decision trees and training is conducted by fitting additive logistic regression models by stage-wise optimisation (using Newton steps) of the Bernoulli log-likelihood. The algorithm works in the logistic framework and yields a predictor function $f(\mathbf{x})$ learnt from iterative hypothesis training. Cross-validation is used to set hyper-parameters like the learning rate, tree depth, and the number of boosting rounds. The class-conditional probabilities are obtained from the predictor function as

$$p(y_* = C_1 \mid \mathbf{x}_*) \quad = \quad \frac{\exp(f(\mathbf{x}_*))}{\exp(f(\mathbf{x}_*)) + \exp(-f(\mathbf{x}_*))}. \qquad (8)$$

The procedure possesses asymptotic optimality as a maximum likelihood predictor [24], [25]. However, the method of converting the output of the predictor function to class-conditional probabilities is not fully probabilistic and does not account for variance in the underlying predictor function[3].

C. Gaussian Process Classification

Binary classification using a Gaussian Process (GP) [22], [26] is formulated by first introducing a *latent* function $f(\mathbf{x})$ and then applying a logistic function σ to obtain the prediction $p(y_* = C_1 \mid \mathbf{x}_*) = \sigma(f(\mathbf{x}_*))$. A GP prior is placed on the latent function $f(\mathbf{x}) \sim \mathcal{GP}(\mu(\mathbf{x}), k(\mathbf{x}, \mathbf{x}'))$ characterized by a *mean* function $\mu(\mathbf{x})$ and a *covariance* (or kernel) function $k(\mathbf{x}, \mathbf{x}')$. GPC training establishes values for the hyper-parameters specifying the kernel function k by maximising the log marginal likelihood of the training data.

Probabilistic predictions for a test point are obtained in two steps. First, the distribution over the latent variable corresponding to the test input is obtained using Equation (9). Here, $p(f \mid \mathcal{X}, \mathcal{Y}) = p(\mathcal{Y} \mid f)p(f \mid \mathcal{X})/p(\mathcal{Y} \mid \mathcal{X})$ is the posterior distribution over latent variables.

$$p(f_* \mid \mathcal{X}, \mathcal{Y}, \mathbf{x}_*) = \int p(f_* \mid \mathcal{X}, \mathbf{x}_*, f)p(f \mid \mathcal{X}, \mathcal{Y})df. \qquad (9)$$

This is followed by *marginalising* over the latent f_* to yield the class likelihood $p(y_* = C_1 \mid \mathcal{X}, \mathcal{Y}, \mathbf{x}_*)$ as

$$p(y_* = C_1 \mid \mathcal{X}, \mathcal{Y}, \mathbf{x}_*) = \int \sigma(f_*)p(f_* \mid \mathcal{X}, \mathcal{Y}, \mathbf{x}_*)df_*. \qquad (10)$$

Exact inference is analytically intractable due to the non-Gaussian logistic likelihood function. Instead we leverage expectation propagation (EP) [27], a method widely used for this purpose.

The GPC framework offers two key benefits over the other approaches discussed here [22]. Firstly, the classification output has a clear probabilistic interpretation as it directly results in the class likelihood. In contrast, neither the SVM nor the Boosting framework provide inherently probabilistic output but instead estimate a suitable calibration. Secondly, and crucially, the GP formulation addresses uncertainty or *predictive variance* in the latent function $f(\mathbf{x})$ via *marginalisation* (or averaging) over all models induced by the training set resulting in the estimate $p(y_* = C_1 \mid \mathcal{X}, \mathcal{Y}, \mathbf{x}_*)$ from Equation (10)[4]. Again this is in contrast to the SVM or Boosting estimate $p(y = C_i \mid \hat{f}, \mathbf{x}_*)$ that rely on a single discriminant estimate $\hat{f} : \mathcal{X} \rightarrow \mathcal{Y}$ learnt via minimisation. In the context of introspection, the ability to account for predictive variance is a key advantage of generative classification approaches[5].

D. Kernel Types

Evaluation of the discriminant function for an SVM and the covariance matrix for GPC inference both require the specification of a kernel function, $k(\cdot, \cdot)$. A rich body of literature exists on different choices of kernels for both frameworks. However, since our focus here is on a like-for-like comparison of different classification frameworks we choose two representative kernels rather than performing exhaustive model selection to optimise performance for a particular application. Firstly, as an example of the simplest kernel function available, we consider the linear kernel defined as

$$k_{LIN}(\mathbf{x}_i, \mathbf{x}_j) \quad = \quad \mathbf{x}_i^T \mathbf{x}_j + c, \qquad (11)$$

where c is a constant real number. The linear kernel is an apt choice where a linear separation of the data in feature space leads to adequate performance or where computational

[2]Throughout this work we use LIBSVM [23] for SVM training, calibration and testing.

[3]Throughout this work we use the Matlab implementation of LogitBoost for classifier training and testing.

[4]This process also gives rise to the well known property of increased variance while far away from the data in GP regression.

[5]Throughout this work we use the GPML toolbox [28] for GPC training and testing.

efficiency is of the essence. Often, however, a more sophisticated, non-linear kernel is required. In this category we use the *squared exponential* (SE) function as a canonical representative. The SE kernel with length scale parameter l is defined as

$$k_{SE}(\mathbf{x}_i, \mathbf{x}_j) = \exp\left(-\frac{1}{2l^2}\|\mathbf{x}_i - \mathbf{x}_j\|^2\right). \quad (12)$$

In the context of an SVM, the SE function is more commonly known as a *radial basis function* (RBF). In the following we will adhere to convention and refer to SE GPCs and RBF SVMs.

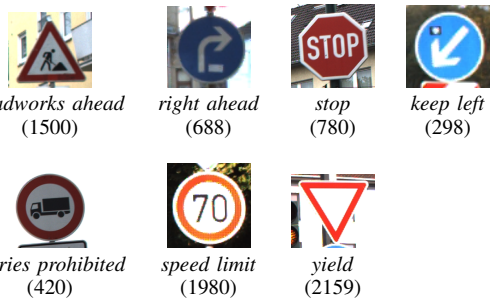
V. Experimental Results

Our experiments investigate the introspective capacity of the classifiers introduced in Section IV in an autonomous driving setting. Specifically, we focus on the *classification* of road signs and the *detection* of traffic lights. In investigating both classification and detection we aim to address the full spectrum of applications commonly encountered in robotics. The two are distinct in that classification addresses the case where a decision is made between two, well-defined classes (e.g. two types of traffic signs) and investigates classifier performance as a third, previously unseen class is presented. The detection case is arguably the more common one in semantic mapping where a single class is separated from a broad (in terms of intra-class variation) *background* class. Here, the concept of a previously unseen class does not exist but the inherent assumption is that the data representing the background class are sufficiently representative to capture any non-class object likely to be encountered. In practice, this is often not the case, leading to a significant number of misclassifications. While it could be argued that this problem is ameliorated somewhat by expanding the dataset used for training, we propose that the complexity of the workspaces encountered during persistent, long-term autonomy will keep perplexing even the most rigorously trained classifier.

A rich body of work on the detection and classification of road signs and traffic lights has established a successful track record of template-based features for this purpose. Specifically, we leverage the approach proposed by Torralba *et al.* [29] in which a dictionary of partial templates is constructed, against which test instances are matched. A single feature consists of an image patch (ranging in size from 8×8 to 14×14 pixels) and its location within the object as indicated by a binary mask (32×32 pixels). For any given test instance, the normalised cross-correlation is computed for each feature in the dictionary. Therefore, per instance (or window, in the detection case) a feature vector of length d is obtained, where d is the size of the dictionary. We found empirically that $d > 200$ leads to negligible performance increase in classification. Throughout our experiments we therefore set $d = 200$.

A. Introspection in Classification

This section investigates classification output when a third, previously unseen class is presented to the classifier. As examples of classes typically encountered in autonomous driving applications we use a subset of the German Traffic Sign Benchmark (GTSRB) dataset [30], which comprises over 50,000 loosely-cropped images of 42 classes of road signs, with associated bounding boxes and class labels. From this dataset we specifically focus on the seven classes shown in Table I. We arbitrarily select two classes for training: *stop* and *lorries prohibited*. To investigate the efficacy of the features used and training procedures employed, classifiers were trained separating these two classes using a balanced training set of 400 data (200 per class) and applying a canonical training procedure for each classifier type, including five-fold cross-validation where appropriate. Classifier performance was evaluated using standard metrics on a hold-out set of another 400 class instances (200 of each class) of the same two classes. The results are shown in Table II. Classification performance is commensurate across all classifiers. The corresponding precision-recall curve confirms the near-perfect separation of the classes and has been omitted here as it is otherwise uninformative. The classifiers are next retrained using the full 800 training data (400 per class) and the same canonical training procedures. They are then applied to 500 instances of the previously unseen class *roadworks ahead*. The resulting normalised entropy histograms are shown in Figure 3. The mean normalised entropies for the GPC-based classifiers are significantly higher than those of the other classification frameworks, indicating that the the GPC-based classifiers exhibit greater uncertainty in their judgement. Conversely, the RBF SVM and the LogitBoost classifier are extremely confident in their classifications with a very narrow distribution around a relatively low value of normalised entropy. This was an effect consistently observed throughout our experiments, which we attribute to the relatively gradual decay of the estimated class posterior probabilities through feature space often encountered far away from the decision boundary. Features from an unseen class which are located in feature space at a distance from the decision boundary



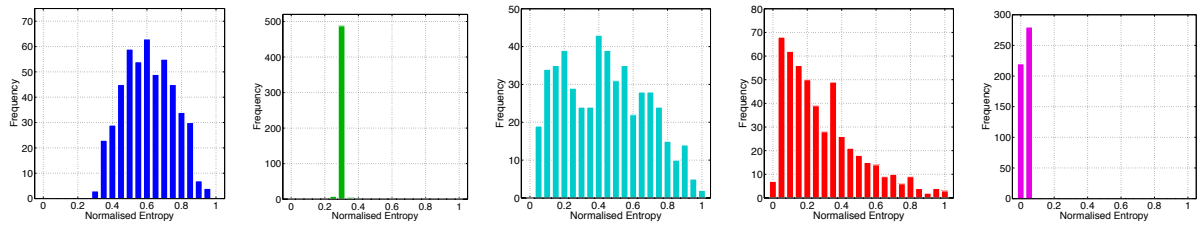
roadworks ahead (1500) | *right ahead* (688) | *stop* (780) | *keep left* (298)

lorries prohibited (420) | *speed limit* (1980) | *yield* (2159)

TABLE I: The seven classes of the German Traffic Sign Recognition Benchmark dataset considered in our work. The numbers in brackets indicate the number of data available per class.

Classifier	Precision	Recall	F_1
SE GPC	1.000	0.990	0.995
RBF SVM	1.000	0.995	0.997
Linear GPC	1.000	0.990	0.995
Linear SVM	1.000	0.990	0.995
LogitBoost	1.000	0.965	0.982

TABLE II: Classification performance when separating *stop* sign from the *lorries prohibited* signs. Note that different class combinations were found to yield classifiers of similar quality.



(a) SE GPC (b) RBF SVM (c) Linear GPC (d) Linear SVM (e) LogitBoost

Fig. 3: Normalised entropy histograms of the marginal probabilities for five classifiers trained on the road sign classes *stop* and *lorries prohibited* and tested on 500 instances of the unseen class *roadworks ahead*. Higher normalised entropy implies more uncertainty in classifier output. Note that the mean normalised entropy for the SE GPC is higher than that of the others.

Test Class	Classifier	Normalised Entropy	
		$\mu\pm$ std. err.	$\sigma\pm$ std. err.
	SE GPC	**0.504 ± 1.92E-03**	0.110 ± 9.35E-05
	RBF SVM	0.313 ± 1.33E-04	0.012 ± 2.12E-06
	Lin GPC	0.245 ± 9.34E-04	0.173 ± 9.19E-05
	Lin SVM	0.106 ± 4.77E-04	0.107 ± 2.51E-04
	Logit	0.015 ± 2.72E-05	0.009 ± 4.11E-05
	SE GPC	**0.487 ± 1.70E-03**	0.139 ± 7.67E-05
	RBF SVM	0.310 ± 1.13E-04	0.017 ± 3.72E-06
	Lin GPC	0.286 ± 8.09E-04	0.179 ± 6.24E-05
	Lin SVM	0.076 ± 3.72E-04	0.097 ± 2.43E-04
	Logit	0.012 ± 1.79E-05	0.007 ± 1.27E-05
	SE GPC	**0.723 ± 4.91E-04**	0.186 ± 1.59E-04
	RBF SVM	0.306 ± 1.03E-04	0.095 ± 7.96E-05
	Lin GPC	0.680 ± 4.75E-04	0.235 ± 1.11E-04
	Lin SVM	0.634 ± 7.29E-04	0.267 ± 4.28E-05
	Logit	0.021 ± 1.07E-04	0.031 ± 7.10E-04
	SE GPC	0.804 ± 6.08E-04	0.163 ± 1.72E-04
	RBF SVM	0.335 ± 1.43E-04	0.050 ± 1.26E-05
	Lin GPC	**0.811 ± 4.39E-04**	0.184 ± 1.66E-04
	Lin SVM	0.642 ± 3.24E-04	0.294 ± 9.19E-05
	Logit	0.017 ± 3.62E-05	0.018 ± 2.06E-04
	SE GPC	**0.259 ± 2.36E-03**	0.116 ± 1.37E-04
	RBF SVM	0.255 ± 1.28E-04	0.027 ± 5.26E-06
	Lin GPC	0.155 ± 9.27E-04	0.140 ± 2.61E-04
	Lin SVM	0.043 ± 7.82E-05	0.059 ± 1.26E-04
	Logit	0.007 ± 1.29E-07	0.007 ± 2.31E-05

TABLE III: Mean and standard deviation normalised entropies (including standard errors) from ten iterations of classifier training and testing, each with a randomly created dictionary and both training and test datasets resampled. Results are presented for classifiers trained on the road sign classes *stop* and *lorries prohibited* and tested on five different unseen classes as shown.

therefore only span a very narrow range of estimated class posterior probabilities.

In order to mitigate any influences of the specific feature set used and the specific training and test data selected we repeated the above experiment across a number of random dictionaries, data samples and unseen classes. Specifically, for each of five different unseen classes, we perform ten iterations of classifier training and testing with a random dictionary and training and test datasets resampled for each run. The results, presented in Table III, are consistent with those in Figure 3 in that the GPCs tend to be more uncertain while SVM and LogitBoost are more confident with an often significantly narrower distribution of normalised entropy values. The results in Table III indicate that the gap in uncertainty between the different frameworks is more pronounced for some unseen classes than for others. We attribute this to the varying degree of similarity in feature space between the unseen class and the classes in the training set. A more in-depth analysis of this phenomenon remains future work.

Classifier	Precision	Recall	F_1
SE GPC	0.976	0.909	0.941
RBF SVM	0.982	0.931	0.956
Linear GPC	0.970	0.912	0.940
Linear SVM	0.979	0.929	0.953
LogitBoost	0.963	0.928	0.945

TABLE IV: Performance on a holdout set of 2000 instances of classifiers trained on data from the TLR data set.

B. Introspection in Detection

We investigate the same classification frameworks as before on the task of traffic light detection. To this end we use the Traffic Lights Recognition (TLR) dataset [31], which is a sequence of colour images taken by a monocular camera from a car driving through central Paris. The TLR dataset comprises of just over 11,000 frames, where most of the traffic lights have been labeled with bounding boxes and further metadata such as the status of the signal or whether a particular label is ambiguous (e.g. the image suffers from motion blur, the scale is inappropriate or a traffic light is facing the wrong way). A few traffic lights have been omitted altogether. As suggested by the authors of [31], we exclude from our experiments any labels of class *ambiguous* or *yellow signal* and any instances which are partially occluded. We also remove any section of the sequence where the car is stationary and the lights are not changing. We split the dataset into two parts (at frame $7,200$ of $11,178$), with an approximately equal number of remaining labels in each part and with no physical traffic lights in common. Positive data are extracted as labeled. Negative *background* data are extracted by sampling patches of random size and position from scenes in the dataset while ensuring that the patches do not overlap with positive instances. The data are then split into training and test sets and classifiers are trained as before.

Again, we first verify the efficacy of the features selected and the training procedures employed. Table IV shows the classification performance for classifiers trained on 1,000 examples and evaluated on a hold-out set of 2,000 data. For completeness, Figure 4 shows the corresponding precision-recall curve. As before, classification performance according to conventional metrics is commensurate across all frameworks. In Figure 5, however, we demonstrate how the lack of introspection can impact classification performance when accept/reject decisions are guided by classification confidence. Specifically, we show the *cumulative* effect of accepting classifications below a given uncertainty threshold. First we note that when classifications are accepted at any level of uncertainty (i.e. up to and including unity normalised

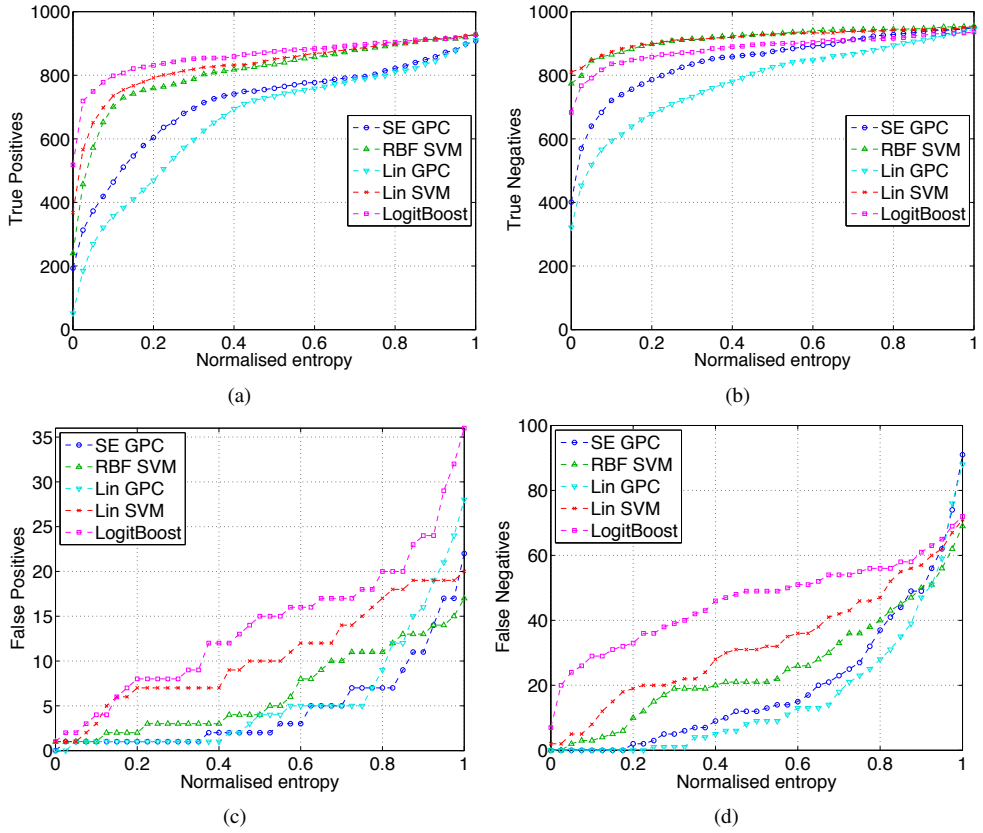


Fig. 5: Cumulative frequency plots of classification confusion (true positives, true negatives, false positives, and false negatives) against normalised entropy. The classifiers have been trained on 500 traffic lights against 500 background patches, and tested on 1,000 instances of each. Note that lower normalised entropy implies more certainty in classification. A more introspective classifier is one that exhibits higher uncertainty (as witnessed by larger normalised entropy in its output) when processing difficult instances. Consequently, class decisions on output above a given normalised entropy threshold are deferred since the output is deemed ambiguous. This is desirable since a single bad decision can have disastrous consequences. (Best viewed in colour.)

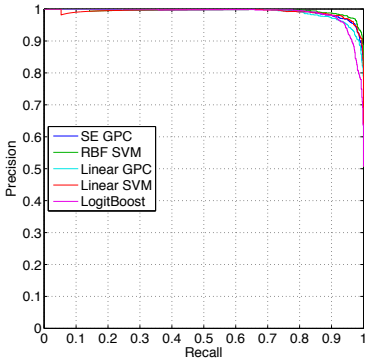


Fig. 4: Precision-recall graph for traffic light detection. Classifier performance is commensurate for all frameworks. (Best viewed in colour.)

entropy) all classification frameworks are commensurate in terms of true positives and true negatives (top row of Figure 5). This further corroborates the accuracy figures in Table IV. However, true positive and negative classifications occur generally at higher certainty (i.e. as normalised entropy tends to zero) for SVMs and LogitBoost classifiers than for the GPC variants. The latter are overall less certain about a significant number of correct classifications. The bottom row of Figure 5 indicates that SVMs and LogitBoost classifiers are also significantly more confident when *misclassifying* data (an example of this is also shown in Fig. 1). Significant numbers of mistakes are made at relatively low normalised entropy thresholds. The GPC variants, in contrast, accumulate comparable numbers of classification errors only at higher normalised entropy thresholds. The price paid for this more realistic assessment of the classification output is a reduction in correct classifications above the normalised entropy threshold. Note that this does not mean that subsequent samples are misclassified. It only implies that some other remedial action might be taken — for example obtaining label confirmation from a human or gathering otherwise additional data to aid disambiguation.

VI. Conclusions

This work demonstrates how performance metrics traditionally used in machine learning for classifier training and evaluation may be insufficient to characterise system performance in a robotics context, where a single misjudgement can have disastrous consequences. To remedy this shortcoming, we propose the concept of *introspection*: the ability to mitigate potentially overconfident classifications by a realistic assessment of predictive variance. Our experimental results imply that, despite commensurate performance as measured by more conventional metrics, GPCs possess a more pronounced introspective capacity than other classification frameworks commonly employed in robotics. We attribute this to their accounting, at test time, for predictive variance over the space of feasible classification models. This is in contrast to other commonly employed classification frameworks which often only consider a one-shot (ML or

MAP) solution. GPCs appear therefore better suited than the other frameworks investigated to applications where a realistic assessment of classification accuracy is required. Crucially, this includes many decision-making problems commonly encountered in robotics.

We have not, at this stage, considered the computational complexity of the approaches presented. Though GPCs in their basic form are notoriously expensive, more elaborate schemes exist which reduce the computational burden required for GPC inference. Our future work will investigate a variety of these schemes for suitability for real-time performance in autonomous driving tasks. Our work also holds implications for robotic active learning and exploration, which opens up additional avenues of research we intend to explore.

VII. Acknowledgements

This work is funded under the European Community's Seventh Framework Programme (FP7/2007-2013) under Grant Agreement Number 269916 and by the UK EPSRC Grant Number EP/J012017/1. We gratefully acknowledge advice from Ian Baldwin on parameter selection for boosting trees in Matlab. We thank Prof. Paul Newman for his support and encouragement for this work.

References

[1] D. Anguelov, B. Taskar, V. Chatalbashev, D. Koller, D. Gupta, G. Heitz, and A. Y. Ng, "Discriminative Learning of Markov Random Fields for Segmentation of 3D Scan Data." in *CVPR (2)*, 2005, pp. 169–176.

[2] O. Ḿartínez-Mozos, R. Triebel, P. Jensfelt, A. Rottmann, and W. Burgard, "Supervised semantic labeling of places using information extracted from sensor data," *Robot. Auton. Syst.*, vol. 55, no. 5, pp. 391–402, 2007.

[3] I. Posner, M. Cummins, and P. Newman, "A generative framework for fast urban labeling using spatial and temporal context," *Autonomous Robots*, 2009. [Online]. Available: http://dx.doi.org/10.1007/s10514-009-9110-6

[4] B. Douillard, D. Fox, and F. Ramos, "Laser and vision based outdoor object mapping," in *Proceedings of Robotics: Science and Systems IV*, Zurich, Switzerland, June 2008.

[5] A. Pronobis and P. Jensfelt, "Large-scale semantic mapping and reasoning with heterogeneous modalities," in *Intl. Conf. on Robotics and Automation*, 2012, pp. 3515 –3522.

[6] S. Sengupta, P. Sturgess, L. Ladick, and P. H. S. Torr, "Automatic dense visual semantic mapping from street-level imagery," in *Robotics and Autonomous Systems, IEEE International Conference on*, 2012.

[7] R. Paul, R. Triebel, D. Rus, and P. Newman, "Semantic categorization of outdoor scenes with uncertainty estimates using multi-class Gaussian process classification," in *Proc. of the International Conference on Intelligent Robots and Systems (IROS)*, 2012, to appear.

[8] C. Bishop, *Pattern recognition and machine learning*. springer New York, 2006, vol. 4.

[9] S. Thrun, M. Montemerlo, H. Dahlkamp, D. Stavens, A. Aron, J. Diebel, P. Fong, J. Gale, M. Halpenny, G. Hoffmann, K. Lau, C. Oakley, M. Palatucci, V. Pratt, P. Stang, S. Strohband, C. Dupont, L.-E. Jendrossek, C. Koelen, C. Markey, C. Rummel, J. van Niekerk, E. Jensen, P. Alessandrini, G. Bradski, B. Davies, S. Ettinger, A. Kaehler, A. Nefian, and P. Mahoney, "Stanley: The robot that won the DARPA Grand Challenge," *Journal of Field Robotics*, vol. 23, no. 9, pp. 661–692, 2006.

[10] A. Huang and S. Teller, "Probabilistic Lane Estimation using Basis Curves," in *Proceedings of Robotics: Science and Systems*, Zaragoza, Spain, June 2010.

[11] D. Meger, P. Forssén, K. Lai, S. Helmer, S. McCann, T. Southey, M. Baumann, J. Little, and D. Lowe, "Curious george: An attentive semantic robot," *Robotics and Autonomous Systems*, vol. 56, no. 6, pp. 503–511, 2008.

[12] J. Velez, G. Hemann, A. Huang, I. Posner, and N. Roy, "Planning to perceive: Exploiting mobility for robust object detection," in *Proc. ICAPS*, 2011.

[13] S. Tellex, P. Thaker, R. Deits, T. Kollar, and N. Roy, "Toward information theoretic human-robot dialog," in *Proceedings of Robotics: Science and Systems*, Sydney, Australia, July 2012.

[14] A. Kapoor, K. Grauman, R. Urtasun, and T. Darrell, "Gaussian processes for object categorization," *International Journal of Computer Vision*, vol. 88, no. 2, pp. 169–188, 2010.

[15] A. Joshi, F. Porikli, and N. Papanikolopoulos, "Multi-class active learning for image classification," in *Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on*. Ieee, 2009, pp. 2372–2379.

[16] A. Holub, P. Perona, and M. Burl, "Entropy-based active learning for object recognition," in *Computer Vision and Pattern Recognition Workshops, 2008. CVPRW'08. IEEE Computer Society Conference on*. IEEE, 2008, pp. 1–8.

[17] T. Hospedales, S. Gong, and T. Xiang, "Finding rare classes: Active learning with generative and discriminative models," *Knowledge and Data Engineering, IEEE Transactions on*, no. 99, pp. 1–1, 2011.

[18] S. Tong and D. Koller, "Support vector machine active learning with applications to text classification," *The Journal of Machine Learning Research*, vol. 2, pp. 45–66, 2002.

[19] W. Zhang, X. Stella, and Y. S. Teng, "Power svm: Generalization with exemplar classication uncertainty," in *Computer Vision and Pattern Recognition, 2012. CVPR 2012. IEEE Conference on*. IEEE, 2012.

[20] C. J. C. Burges, "A Tutorial on Support Vector Machines for Pattern Recognition," *Data Mining and Knowledge Discovery*, vol. 2, no. 2, pp. 121–167, 1998.

[21] J. C. Platt, "Probabilistic outputs for support vector machines and comparisons to regularized likelihood methods," in *Advances In Large Margin Classifiers*. MIT Press, 1999, pp. 61–74.

[22] C. Rasmussen and C. Williams, "Gaussian processes for machine learning. 2006," *The MIT Press, Cambridge, MA, USA*.

[23] C.-C. Chang and C.-J. Lin, "LIBSVM: A library for support vector machines," *ACM Transactions on Intelligent Systems and Technology*, vol. 2, pp. 27:1–27:27, 2011, software available at http://www.csie.ntu.edu.tw/~cjlin/libsvm.

[24] J. Friedman, T. Hastie, and R. Tibshirani, "Additive logistic regression: a statistical view of boosting," *Annals of Statistics*, vol. 28, p. 2000, 1998.

[25] T. Hastie and R. Tibshirani, *Generalized additive models*. Chapman & Hall/CRC, 1990.

[26] C. Williams and D. Barber, "Bayesian classification with gaussian processes," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 20, no. 12, pp. 1342–1351, 1998.

[27] T. Minka, "Expectation propagation for approximate bayesian inference," in *Proceedings of the Seventeenth conference on Uncertainty in artificial intelligence*. Morgan Kaufmann Publishers Inc., 2001, pp. 362–369.

[28] C. E. Rasmussen and H. Nickisch, "Gaussian processes for machine learning (GPML) toolbox," *J. Mach. Learn. Res.*, vol. 11, pp. 3011–3015, Dec. 2010. [Online]. Available: http://dl.acm.org/citation.cfm?id=1756006.1953029

[29] A. Torralba, K. P. Murphy, and W. T. Freeman, "Sharing visual features for multiclass and multiview object detection," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 29, no. 5, pp. 854–869, May 2007.

[30] J. Stallkamp, M. Schlipsing, J. Salmen, and C. Igel, "Man vs. computer: Benchmarking machine learning algorithms for traffic sign recognition," *Neural Networks*, no. 0, pp. –, 2012. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0893608012000457

[31] R. C. of Mines ParisTech, "Traffic lights recognition (TLR) data set," http://www.lara.prd.fr/benchmarks/trafficlightsrecognition.

F Reading the Road: Road Marking Classification and Interpretation

Authors: Bonolo Mathibela, Ingmar Posner, Paul Newman

Link to Table of Contents.

IEEE Transactions on Intelligent Transportation Systems

Reading the Road: Road Marking Classification and Interpretation

Journal:	*Transactions on Intelligent Transportation Systems and Intelligent Transportation Systems Magazine*
Manuscript ID:	T-ITS-14-03-0166
Manuscript Type:	Regular Papers
Date Submitted by the Author:	03-Mar-2014
Complete List of Authors:	Mathibela, Bonolo; University of Oxford, Information Engineering Posner, Ingmar; University of Oxford, Information Engineering Newman, Paul; University of Oxford, Information Engineering
Keywords:	Autonomous Vehicles, Scence Understanding, Intelligent Transportation Systems, Machine Learning
Abstract:	Road markings embody the rules of the road whilst capturing the upcoming road layout. These rules are diligently studied and applied to driving situations by human drivers who have read Highway Traffic driving manuals (road marking interpretation). An autonomous vehicle must however be taught to read the road, as a human might. This paper addresses the problem of automatically reading the rules encoded in road markings, by classifying them into seven distinct classes: single boundary, double boundary, separator, zig-zag, intersection, boxed junction and special lane. Our method employs a unique set of geometric feature functions within a probabilistic RUSBoost and Conditional Random Field (CRF) classification framework. This allows us to jointly classify extracted road markings. Furthermore, we infer the semantics of road scenes (pedestrian approaches and no drive regions) based on marking classification results. Finally, our algorithms are evaluated on a large real-life ground truth annotated dataset from our vehicle.

SCHOLARONE™
Manuscripts

PLEASE KEEP CONFIDENTIAL

Reading the Road: Road Marking Classification and Interpretation

Bonolo Mathibela, Ingmar Posner and Paul Newman

***Abstract*—Road markings embody the rules of the road whilst capturing the upcoming road layout. These rules are diligently studied and applied to driving situations by human drivers who have read Highway Traffic driving manuals (road marking interpretation). An autonomous vehicle must however be taught to read the road, as a human might. This paper addresses the problem of automatically reading the rules encoded in road markings, by classifying them into seven distinct classes: single boundary, double boundary, separator, zig-zag, intersection, boxed junction and special lane. Our method employs a unique set of geometric feature functions within a probabilistic RUSBoost and Conditional Random Field (CRF) classification framework. This allows us to jointly classify extracted road markings. Furthermore, we infer the semantics of road scenes (pedestrian approaches and no drive regions) based on marking classification results. Finally, our algorithms are evaluated on a large real-life ground truth annotated dataset from our vehicle.**

***Index Terms*—Road marking classification, scene understanding, scene interpretation, situational awareness**

I. Introduction

For life long autonomous driving to occur, there needs to be some degree of situational awareness on the part of the robotic car, allowing it to safely navigate its environment. Formally, situational awareness is defined as a three level process involving "the perception of elements in the environment within a volume of time and space, and the comprehension of their meaning, and the projection of their status in the near future" [1]. In the context of autonomous driving, the ultimate goal of situational awareness is to assist in informative decision making.

We approach situational awareness in terms of interpreting the underlying meaning of road markings, thus allowing a vehicle to *read the road* ahead, as shown in Figure 1. Here, reading the road refers to an online process of classifying and interpreting road markings along unknown roadways.

This greatly contrasts offline approaches (such as [2], [3]) where a semantic prior map (of road markings for example) is first created by surveying known roadways, and a vehicle localises against this map when traversing the same known roadway. The two main disadvantages of the offline approach is that the vehicle cannot interpret a new area without first creating an offline semantic map of that area; and roads are changing due to maintenance, construction and roadworks [4], [5]. Although roadworks are temporary, their frequency of occurrence is surprisingly high. In 2009/2010 alone, Transport for London reported an estimated 370 000 roadworks [6], a figure that is typical for most major cities. Thus our online approach to reading the road allows a vehicle to interpret any road scene without relying on offline semantic maps.

Manuscript received ...
Mobile Robotics Group, University of Oxford
{bonolo,ingmar,pnewman}@robots.ox.ac.uk



Figure 1: Reading the road requires classifying and interpreting the meaning of road markings in the road scene. Here, each colour represents a unique road marking class.

Highway Authorities worldwide publish technical reports detailing meanings and placements of road markings to aid human drivers in reading the road scene ahead. A driver can predict upcoming road topology, know the set of permissible manoeuvre and keep to the correct lane, based on interpreting road markings. An autonomous vehicle must however be taught to automatically read and interpret road semantics as a human might - this motivates our work.

This paper makes three contributions: Firstly, we propose a new probabilistic framework for classifying and interpreting road markings in images. Our framework accommodates a varying graphical model structure (each road scene gives rise to a unique structure). Secondly, we create a benchmarking ground truth class annotated dataset containing 2068 images and over 13099 unique annotations, the largest of its kind to date, to the best of our knowledge. Large road marking databases are hard to come by, apart from [7] which does not contain class labels (and has just 116 images) and [8] which has bounding box ground truth labels of painted road signs in 1208 images. Thirdly we show how traffic-light controlled pedestrian crossings and no-drive regions can be automatically detected from classified road markings.

For Review Only

Following a review of related works in the next section, we discuss our representation in Section III, our road marking classification framework in Section IV, evaluations in Section V, and finally conclude in Section VI.

II. Related Works

Road marking classification and interpretation is a problem that is of great interest to the intelligent transportation community. In earlier work, Charnonnier et al. extract parameters such as dash modulation and line width by representing lane markings as polygons of grouped detections; and could also detect arrow type road markings [9].

The simplest approaches to road marking classification use template matching, such as [10] for the recognition of crosswalks and arrows using single or repetitive rectangular pattern templates, and [11] for detecting arrow markings. Template matching approaches generally perform poorly when the road markings are partially observed or occluded.

Descriptor and Fourier-analysis based approaches have also been proposed before. Wu and Ranganathan's HOG-based (Histograms of Oriented Gradients) [12] road marking speed sign, arrow and word detection [8] algorithm uses Feature matching, followed by structural matching to capture the 2D geometry of FAST interest points. Collado et al.'s lane interpretation system for classifying continuous, discontinuous and merge type lanes [13] uses Fourier analysis on the intensity line profile of thresholded images for classification. Thresholds are applied to the resulting Fourier power spectrums within heuristically determined frequency bands. In later work [14] identify one-way, two-way or freeway roads based on Fourier analysis of road markings to classify them as continuous, broken or merge lines.

Prior geometric information has also been exploited before: zebra crossings and dashed lines were reconstructed in [15] where known 3D size and shape information of dashed lines and zebra crossing road markings was exploited. A signature-based recognition approach exploiting the alignment of edges with the road axis was used: projections of the edges along horizontal and longitudinal axes provide accumulation signature features [15] (e.g. repetitive peaks for zebra crossings versus a pair of peaks for dashed lines in the transverse axis) where the road axis is estimated to be the most frequently occurring orientation of 2D line edges. The system was tested on just 1km of data (150 stereo pair images).

In recent years, classifier based approaches have become popular, and examples include a decision tree classifier for intersection road markings trained on six numerical features based on relationships between the road marking border points and an extracted RANSAC line fit [16], moments to capture shapes of arrows and straight lines [17], and an Artificial Neural Network (ANN) classifier with invariant spatial moments, histogram projections and normalised angular measurement features for recognising 6 road marking arrow types and 17 alpha numeric characters [18]. Finally, a generative learning method based on eigenvectors was used by [19] to recognise arrow and numbered markings.



Figure 2: Zig-zag approach road markings lead up to a traffic light controlled pedestrian crossing. Classifying zig-zag approaches therefore provides strong priors for the location of traffic lights and pedestrians.

In the most related work, Duchow and Koertner detect individual lane segments and aggregate these into intersection lanes [20]. A machine learning approach is proposed where a set of features (width, length, orientation, position, intensity differences and deviations) is extracted from triplet groups of lane marking segments and a Support Vector Machine (SVM) is trained to classify lane segments as belonging to the same lane or not [20]. Triplets are then combined to form lane boundaries. Importantly, this approach requires no use of prior information (digital maps or geometric road models) and neighbourhood information is captured by considering triplets of road markings at a time. Although [20] considers only intersection classes, their work is the most related to ours as relationships between neighbouring road marking segments are exploited by the classification framework and no explicit geometric priors are used, similar to our approach.

Unlike previous work which focuses on finding arrows and painted traffic signs on roads [8], [11], [16]–[19], classifying solid and dashed lines [13], intersections [20] and zebra crossings [10], [15], to the best of our knowledge, we are the first to address the problem of road marking joint classification (where markings are classified together, relative to each other) and interpretation (deriving the semantics of road scenes from road markings) for situational awareness. Furthermore, our framework scales well with the number of road marking classes to be classified. Thus, with available data, additional classes can readily be incorporated into the classification framework.

III. Representation

Our goal is for a vehicle to read and interpret the rules of the road encoded in road markings in order to gain a sense of situational awareness of the current and upcoming road topology. For example, observing zig-zag approach road markings informs a vehicle that there is an upcoming traffic light controlled pedestrian crossing (see Figure 2). Successful recognition of zig-zag approaches therefore provides strong priors for the location of traffic lights and pedestrians, which is useful also for cuing the appropriate detector modules.

For Review Only



Class	Description	Abbreviation	Colour
1	Single boundary	SGL	Red
2	Double Boundary	DBL	Green
3	Separator	DIV	Magenta
4	Zig-zag	ZIG	Cyan
5	Intersection	INT	Yellow
6	Boxed Junction	BJ	Blue
7	Special Lane	SPL	Orange

Table I: Seven road marking classes we classify.

A. Seven Road Marking Classes

We classify the seven types of road markings shown in Table I and the accompanying figure. Here, three observations are made:

- The number of lanes influences the type of road markings present. We presently do not rely on having prior knowledge of the number of lanes present, however in future work this is a parameter we will learn from combining local vehicle observations with web-based sources.
- The class of a specific road marking is strongly influenced by that of its neighbours (notice how similar colours/classes are clustered). We therefore model the relationships (connectivity) between road markings based on proximity.
- The perspective effect in the images distorts the true lengths, orientations and positions of road markings. Our method thus begins by removing the perspective distortion to recover the true geometric properties of road markings.

B. Inverse Perspective Mapping (IPM)

As a pre-processing step, we begin by removing the perspective effect from images using Inverse Perspective Mapping [21]. Perspective in road images distorts true lengths, positions and orientations of road markings as shown in Figure 3. This effect is especially pronounced in zig-zag type markings (cyan colour) where the lengths of each road marking segment appear to be different when, in reality, they are the same length and one of two opposing orientations. To address this, the linear perspective can either be implicitly modelled [9], [16] or Inverse Perspective Mapping (IPM) can be performed [8], [17]–[20], [22] to create a perspective free bird's-eye-view image. We adopt the latter approach and compute a homography matrix $\mathbf{H} = \mathbf{KRK^{-1}}$, which defines the one-to-one relations between the camera image pixels and synthesised bird's-eye-view pixels. Here, $\mathbf{R}$ is the virtual rotation matrix and $\mathbf{K}$ is the camera's intrinsic parameters.

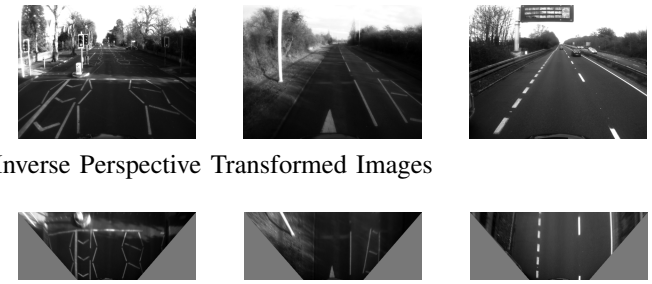


Figure 3: Inverse Perspective Mapping (IPM) to remove the perspective effect in images. Notice how, in the IPM images, parallel lines no longer intersect at the vanishing point. Thus true lengths, orientations and positions can be recovered.

C. Line Segment Representation

Assuming road markings have already been detected, a geometric model is assigned to each marking based on the system application. For lane detection and tracking applications, smoothness and continuity is an important consideration, thus spline [23] based geometric models are often used. We choose a line segment geometric model similar to [13], [20]. A painted road marking segment is therefore represented by a line segment (or series of line segments) parameterised by two endpoints p_A and p_B, where the midpoint locations, $p_C \triangleq \frac{(p_A+p_B)}{2}$, represent node locations for our road marking connectivity graphical structure. Thus we frame marking classification in terms of assigning the most likely road marking class to each node.

IV. Conditional Random Fields: Road Marking Classification

Conditional Random Fields (CRFs) are undirected probabilistic graphical models for relational learning, initially proposed by Lafferty et al. [24]. CRFs directly model the conditional distribution over hidden variables $\mathbf{x}$, given observations $\mathbf{z}$ as $p(\mathbf{x}|\mathbf{z})$. In the graphical model, each node represents the hidden states and observations. Thus, given a set of n hidden states $\mathbf{x} = \{\mathbf{x_1}, \mathbf{x_2}, \ldots, \mathbf{x_n}\}$ and data $\mathbf{z}$, the nodes $\mathbf{x_i}$ and their associated graph structure define the conditional distribution $p(\mathbf{x}|\mathbf{z})$ over the hidden states (see Figure 5). Each fully connected subgraph of this graph structure represents a clique in the graph. Let C be the set of cliques in the graph, then a CRF factorizes the conditional distribution $p(\mathbf{x}|\mathbf{z})$ into

For Review Only

a product of clique potentials $\phi_c(\mathbf{z}, \mathbf{x_c})$ where $c \in C$, and $\mathbf{x_c}$ and $\mathbf{z}$ are the hidden nodes and data of clique c, respectively. Each clique potential $\phi_c(\mathbf{z}, \mathbf{x_c})$ is a strictly positive function of the variable configuration and captures the compatibility amongst variables in the clique (thus large values imply a more likely configuration) [25]. The Conditional Random Field model can therefore be written as:

$$p(\mathbf{x}|\mathbf{z}) = \frac{1}{Z(\mathbf{z})} \prod_{c \in C} \phi_c(\mathbf{z}, \mathbf{x_c}) \quad (1)$$

where the normalisation factor (partition function) is $Z(\mathbf{z}) = \sum_x \prod_{c \in C} \phi_c(\mathbf{z}, \mathbf{x_c})$, which sums over all possible label assignments. In the log-linear parameterisation of CRFs, the potentials are described using a log linear combination of feature functions:

$$\phi_c(\mathbf{z}, \mathbf{x_c}) = \exp(\mathbf{w}_c^T \cdot \mathbf{f_c}(\mathbf{z}, \mathbf{x_c})) \quad (2)$$

where $\mathbf{w}_c^T$ is the weight vector we need to learn and $\mathbf{f_c}(\mathbf{z}, \mathbf{x_c})$ is a feature function extracting a vector of features in a clique. Here, large weights imply high compatibility and low weights a less likely configuration. Thus, the log-likelihood parameterisation of a CRF can be written as

$$p(\mathbf{x}|\mathbf{z}) = \frac{1}{Z(\mathbf{z})} \exp\left\{ \sum_{c \in C} \mathbf{w}_c^T \cdot \mathbf{f_c}(\mathbf{z}, \mathbf{x_c}) \right\} \quad (3)$$

where $Z(\mathbf{z}) = \sum_x \exp\left\{\sum_{c \in C} \mathbf{w}_c^T \cdot \mathbf{f_c}(\mathbf{z}, \mathbf{x_c})\right\}$ is the new partition function.

CRFs avoid encoding distributions over just the variable $\mathbf{z}$ which allows great flexibility in choosing a rich set of continuous observed variables (feature functions) whose dependencies may be difficult to model (no simple parametric form) or understand. We can therefore exploit domain knowledge to define a set of discriminative features without needing to model their joint distribution [26].

A. Graphical Structure

The graph structure (connectivity) captures relationships between road marking segments. In the Figure of Table I, we observe that road markings belonging to the same class tend to be spatially clustered into groups (notice how the same colours in the image appear clustered). This implies that adjacent road markings are likely to belong to the same class, thus nodes should be connected to their closest neighbours. We therefore use Delaunay triangulation (the nearest neighbour graph is a subgraph of Delaunay Triangulation [27]) to create a graph structure of road markings, assuming a locally planar road surface. Here, a unique graph structure exists per training example as illustrated by Figure 4.

B. Conditional Log Likelihood Training and Inference

In parameter estimation, we are given the training dataset of M examples and associated graphical structures. Our aim is to estimate the parameters $\mathbf{w}_c^T$ in Equation 3 that best

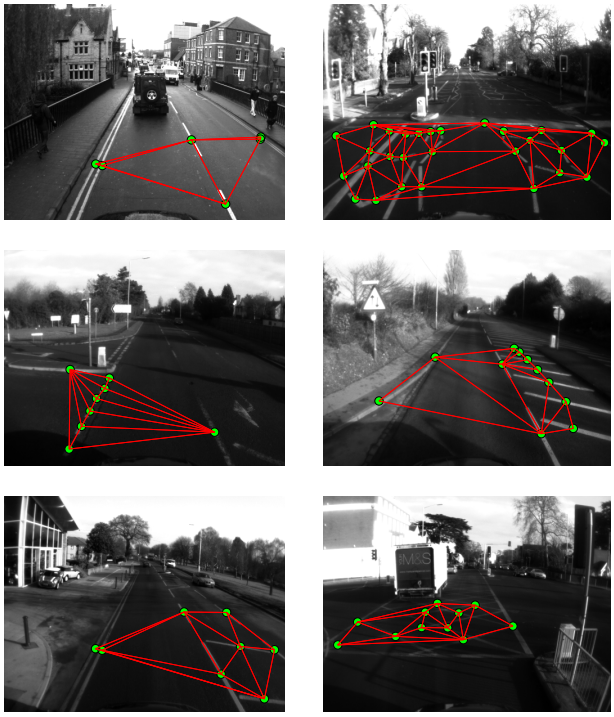


Figure 4: Road marking graph structures created using Delaunay Triangulation. Notice how each example is unique.

model the data. Let $\theta = [\mathbf{v}, \mathbf{w}]$ represent the combined parameters of the unary and binary features and $F(x, y)$ represent all the features. The CRF learns the weights discriminatively by maximising the conditional log likelihood of the training data. Specifically, the log likelihood for M training examples is given by:

$$\log(p(\mathbf{x}|\mathbf{z})) = \sum_{m=1}^{M} \theta^T F(\mathbf{z}_m, \mathbf{x}_m) - \sum_{m=1}^{M} \log \mathbf{Z}(\theta, \mathbf{z}_m) \quad (4)$$

This log likelihood function can be shown to be concave because $\log \mathbf{Z}(\theta, \mathbf{z})$ is concave [26]. No analytical form for this log likelihood maximum exists thus we perform parameter estimation using an L2-regularised log-likelihood and estimate the maximum using an iterative line search method (quasi-Newton approach) which takes steps in parameter space to improve the objective. L2-regularisation is necessary because as more features are added to the CRF model, the negative log-likelihood function decreases which could lead to over-fitting. The regularisation essentially adds a penalty for large parameter values [26].

Inference refers to estimating the marginal distribution of each hidden variable $\mathbf{x_i}$ or the most likely configuration of all hidden variables $\mathbf{x}$. These tasks rely on using belief propagation, a message passing algorithm for estimating marginal distributions over a graph [28]. In belief propagation, messages are passed through the graph structure of

For Review Only

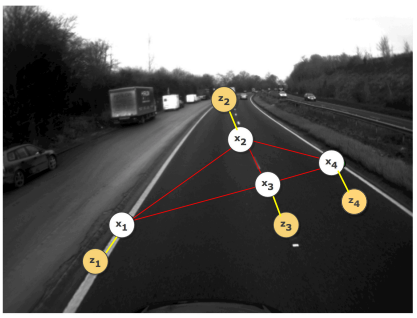


Figure 5: Our CRF model for road marking classification. The hidden states $\mathbf{x_i}$ are the unknown road marking classes and the observations $\mathbf{z_i}$ (shaded orange) correspond to line segment endpoints extracted from each road marking. This graph is a probability distribution p($\mathbf{x}|\mathbf{z}$) and each road image example gives rise to a unique graph.

the model and each node passes on received messages to its neighbours. Although this method is exact for trees, forests and other graphs not containing loops, our Delaunay triangulation connectivity creates loopy graphs, therefore we apply loopy belief propagation (LPB) [29], an approximate inference algorithm not guaranteed to necessarily converge to the true probability distribution (inference is intractable). We however found the performance of the LBP algorithm for inference in our CRF model to produce good results, even though convergence is not necessarily guaranteed.

C. Feature Functions

Feature functions for road marking classification describe geometric and spatial properties of line segments and their endpoints. The CRF learning algorithm associates an appropriate weight with each feature function given the training data. In CRFs, local (unary) features depend only on a single hidden state i. Global (binary) features depend on two hidden states i and j.

1) Unary Feature Functions : We formulate eleven geometric and spatial unary features capturing the properties of a single node. Let us denote the line segment endpoints for node i as p_{Ai} and p_{Bi}, the midpoint as p_{Ci}, and the x and y segment endpoints for node i as ($p_{Ai,x}$ and $p_{Bi,x}$) and ($p_{Ai,y}$ and $p_{Bi,y}$) respectively. Let $isint()$ be a function that returns 1 if there is an intersection and 0 otherwise, and l_i denote the line segment formed by joining p_A and p_B. Then the unary feature functions are:

Length: This feature measures the length of a line segment and is useful for separating typically long road markings (boundary, separator), medium length (zig-zag, special lane) and short markings (intersection and special lane).

$$f_{length}(i, \mathbf{z}_i) = \| p_{Ai} - p_{Bi} \| \quad (5)$$

X-midpoint location: This feature measures the spatial x location of the road marking in the image. This captures the fact that boundary or zig-zag type markings tend to be located towards the edges of the image whilst boxed junctions tend to take up the entire image. Similarly, special lanes and intersections are spatially located near the road boundaries.

$$f_{midx}(i, \mathbf{z}_i) = p_{Ci,x} \quad (6)$$

Y-midpoint location: For completion we also consider the y-midpoint location of the road marking. Although we removed the perspective distortion in the image, this feature is useful particularly at intersections and roundabouts where the overall orientation of road markings appears rotated.

$$f_{midy}(i, \mathbf{z}_i) = p_{Ci,y} \quad (7)$$

Delta-x: In addition to capturing the location and length of a road marking line segment, it is also important to have features that capture the regularities. The change in x coordinates (delta x) feature captures the regular pattern of zig-zag approaches. Recall how markings of the same length and one of two orientations denote zig-zags. The delta x (and delta y) feature is constant for zig-zag type markings. For vertical type markings (boundaries, special lanes etc.) this feature is close to zero.

$$f_{deltax}(i, \mathbf{z}_i) = \| p_{Ai,x} - p_{Bi,x} \| \quad (8)$$

Delta-y: This feature captures similar geometric properties to delta x. Specifically, for zig-zag markings, this feature is constant. For vertical markings it simply reduces to the approximate length of the road marking and for boxed junctions this feature varies depending on the typed of boxed junction.

$$f_{deltay}(i, \mathbf{z}_i) = \| p_{Ai,y} - p_{Bi,y} \| \quad (9)$$

Orientation: The orientation of a line segment is measured anticlockwise relative to the x-axis. This feature approximately evaluates to one of two values for zig-zag type markings and, with the exception of the boxed junction class, is approximately constant for road markings belonging to the same class in an image. It also roughly captures the vehicle orientation (which influences the relative orientation of the road markings).

$$f_{angle}(i, \mathbf{z}_i) = \angle(p_{Ai} - p_{Bi}, \text{x} - \text{axis}) \quad (10)$$

Furthest vertical distance: This feature is simply a measure of how far away from the top of the image a road marking is.

$$f_{vert}(i, \mathbf{z}_i) = \max(p_{Ai,y}, p_{Bi,y}) \quad (11)$$

Proportion segments to the left: This feature measures the percentage of road markings located to the left of the i^{th} node. This captures the overall distribution of road markings

For Review Only

in the image relative to the position of node i. Intuitively, we expect this class distribution to be different across classes with boundary type markings having low proportions and separator markings proportions close to 0.5.

$$f_{propleft}(i,j,\mathbf{z}_i) = \frac{1}{N}(\{(p_{left}) : \min_{j=1:N}(p_{Aj,x}, p_{Bj,x})\} < \min(p_{Ai,x}, p_{Bi,x})) \quad (12)$$

Proportion segments to the right: This feature measures the percentage of road markings located to the right of the i^{th} node. It captures the overall distribution of road markings in the image relative to the position of node i similar to the previous feature. The symmetry of the road markings on the road is also captured by this and the previous proportion features.

$$f_{propright}(i,j,\mathbf{z}_i) = \frac{1}{N}(\{(p_{right}) : \max_{j=1:N}(p_{Aj,x}, p_{Bj,x})\} > \max(p_{Ai,x}, p_{Bi,x})) \quad (13)$$

Closest midpoint distance: This feature is especially important for double boundary type road markings since they comprise of two road markings of similar length and orientation, running parallel. Thus the midpoint distance between double boundary road markings is small compared to the midpoint distances between other marking types.

$$f_{midist}(i,j,\mathbf{z}_i) = \parallel p_{Ci} - p_{Cj} \parallel \quad (14)$$

Number of intersecting line segments: This feature is most important for boxed junction markings occurring at intersections/junctions. This is because for other classes, mostly no intersections occur. The feature function counts the number of line segments j that intersect with the line segment associated with node i.

$$f_{int}(i,j,\mathbf{z}_i) = \sum_{j=1}^{N-1} isintersect(l_i, l_j) \quad (15)$$

Conditional Random Fields are able to incorporate continuous local features like the ones described above. CRF potentials are formed by exponentiating a linear function of the feature functions. To model the more complex non linear relationships between hidden states and feature values however, it is better to discretise (or binarise) the features [30].

2) RUSBoost Features : Instead of directly using the local features described in Section IV-C1 in our CRF model, we first train a Random Under Sampling Boost (RUSBoost) [31] classifier and use these scores as input to our CRF. This is a design choice shown to be effective by Friedman et al. [32] who proposed first learning AdaBoost classifiers from trained labeled data. Other variants of this approach, include using SVM scores and CRFs [30] to build a two stage classifier.

Specifically, we choose RUSBoost because it is robust to skewed training data - the number of examples, per class, in our datasets is not balanced. For example, separator markings occur five times more frequently than intersection type markings (see Table II).

Class	Description	Train	Proportion	Test	Proportion
1	Single Boundary	794	12.19%	778	11.81%
2	Double Boundary	1012	15.54%	961	14.59%
3	Separator	1491	22.89%	1471	22.34%
4	Zig-zag	523	8.03%	549	8.34%
5	Intersection	287	4.41%	395	6.00%
6	Boxed Junction	1566	24.04%	1573	23.89%
7	Special Lane	841	12.91%	858	13.03%

Table II: Class distribution of our road marking dataset showing the number and proportion of training and test examples

We train a RUSBoost classifier using the eleven features described in Section IV-C1. The classification output for each training example is a 7x1 vector assigning a score to each class. The RUSBoost classifier learns an independent model which does not account for relationships between adjacent nodes, thus we use this score vector as input features for our CRF which learns statistical dependencies between adjacent hidden states.

3) Binary Feature Functions : For consistency (relational) across the graphical model, it is necessary to define features relating hidden states in the CRF to each other. Such pairwise features take into account the effect that the feature functions have on the transitions from one state to the other (for example a neighbour node transitioning from a zig-zag to a boxed junction class). We derive edge (binary) features from the RUSBoost scores (new node features). Here, the edge features are taken to be the node features from both edge ends [33].

V. Experimental Results

We collected a benchmarking dataset containing 2068 hand labelled images spanning city, residential and motorway roads. The road marking dataset class distribution is shown in Table II where a skewed distribution is noted. Some classes, such as intersections occur less frequently than others such as separators.

We evaluate our system performance in two ways:

- without relational learning (similar to Duchow and Koerther [20])
- with relational learning (our proposed method)

Our road marking classification framework was implemented using the Undirected Graphical Models (UGM) Toolbox by Mark Schmidt [33] for probabilistic graphical models.

A. Results

1) No Relational Learning : The performance of our road marking classification framework without relational learning (only RUSBoost classifier) was evaluated using the datasets described above. Although the last three feature functions, $f_{propleft}$, $f_{propright}$, and f_{midist} take other road markings

For Review Only

PREDICTION

GROUND TRUTH	SGL	DBL	DIV	ZIG	INT	BJ	SPL
SGL	**0.878**	0.008	0.024	0.004	0.001	0.033	0.052
DBL	0.007	**0.85**	0.007	0.007	0.007	0.096	0.026
DIV	0.01	0	**0.867**	0.003	0.003	0.033	0.085
ZIG	0.003	0.007	0.01	**0.712**	0.053	0.154	0.061
INT	0	0.016	0.004	0.044	**0.558**	0.193	0.185
BJ	0.003	0.01	0.024	0.012	0.019	**0.89**	0.042
SPL	0.009	0.021	0.051	0.015	0.066	0.124	**0.713**

(a) Per class Precision Matrix (no relational learning)

PREDICTION

GROUND TRUTH	SGL	DBL	DIV	ZIG	INT	BJ	SPL
SGL	**0.955**	0.007	0.014	0.005	0.003	0.018	0.051
DBL	0.009	**0.952**	0.005	0.015	0.02	0.065	0.033
DIV	0.021	0	**0.933**	0.007	0.01	0.034	0.156
ZIG	0.003	0.005	0.005	**0.887**	0.091	0.067	0.049
INT	0	0.008	0.001	0.04	**0.709**	0.062	0.108
BJ	0.005	0.012	0.02	0.027	0.058	**0.703**	0.061
SPL	0.008	0.015	0.022	0.018	0.109	0.051	**0.542**

(b) Per class Recall Matrix (no relational learning)

Table III: Per class Precision and Recall Confusion Matrices For No Relational Learning.

PREDICTION

GROUND TRUTH	SGL	DBL	DIV	ZIG	INT	BJ	SPL
SGL	**0.932**	0.008	0.018	0	0.001	0.013	0.028
DBL	0.003	**0.9**	0.01	0.002	0.021	0.044	0.019
DIV	0.014	0.002	**0.879**	0.005	0.003	0.032	0.065
ZIG	0.002	0.004	0.005	**0.898**	0.051	0.032	0.009
INT	0	0.02	0.009	0.003	**0.792**	0.041	0.135
BJ	0.003	0.022	0.017	0.007	0.015	**0.906**	0.031
SPL	0.019	0.02	0.082	0.021	0.055	0.067	**0.736**

(a) Per class Precision Matrix (with relational learning)

PREDICTION

GROUND TRUTH	SGL	DBL	DIV	ZIG	INT	BJ	SPL
SGL	**0.941**	0.006	0.01	0	0.003	0.006	0.026
DBL	0.004	**0.928**	0.007	0.004	0.053	0.028	0.022
DIV	0.028	0.003	**0.915**	0.015	0.01	0.031	0.117
ZIG	0.001	0.002	0.002	**0.929**	0.073	0.011	0.006
INT	0	0.007	0.002	0.002	**0.686**	0.009	0.054
BJ	0.005	0.035	0.018	0.018	0.058	**0.879**	0.055
SPL	0.021	0.018	0.047	0.033	0.116	0.036	**0.721**

(b) Per class Recall Matrix (with relational learning)

Table IV: Per class Precision and Recall Confusion Matrices With Relational Learning.

into account, the relationships between adjacent node labels (or cliques) is not explicitly modelled. The resulting per class precision and recall confusion matrices are shown in Table III. For a given confusion matrix, the diagonal elements are large ratios (close to 1) for good classification performance (i.e. the ground truth and predicted classes correspond exactly). Thus, a strong bright diagonal in a confusion matrix illustrates good classification performance.

Our experiments resulted in per class precision values ranging between 0.56 and 0.89 and per class recall ranging from 0.54 to 0.96 as shown in Table III. The intersection class precision is very low because examples belonging to this class are commonly misclassified as special lanes, boxed junctions or sometimes zig-zag. Per class recall values for the special lane class are also very low because road markings predicted as separator, single boundary, zig-zag approach, intersection or boxed junction actually belong to special lane. The RUSBoost model is discriminative and assigns road marking classes based on a strong learner given the feature functions. We show in the next section how introducing a relational learning framework, which models that adjacent markings are likely to belong to the same class, reduces the misclassification errors.

2) Classification with Relational Learning: We further tested our system using our CRF relational learning framework. Here, relationships between adjacent road marking classes are explicitly captured. The resulting per class precision and recall confusion matrices are shown in Table IV - the per class precision values range between 0.74 and 0.93 whilst the per class recall ranges from 0.7 to 0.94. The precision is lowest for the intersection and special lane classes as their visual appearance is similar (see Figure 8) and the intersection class is under represented in the training data (4.41% of the training data). Similarly, the recall is lowest for the intersection and special lane classes. The remaining classes show very good performance with precision and recall values of between 0.88 to 0.94. We quantify the per class percentage improvement gained by adding relational information in the road marking classification framework in Table V, where performance improvements of up to 23% are achieved across classes.

Figure 6 shows classification results where each colour represents the class the road marking is assigned to. To better visualise the errors, we plot these same images in Figure 7 where green represents correctly classified road markings, and red lines show the misclassifications.

For Review Only

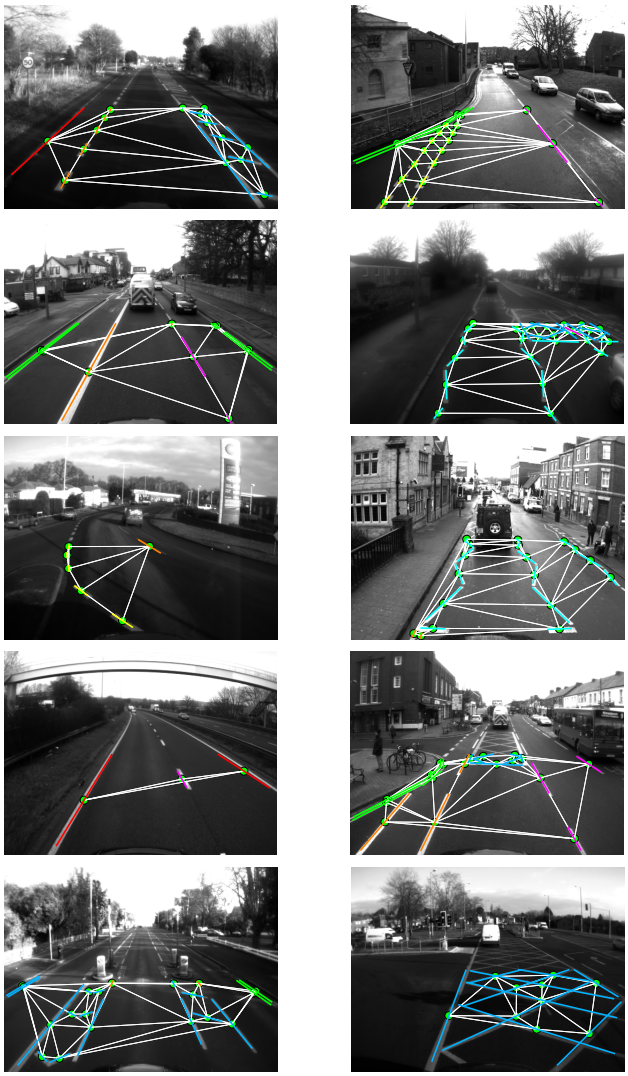


Figure 6: Classification results where each colour represents a different class (see Table I for colour code). The graphical model structure is superimposed on the image.



Figure 7: Classification results from our system (green lines are correctly classified, whilst red lines represent misclassifications). These images correspond to Figure 6.

Class	Name	Precision Change (%)	Recall Change (%)
1	SGL	4.9	-1.7
2	DBL	8	-2.2
3	DIV	2.5	-1.7
4	ZIG	16.7	5.1
5	INT	23.4	0.5
6	BJ	1.6	19.2
7	SPL	1.6	16.6

Table V: Percentage improvements in terms of precision and recall when relational learning is used.



Figure 8: Special lane and intersection type road markings have very similar visual appearance (short, repetitive).

PLEASE KEEP CONFIDENTIAL



Figure 9: Boxed junctions describe areas of the road that a vehicle should not be stationary in (but may possibly quickly traverse) such as separator boxed junctions (left) and intersection boxed junctions (right).

VI. Conclusions and Future work

In this paper, we presented a framework for situational awareness based on reading and interpreting road markings. Specifically, we classified road markings into seven distinct classes: single boundary, double boundary, separator, zig-zag, intersection, boxed junction and special lane. Our system achieved a precision of between 74% and 93% and corresponding recall of 70% to 94% across all classes. Overall, our system allows a vehicle to read and interpret the road by understanding the rules of the road encoded in the type of painted road markings visible.

In future work, we will investigate using the output of our road marking classification system for road scene semantic interpretation for situational awareness. To test this, we performed some preliminary experiments to detect traffic light controlled pedestrian crossings (zig-zag approaches) and no drive regions (boxed junctions - see Figure 9).

Here, we defined a zig-zag approach (or boxed junction) to exist if a proportion of over 0.7 of classified individual road marking segments belong to class zig-zag (or boxed junction). This threshold was determined empirically to capture our observation that zig-zag approaches (or boxed junction) occur when the vast majority of road markings in a road scene belong to that class. Our experimental results showed that, even with this straight forward approach, our classifier achieved 88% and 86% prediction accuracies for the zig-zag approach and boxed junction scenes respectively. Examples of correctly classified scenes are shown in Figures 10 and 11 respectively. A vehicle can therefore prepare for the possibility of encountering humans on the road, or areas where it may not be stationary (but may possibly momentarily drive through) - thus gaining a dynamic sense of situational awareness, like a human. Investigating this further forms the subject of our future work.

References

[1] M. Endsley, "Towards a Theory of Situation Awareness in Dynamic Systems," *Human Factors*, vol. 37, no. 1, pp. 32–64, 1995.

[2] A. Wimmer, T. Jungel, M. Glück, and K. Dietmayer, "Automatic generation of a highly accurate map for driver assistance systems in road construction sites," in *IEEE Intelligent Vehicles Symposium*, June 2010, pp. 281–286.

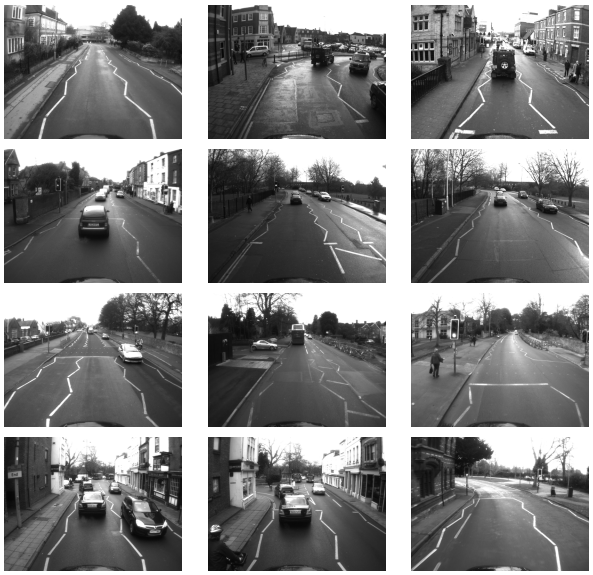


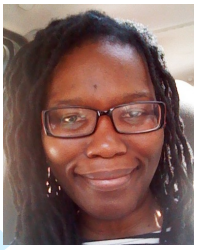
Figure 10: Examples of zig-zag approaches correctly classified by our algorithm, with 88% accuracy. Thus our vehicle can anticipate upcoming traffic light controlled pedestrian crossings.



Figure 11: Examples of boxed junctions correctly classified by our algorithm (86% prediction accuracy). Thus no stopping or traversal areas are automatically detected.

For Review

[3] N. Fairfield and C. Urmson, "Traffic light mapping and detection," in *International Conference on Robotics and Automation*, May 2011, pp. 5421–5426.
[4] B. Mathibela, M. Osborne, I. Posner, and P. Newman, "Can Priors Be Trusted? Learning to Anticipate Roadworks," in *IEEE Conference on Intelligent Transportation Systems*, Anchorage, AK, USA, 2012.
[5] B. Mathibela, I. Posner, and P. Newman, "A roadwork scene signature based on the opponent colour model," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2013, pp. 4394–4400.
[6] G. Emmerson, "Minimising disruption from roadworks," May 2010, Transport for London Surface Transport Panel.
[7] T. Veit, J.-P. Tarel, P. Nicolle, and P. Charbonnier, "Evaluation of road marking feature extraction," in *International Conference on Intelligent Transportation Systems*, 2008, pp. 174–181.
[8] T. Wu and A. Ranganathan, "A practical system for road marking detection and recognition," in *IEEE Intelligent Vehicles Symposium*, 2012, pp. 25–30.
[9] P. Charbonnier, F. Diebolt, Y. Guillard, and F. Peyret, "Road markings recognition using image processing," in *IEEE Conference on Intelligent Transportation Systems*, 1997, pp. 912–917.
[10] P. Foucher, Y. Sebsadji, J.-P. Tarel, P. Charbonnier, and P. Nicolle, "Detection and recognition of urban road markings using images," in *International Conference on Intelligent Transportation Systems*, 2011, pp. 1747–1752.
[11] S. Vacek, C. Schimmel, and R. Dillmann, "Road-marking analysis for autonomous vehicle guidance," in *European Conference on Mobile Robots*, 2007.
[12] N. Dalal and B. Triggs, "Histograms of Oriented Gradients for human detection," in *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, vol. 1, 2005, pp. 886–893.
[13] J. Collado, C. Hilario, A. de la Escalera, and J. Armingol, "Detection and classification of road lanes with a frequency analysis," in *Proceedings of the Intelligent Vehicles Symposium*, 2005, pp. 78–83.
[14] J. Collado, C. Hilario, A. D. L. Escalera, and J. Armingol, "Adaptative road lanes detection and classification," in *Advanced Concepts for Intelligent Vision Systems*. Springer, 2006, pp. 1151–1162.
[15] B. Soheilian, N. Paparoditis, and D. Boldo, "3D road marking reconstruction from street-level calibrated stereo pairs," *ISPRS Journal of Photogrammetry and Remote Sensing*, vol. 65, no. 4, pp. 347–359, 2010.
[16] R. Danescu and S. Nedevschi, "Detection and classification of painted road objects for intersection assistance applications," in *IEEE Conference on Intelligent Transportation Systems*, 2010, pp. 433–438.
[17] L. Yunchong, H. Kezhong, and J. Peifa, "Road markers recognition based on shape information," in *Intelligent Vehicles Symposium*, 2007, pp. 117–122.
[18] A. Kheyrollahi and T. Breckon, "Automatic real-time road marking recognition using a feature driven approach," *Machine Vision and Applications*, vol. 23, no. 1, pp. 123–133, 2012.
[19] M. Noda, T. Takahashi, I. Ide, H. Murase, Y. Kojima, and T. Naito, "Recognition of Road Markings from In-Vehicle Camera Images by a Generative Learning Method," in *Machine Vision Application*, 2009, pp. 514–517.
[20] C. Duchow and B. Kortner, "Aggregating lane markings into lanes for intersection assistance," in *IEEE Intelligent Vehicles Symposium*, 2007, pp. 722–727.
[21] R. Hartley and A. Zisserman, *Multiple View Geometry in Computer Vision*. Cambridge Univ Press, 2000, vol. 2.
[22] H. Ishida, K. Kidono, Y. Kojima, and T. Naito, "Road marking recognition for map generation using sparse tensor voting," in *International Conference on Pattern Recognition*, 2012, pp. 1132–1135.
[23] A. Huang, D. Moore, M. Antone, E. Olson, and S. Teller, "Finding multiple lanes in urban road networks with vision and Lidar," *Autonomous Robots*, vol. 26, no. 2-3, pp. 103–122, 2009.
[24] J. Lafferty, A. McCallum, and F. Pereira, "Conditional Random Fields: Probabilistic models for segmenting and labeling sequence data," in *International Conference on Machine Learning*, 2001.
[25] F. Ramos, D. Fox, and H. Durrant-Whyte, "CRF-Matching: Conditional Random Fields for Feature-Based Scan Matching," in *Robotics Science and Systems*, 2007.
[26] D. Kollar and N. Friedman, *Probabilistic Graphical Models: Principles and Techniques*. The MIT Press, 2009.
[27] D.-T. Lee and B. Schachter, "Two algorithms for constructing a Delaunay triangulation," *International Journal of Computer and Information Sciences*, vol. 9, no. 3, pp. 219–242, 1980.
[28] C. Bishop, *Pattern Recognition and Machine Learning*. Springer, 2006.
[29] K. Murphy, Y. Weiss, and M. Jordan, "Loopy belief propagation for approximate inference: An empirical study," in *Proceedings of the Conference on Uncertainty in Artificial Intelligence*. Morgan Kaufmann Publishers Inc., 1999, pp. 467–475.
[30] G. Hoefel and C. Elkan, "Learning a two-stage SVM/CRF sequence classifier," in *Proceedings of the ACM Conference on Information and Knowledge Management*, 2008, pp. 271–278.
[31] C. Seiffert, T. Khoshgoftaar, J. van Hulse, and A. Napolitano, "RUSBoost: Improving classification performance when training data is skewed," in *International Conference on Pattern Recognition*, 2008, pp. 1–4.
[32] S. Friedman, H. Pasula, and D. Fox, "Voronoi Random Fields: Extracting Topological Structure of Indoor Environments via Place Labeling," in *Proceedings of the International Joint Conference on Artificial Intelligence*, vol. 7, 2007, pp. 2109–2114.
[33] M. Schmidt. (2010) UGM: A Matlab Toolbox for Probabilistic Undirected Graphical Models. [Online]. Available: http://www.di.ens.fr/ mschmidt/Software/UGM.html



Bonolo Mathibela obtained her BSc. in Mechatronics Engineering from the University of Cape Town in 2008. She then undertook a PhD in Robotics at the University of Oxford, where she joined the Mobile Robotics Group (MRG). Her research interests include situational awareness in intelligent vehicles specifically biologically inspired roadwork scene understanding, colour spaces, roadwork mapping, road marking detection and interpretation, and road boundary estimation. She tackles the question of road semantic understanding and interpretation for self-driving cars and intelligent vehicles.



Ingmar Posner obtained an MEng degree in Electronic Systems Engineering from Aston University and a DPhil from Oxford University on the modeling and processing of underwater sonar signals in bioacoustics. He is a University Lecturer in Engineering Science at the University of Oxford and co-lead of the Mobile Robotics Group (MRG). His research tackles questions such as what semantic information can be inferred about the environment the robot has traversed and how this knowledge can feed into the decision-making process of an autonomous agent such as a self-driving car? His current research focus lies on closing the action-perception loop in semantic mapping to enable robust robot decision-making and online planning and exploration in the context of, amongst others, autonomous transport.



Paul Newman obtained an M.Eng. in Engineering Science from Oxford University in 1995. He then undertook a Ph.D. in autonomous navigation at the Australian Center for Field Robotics, University of Sydney, Australia. In 1999 he returned to the United Kingdom to work in the commercial subsea navigation industry. In late 2000 he joined the Dept of Ocean Engineering at M.I.T. where as a post-doc and later a research scientist, he worked on algorithms and software for robust autonomous navigation for both land and sub-sea agents. In early 2003 he returned to Oxford as a Departmental Lecturer in Engineering Science before being appointed to a University Lectureship in Information Engineering and becoming a Fellow of New College in 2005. He heads the Oxford Mobile robotics Research group and has research interests in autonomous navigation particularly Simultaneous Localisation and Mapping. He is on the editorial board of the International Journal of Robotics Research and The Journal of Field Robotics and a IEEE R.A.S European Distinguished Lecturer for 2008 and 2009.

For Peer Review Only

References

[1] A. Irschara, C. Zach, J.-M. Frahm, and H. Bischof. From Structure-from-Motion Point Clouds to Fast Location Recognition. In *CVPR*, 2009.

[2] Y. Li, N. Snavely, and D. P. Huttenlocher. Location Recognition using Prioritized Feature Matching. In *ECCV*, 2010.

[3] Y. Li, N. Snavely, D. P. Huttenlocher, and P. Fua. Worldwide Pose Estimation Using 3D Point Clouds. In *ECCV*, 2012.

[4] T. Sattler, B. Leibe, and L. Kobbelt. Fast Image-Based Localization using Direct 2D-to-3D Matching. In *ICCV*, 2011.

[5] T. Sattler, B. Leibe, and L. Kobbelt. Improving Image-Based Localization by Active Correspondence Search. In *ECCV*, 2012.

[6] C. Häne, C. Zach, A. Cohen, R. Angst, and M. Pollefeys. Joint 3D Scene Reconstruction and Class Segmentation. In *CVPR*, 2013.

[7] N. Savinov C. Häne and M. Pollefeys. Class Specific 3D Object Shape Priors Using Surface Normals. In *CVPR*, 2014.

[8] M. Cummins and P. Newman. Appearance-only slam at large scale with fab-map 2.0. *IJRR*, 30(9):1100–1123, August 2011.

[9] D. Galvez-Lopez and J. D. Tardos. Real-time loop detection with bags of binary words. In *IROS*, pages 51 –58, sept. 2011.

[10] N. Sünderhauf and P. Protzel. BRIEF-Gist - Closing the loop by simple means. In *IROS*, pages 1234–1241, 2011.

[11] D. Galvez-Lopez and J. D. Tardos. Bags of binary words for fast place recognition in image sequences. *IEEE Transactions on Robotics*, 28(5):1188–1197, October 2012.

[12] Gim Hee Lee, Friedrich Fraundorfer, and Marc Pollefeys. Structureless pose-graph loop-closure with a multi-camera system on a self-driving car. In *IROS*, 2013.

[13] David Nistér and Henrik Stewénius. Scalable recognition with a vocabulary tree. In *CVPR*, 2006.

[14] D. Nister. An efficient solution to the five-point relative pose problem. *PAMI*, 26(6):756–770, 2004.

[15] R. I. Hartley and A. Zisserman. *Multiple View Geometry in Computer Vision*. Cambridge Univ. Press, 2nd edition, 2004.

[16] Shahriar Negahdaripour Berthold K. P. Horn, Hugh M. Hilden. Closed-form solution of absolute orientation using unit quaternions. *Journal of the Optical Society of America A*, 1987.

[17] Gim Hee Lee, Bo Li, Marc Pollefeys, and Friedrich Fraundorfer. Minimal solutions for pose estimation of a multi-camera system. In *ISRR*, 2013.

[18] G. H. Lee and M. Pollefeys. Unsupervised Learning of Threshold for Geometric Verification in Visual-Based Loop-Closure. In *ICRA*, 2014.

[19] Christopher Urmson, Joshua Anhalt, Hong Bae, J. Andrew (Drew) Bagnell, Christopher R. Baker, Robert E Bittner, Thomas Brown, M. N. Clark, Michael Darms, Daniel Demitrish, John M Dolan, David Duggins, David Ferguson , Tugrul Galatali, Christopher M Geyer, Michele Gittleman, Sam Harbaugh, Martial Hebert, Thomas Howard, Sascha Kolski, Maxim Likhachev, Bakhtiar Litkouhi, Alonzo Kelly, Matthew McNaughton, Nick Miller, Jim Nickolaou, Kevin Peterson, Brian Pilnick, Ragunathan Rajkumar, Paul Rybski, Varsha Sadekar, Bryan Salesky, Young-Woo Seo, Sanjiv Singh, Jarrod M Snider, Joshua C Struble, Anthony (Tony) Stentz, Michael Taylor , William (Red) L. Whittaker, Ziv Wolkowicki, Wende Zhang, and Jason Ziglar. Autonomous driving in urban environments: Boss and the urban challenge. *Journal of Field Robotics Special Issue on the 2007 DARPA Urban Challenge, Part I*, 25(8):425–466, June 2008.

[20] Ashley Napier, Gabe Sibley, and Paul Newman. Real-time bounded-error pose estimation for road vehicles using vision. In *13th Intl. IEEE Conference on Intelligent Transportation Systems (ITSC)*, pages pp. 1141–1146, Funchal, Portugal, September 2010. 09.

[21] M.A. Fischler and R.C. Bolles. Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. *Comm. ACM*, 24(6):381–395, 1981.

[22] D. Lowe. Distinctive image features from scale-invariant keypoints. *IJCV*, 60(2):91–110, 2004.

[23] L. Ladicky, C. Russel, P. Kohli, and P. Torr. Associative Hierarchical Random Fields. *PAMI*, 99, 2013.

[24] J. Malik, S. Belongie, T. Leung, and J. Shi. Contour and texture analysis for image segmentation. *IJCV*, 43(1):7–27, 2001.

[25] T. Ojala, M. Pietikainen, and D. Harwood. Performance evaluation of texture measures with classification based on Kullback discrimination of distributions. In *ICPR*, 1994.

[26] T. Sattler, T. Weyand, B. Leibe, and L. Kobbelt. Image Retrieval for Image-Based Localization Revisited. In *BMVC*, 2012.

[27] S. M. Bileschi. *StreetScenes: Towards Scene Understanding in Still Images*. PhD thesis, MIT, 2006.

[28] N. Snavely, S.M. Seitz, and R. Szeliski. Photo Tourism: Exploring Photo Collections in 3D. In *SIGGRAPH*, 2006.

[29] C. Wu, S. Agarwal, B. Curless, and S.M. Seitz. Multicore bundle adjustment. In *CVPR*, 2011.

[30] O. Chum and J. Matas. Optimal Randomized RANSAC. *PAMI*, 30(8):1472–1482, 2008.

[31] O. Chum, J. Matas, and J. Kittler. Locally Optimized RANSAC. In *DAGM*, 2003.

[32] C. Wu. SiftGPU: A GPU Implementation of Scale Invariant Feature Transform (SIFT). http://cs.unc.edu/~ccwu/siftgpu, 2007.