The International Journal of Robotics Research

http://ijr.sagepub.com/

Classification and Semantic Mapping of Urban Environments

B. Douillard, D. Fox, F. Ramos and H. Durrant-Whyte

The International Journal of Robotics Research 2011 30: 5 originally published online 29 July 2010
DOI: 10.1177/0278364910373409

The online version of this article can be found at:
http://ijr.sagepub.com/content/30/1/5

Published by:
SAGE
http://www.sagepublications.com

On behalf of:
ijrr
Multimedia Archives

The International Journal of
Robotics Research
30(1) 5–32

DOI: 10.1177/0278364910373409
ijr.sagepub.com

Classification and Semantic Mapping of Urban Environments

B Douillard[2], D Fox[1], F Ramos[2] and H Durrant-Whyte[2]

Abstract

In this paper we address the problem of classifying objects in urban environments based on laser and vision data. We propose a framework based on Conditional Random Fields (CRFs), a flexible modeling tool allowing spatial and temporal correlations between laser returns to be represented. Visual features extracted from color imagery as well as shape features extracted from 2D laser scans are integrated in the estimation process. The paper contains the following novel developments: (1) a probabilistic formulation for the problem of exploiting spatial and temporal dependencies to improve classification; (2) three methods for classification in 2D semantic maps; (3) a novel semi-supervised learning algorithm to train CRFs from partially labeled data; (4) the combination of local classifiers with CRFs to perform feature selection on high-dimensional feature vectors. The system is extensively evaluated on two different datasets acquired in two different cities with different sensors. An accuracy of 91% is achieved on a seven-class problem. The classifier is also applied to the generation of a 3 km long semantic map.

Keywords

Multi-sensor function, semantic mapping, urban ground vehicles

1. Introduction

Classification and semantic mapping are essential steps toward the long-term goal of equipping a robot with the ability to understand its environment. Classifiers generate semantic information which can enable robots to perform high-level reasoning about their environments. For instance, in search and rescue tasks, a mobile robot that can reason about objects such as doors, and places such as rooms, is able to coordinate with first responders in a much more natural way. It can accept commands such as "Search the room behind the third door on the right of this hallway", and send information such as "There is a wounded person behind the desk in that room" (Kumar et al. 2004). As another example, consider autonomous vehicles navigating in urban areas. While the recent success of the DARPA Urban Challenge (see http://www.darpa.mil/grandchallenge/index.asp) demonstrates that it is possible to develop autonomous vehicles that can navigate safely in constrained settings, successful operation in more realistic, populated urban areas requires the ability to distinguish between objects such as cars, people, buildings, trees, and traffic lights.

In this paper a classification framework based on Conditional Random Fields (CRFs) is proposed. CRFs are discriminative models for classification of structured (dependent) data (Lafferty et al. 2001). CRFs provide a flexible framework in which different types of spatial and temporal dependencies can be represented.

1.1. Overview

The sequence of operations involved in the proposed classification systems is described in Figure 1. At the input of the processing pipeline is the raw data: in the experiments described in this paper, it is for instance acquired by a modified car equipped with vision and 2D ranging sensors.

The first preprocessing phase contains two operations: (1) projection of the laser returns onto the image and (2) definition of Regions of Interest (ROIs) in the image based on the projected returns. A ROI is defined around each projected point. Feature extraction is then performed in each of the ROIs. As we describe in the following, the feature extraction stage is the key to achieving good classification results. A few vision features are illustrated in

[1] University of Washington, Seattle, Washington, USA
[2] Australian Centre for Field Robotics, University of Sydney, Sydney, Australia

Corresponding author:
B. Douillard, Australian Centre for Field Robotics, University of Sydney, Sydney, NSW 2006 Australia
Email: b.douillard@cas.edu.au

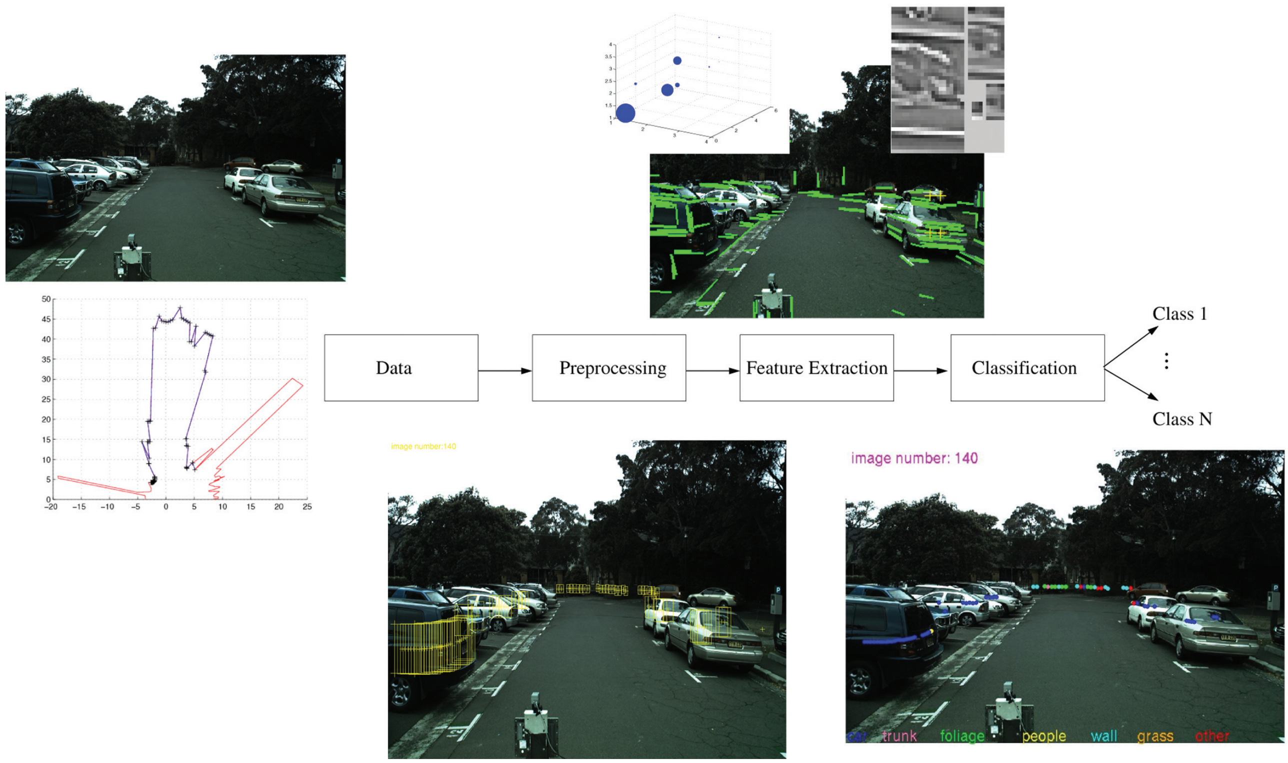
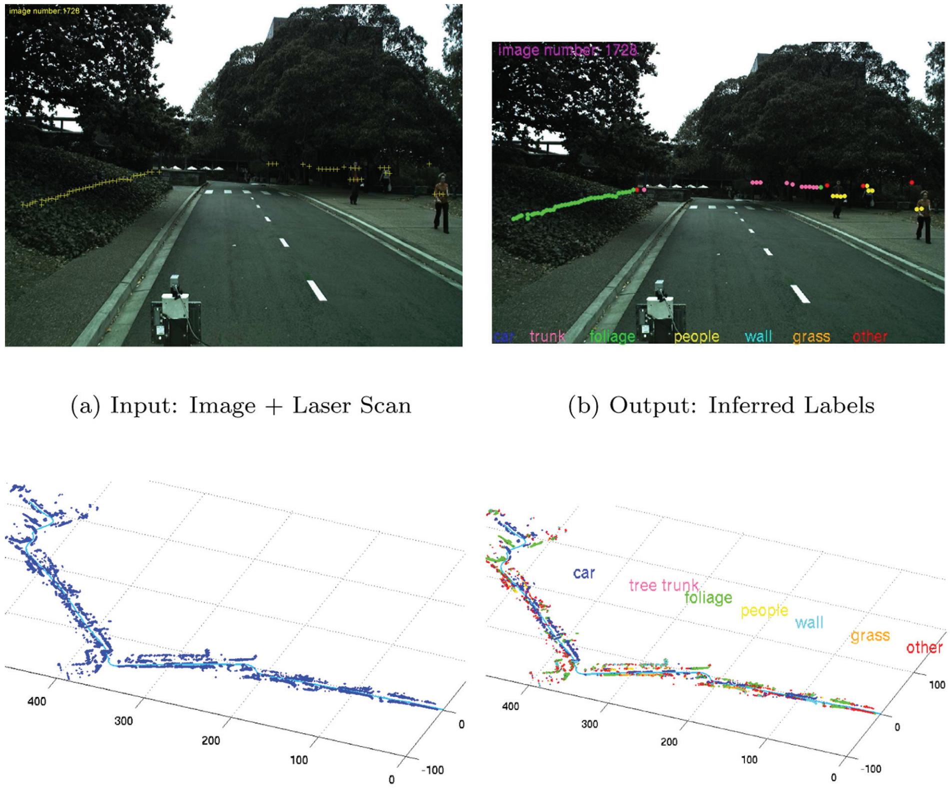


Fig. 1. The classification workflow (following the layout proposed by Duda et al. (2001)). On the left, an example of input data is shown: a color image and a 2D laser scan. The red part of the scan does not fall within the field of view of the camera and is disregarded during the rest of the processing. Below the “Preprocessing” block is an example of Region of Interest (ROI) generation. The ROI defined around the projection of each laser return in the image is indicated by a yellow box. Above the "Feature Extraction” box, a few examples of vision features computed in the ROI in the far right of the scene. The green lines are extracted based on an edge detector (see http://www.cs.uiuc.edu/homes/dhoiem/). Features such as the maximum length of the lines in a ROI or the count of vertical lines are computed (as detailed in Section 6.2). The 3D plot represents the RGB space, and the size of the blue dots is mapped to the number of counts in the bins of a RGB histogram. The third inset represents texture coefficients obtained with the Steerable Pyramid descriptor (Simoncelli and Freeman 1995). The full set of features also includes laser features which are not illustrated here but developed in Section 6.1. The image on the right shows the inferred labels. The estimate associated to each return is indicated by the color of the return. The legend is provided at the bottom of the image.

Figure 1. The final stage of the processing pipeline contains the actual classifier. In the case of Figure 1, the classifier estimates the label of the features representing each laser return. Possible labels include “car”, “people” and “foliage”.

In this paper, the flexibility of CRF-based classification is presented using various models of increasing complexity integrating 2D laser scans and imaging data. We start with a simple chain CRF formed by linking consecutive laser beams in the scans. This configuration models the geometrical structure of a scan and captures the typical shapes of objects. Temporal information is then incorporated by adding links between consecutive laser scans based on correspondences obtained by a scan matching algorithm. This leads to a network in which estimation is equivalent to a filtering algorithm, thus taking temporal as well as spatial dependencies into account. This network, and its associated estimation machinery, allows for temporal smoothing as the network grows with the registration of incoming scans. Finally, it is shown that a CRF can be used to capture the various structures characterizing a geometric map. This involves defining a network on a set of already aligned laser scans and running estimation as a batch process. In the map-sized network obtained in this way, classification is performed jointly across the whole laser map and can, in turn, exploit larger geometric structures to improve local classification. Some of the inputs and outputs of the model are illustrated in Figure 2.

By building on the recently developed Virtual Evidence Boosting (VEB) procedure (Liao et al. 2007), a novel Maximum Pseudo-Likelihood (MPL) learning approach is proposed, that is able to automatically select features during the learning phase. Expert knowledge about the problem is encoded as a selection of features capturing particular properties of the data such as geometry, color and texture. An extension of MPL learning to the case of partially labeled data is proposed thus significantly reducing the burden of manual data annotation.



(a) Input: Image + Laser Scan

(b) Output: Inferred Labels

(c) Input: Laser Map (+ Images)

(d) Output: Semantic Map

Fig. 2. Examples of inputs and outputs of the 2D classification system. (a) One possible input of the system: a laser scan and the corresponding image. In this figure the laser returns are projected onto the image and represented by yellow crosses. The laser scanner used in the corresponding experiments can be seen at the bottom of the image. (b) The output obtained from (a). For each laser return, the system estimates a class label which is here indicated by the color of the return. (c) A second possible input of the system: a set of aligned 2D laser scans. The platform's trajectory is displayed in magenta. The system also requires the images acquired along the trajectory to perform classification. (d) The output obtained from (c). The system estimates a class label for each laser return is the 2D map. The units of the axes are meters.

Based on two datasets acquired with different platforms in two different cities, eight different sets of results are presented. This allows for an investigation of the performance of the models applied to large-scale feature networks. The generation of semantic maps based on this framework is demonstrated. One of these networks involves the generation of a 3 km long semantic map and achieves an accuracy of 91% on a seven-class problem. While the test networks contain on average 7,500 nodes, the associated inference time is less than 11 seconds on a standard PC.

The paper is concluded with a discussion on the limitations of the proposed networks in terms of their smoothing effect. The fundamental importance of features extracted from data in the generation of accurate classifiers is also highlighted.

This paper makes a number of contributions:

- Spatial and temporal correlations between laser returns are represented using a single framework based on CRFs.
- Filtering and smoothing are shown to be particular instances of the inference process in this general representation.
- The model is shown to also support the generation of large-scale (a few kilometers long) 2D semantic maps, while also being demonstrated on 3D data.
- An extension of MPL learning is proposed to train the models from partially labeled data. It is based on a formulation of CRFs that combine local classifiers rather than reasoning directly on high-dimensional features as implemented in standard log-linear formulations.
- The model can deal with different sensors as it is able to incorporate multi-modal data by means of a feature selection process in high-dimensional feature vectors (using Logitboost).
- The model instantiated in its filtering, smoothing and mapping version is demonstrated on real-world datasets acquired in two different cities by two different vehicles. A total of eight experiments are reported.

1.2. Paper Structure

This paper is organized as follows. Section 2 discusses related work. Section 3 introduces CRFs as well as a novel extension of MPL learning for training from partially labeled data. Section 4 presents the core of the model. In particular, the instantiation of the model from data and its ability to represent spatial and temporal correlations are developed. Section 5 shows how the model can be deployed for the generation of semantic maps. Section 6 presents the various features used in our implementation. Section 7 proposes an experimental evaluation of the model when used as a filter or a smoother. Section 8 proposes a second experimental evaluation in which the model is used to generate semantic maps. Section 9 discusses the limitations of the proposed approach by analyzing the nature of network links. Section 10 concludes. Note that most of the figures need to be seen in color, as in the online version of this article.

2. Related Work

Most current approaches to mapping focus on building geometric representations of an environment. The Simultaneous Localization and Mapping (SLAM) framework, in particular, has addressed the problem of building accurate geometric representation of an environment based on laser data, on vision data, or combined laser and vision (Durrant-Whyte and Bailey 2006). Landmark models combining visual and geometric feature have been designed in conjunction with Bayesian filters so that the landmark representation can be updated over time (Douillard et al. 2007b; Kumar et al. 2007; Kaupp et al. 2007). All of these techniques reconstruct the geometry and the visual appearance of the environment but do not readily allow the identification of objects in a scene. As formulated by Pantofaru et al. (2003), the next natural step is to extract a *symbolic* representation of the environment in which objects and structures of interest are labeled.

Semantic representations can be extremely valuable since they enable robots to perform high-level reasoning about environments and objects therein. Martinez-Mozos et al. (2007) propose a method for classifying the pose of a robot into semantic classes corresponding to places. Adaboost (Schapire and Singer 1999) is used for training on features extracted from 2D laser scans and vision data. There are four main differences between the work described in this paper and the method of Martinez-Mozos et al. (2007). First, our method is developed for outdoor rather than indoor applications. Second, the proposed method performs object recognition rather than place recognition. The difference of scale between the two problems is crucial. The number of features which can be gathered in a given environment is often much larger than the number of features which can be extracted from one object in the same environment. As a consequence, the extraction of discriminative patterns is facilitated in a place recognition problem. Third, in order to perform multi-class classification, Martinez-Mozos et al. (2007) combines binary classifiers in a heuristic manner (this is further developed in Martinez-Mozos et al. (2005)) while the approach proposed here extends to an arbitrary number of classes without any modification. Finally, the model developed in this paper outputs a dense semantic map of the whole environment (as illustrated in Figure 12) while the system in Martinez-Mozos et al. (2007) provides labels tied to the trajectory of the robot.

In the context of outdoor environments, various approaches have been proposed to exploitspatio-temporal information in performing classification of dynamic objects. The work of Luber et al. (2008) extends the tracking algorithms developed by Schulz (2006) and Toyama (2001) to represent the changing appearance of objects imaged with a 2D laser scanner. This model integrates both classification and tracking, and is able to represent five classes including pedestrians, skaters and bicycles. The specificity of the method lies in the use of an unsupervised learning algorithm. As suggested, unsupervised training is possible only if a few strong features allow the observations to be separated into distinct clusters corresponding to classes. In the application described, such features include the velocity of the track for example. Once the unsupervised clustering has been applied, the mapping from clusters to class labels requires the intervention of an operator to specify which components in the cluster model correspond to which classes. To avoid this external intervention our approach uses supervised and semi-supervised learning algorithms.

Other approaches to dynamic object detection based on 2D laser data and monocular imagery were developed by Katz et al. (2008a,b) and Monteiro et al. (2006). These emphasize the role of spatial and temporal integration to achieving robust recognition as exploited in this work. Other work, directly estimating the class of an object without consideration of temporal correlations was developed by Posner et al. (2007, 2009). The authors combine 3D laser range data with camera information to classify surface types such as brick, concrete, grass or pavement in outdoor environments. Each laser scan is considered independently for classification. Other work shows that performance can be improved by jointly classifying laser returns using techniques such as associative Markov networks (Triebel et al. 2006), Relational Markov Networks (Limketkai et al. 2005) and other "object-oriented" types of models (Anguelov et al. 2004).

In this paper CRFs (Douillard et al. 2007a, 2008) are employed as a method for structured modeling of both temporal and spatial relations between sensor data, features and object models. Structured classification is also demonstrated in Posner et al. (2008) where objects are classified based on monocular imagery and laser data. This approach does not incorporate temporal information and while it is designed to handle multi-modal data, user-specified inputs are required for each modality. A structured model is used by Anguelov et al. (2005) where a Markov Random Field model is used to segment objects from 3D laser scans. The model employs simple geometric features to classify four classes: ground, building, tree and shrubbery. Friedman et al. (2007) introduced Voronoi Random Fields, which generate semantic place maps of indoor environments by labeling the points on a Voronoi graph of a laser map using CRFs.

Object recognition has been a major research topic in the computer vision community (Murphy et al. 2003; Torralba et al. 2004; Viola and Jones 2004; Fei-Fei and Perona 2005; Hoiem et al. 2006; Vedaldi et al. 2007; Felzenszwalb et al. 2008; Sudderth et al. 2008). However, direct application of the algorithms to robotics problems is not always feasible or appropriate. In particular, the sequential, real-time, multi-perspective views encountered in robotic navigation is conceptually different from most vision-based object recognition tasks. Indeed, robots can often exploit temporal and spatial correlation between views to aid object classification. It is also most common that robots do not need to use vision alone and can benefit from other ranging or location information to aid the classification task.

3. Structured Classification

This paper builds on previous work by addressing the classification problems of Douillard et al. (2007a, 2008) and by combining multi-modal data fusion, structured reasoning and temporal estimation into a single integrated model.

This section introduces techniques to jointly classify structured (dependent) data. They are divided into two general classes: generative and discriminative. Section 3.1 explains why a discriminative approach is chosen, and in particular it introduces the CRF framework. Inference mechanisms and learning algorithms are then discussed. The standard MPL approach to CRF training is explained. A modified version of MPL learning for training from partially labeled data is introduced. It is shown that the use of boosting classifiers within the models reduces the complexity of the learning problem.

3.1. Generative or Discriminative?

Generally, probabilistic models fall into two categories: generative and discriminative (Vapnik 2000; Ng and Jordan 2002). A generative model is a joint probability of all variables, whereas a discriminative model provides a model only of the target variables conditioned on the observed variables. A generative model can be used, for example, to simulate (that is, generate) values of any variable in the model, whereas a discriminative model allows only sampling of the target variables conditioned on the observed quantities (Bishop 2006). In the context of classification, discriminative models directly represent the conditional distribution of the hidden labels given all of the observations, $p(\mathbf{x}|\mathbf{z})$. In contrast, generative models represent the joint distributions $p(\mathbf{x}, \mathbf{z})$ and the Bayes rule is used to extract an estimate $p(\mathbf{x}|\mathbf{z})$ over class labels.

There are a number of advantages to using discriminative classifiers. As developed by Ng and Jordan (2002), one advantage was articulated by Vapnik (2000) and relies on the intuitive wisdom that "one should solve the (classification) problem directly and never solve a more general problem as an intermediate step (such as modeling $p(\mathbf{z}|\mathbf{x})$)". The term $p(\mathbf{z}|\mathbf{x})$ is called the sensor model in the context of robotics studies. This term is required to extract an estimate over classes $p(\mathbf{x}|\mathbf{z})$ from the joint distribution $p(\mathbf{x}, \mathbf{z})$. In contrast, a sensor model is not needed when the conditional distribution $p(\mathbf{x}|\mathbf{z})$ is represented directly.

In addition, modeling the term $p(\mathbf{z}|\mathbf{x})$ is likely to be computationally hard. It can be intuitively appreciated that devising the model of a sensor (such as a camera or a laser) is indeed a difficult task. This aspect often causes the designer to assume the independence of the observations given the states. Owing to these assumptions, the resulting generative representation cannot exploit inherent correlations in adjacent observations. Better performances of discriminative models were, in fact, observed in several studies. Ng and Jordan (2002) compared a logistic regression classifier (a discriminative model) with a naive Bayes classifier (a generative model). CRFs (a discriminative model) have also been shown to provide better performance than Markov Random Fields (a generative model) in various studies. These include man-made structure detection systems based on vision data (Kumar 2005) with networks instantiated as 2D lattices. Similar observations were made in part-of-speech tagging experiments with chain networks (Lafferty et al. 2001).

Since the problem considered here is the classification of laser returns into semantic classes, that is, building the mapping $p(\mathbf{x}|\mathbf{z})$, without the need for an explicit sensor model ($p(\mathbf{z}|\mathbf{x})$) or an explicit model of the data ($p(\mathbf{z})$), we choose a discriminative representation as the base model.

3.2. Conditional Random Fields

CRFs are undirected graphical models developed for labeling sequence data (Lafferty et al. 2001). CRFs directly model $p(\mathbf{x}|\mathbf{z})$, the *conditional* distribution over the hidden variables $\mathbf{x}$ given observations $\mathbf{z}$. Here the set $\mathbf{x}$ represents the class labels to be estimated and the set $\mathbf{z}$ contains the raw data. The CRF fundamental is briefly discussed below, further details can be found in Sutton and McCallum (2006).

In this work we consider a particular type of CRF which are often referred to as pairwise CRFs. They contain only two types of potential functions: local potentials ϕ_A and pairwise potentials ϕ_I. In addition, we assume the hidden states to be discrete since we consider classification networks only. The conditional distribution over all of the labels $\mathbf{x}$ given the observations $\mathbf{z}$ becomes

$$p(\mathbf{x}|\mathbf{z}) = \frac{1}{Z(\mathbf{z})} \prod_i \phi_A(x_i, \mathbf{z}) \prod_e \phi_I(x_{e_1}, x_{e_2}, \mathbf{z}), \quad (1)$$

where

$$\phi_A(x_i, \mathbf{z}) = \exp\left(\boldsymbol{\lambda}_A \mathbf{A}\left(x_i, \mathbf{f}_A(\mathbf{z}, x_i)\right)\right), \quad (2)$$

$$\phi_I(x_{e_1}, x_{e_2}, \mathbf{z}) = \exp\left(\boldsymbol{\lambda}_I \mathbf{I}\left(x_{e_1}, x_{e_2}, \mathbf{f}_I(\mathbf{z}, \{x_{e_1}, x_{e_2}\})\right)\right). \quad (3)$$

The term $Z(\mathbf{z})$ refers to the partition function, i ranges over the set of nodes and e over the set of edges. The functions $\mathbf{f}_A$ and $\mathbf{f}_I$ extract the features required by functions $\mathbf{A}$ and $\mathbf{I}$, respectively. Functions $\mathbf{f}_A$ and $\mathbf{f}_I$ correspond to the feature extraction step appearing in Figure 1. The functions $\mathbf{A}$ and $\mathbf{I}$ are the *association* and *interaction* potentials, respectively. An association potential $\mathbf{A}$ can be a classifier which estimates the class label of node x_i but does not take into account information contained in the structure of the neighborhood. An interaction potential $\mathbf{I}$ is a function associated with each edge e of the CRF graph, where x_{e1} and x_{e2} are the nodes connected by edge e. Intuitively, interaction potentials measure the compatibility between neighboring nodes and act as smoothers by correlating the estimation across the network. The terms $\boldsymbol{\lambda}_A$ and $\boldsymbol{\lambda}_I$ are sets of weights multiplying the output of the functions $\mathbf{A}$ and $\mathbf{I}$, respectively. These weights are estimated during the training phase.

To differentiate between the terms $\mathbf{A}$ and ϕ_A, the latter will be called the *local potential*. Depending on the context, ϕ_A will either return a scalar or a vector; this will be indicated in the text. In the equations above, it returns a scalar. Also, to differentiate between the terms $\mathbf{I}$ and ϕ_I, the latter will be called the *pairwise potential*. The term ϕ_I can either be a scalar or a matrix, which will be clear from the context. In the equations above, it is a scalar. When ϕ_I is a matrix, its size is $[L \times L]$, where L is the number of classes, and it is referred to as the *pairwise matrix*. To simplify the notation, the dependency of the terms ϕ_A and ϕ_I on $\mathbf{z}$ will not be made explicit in the remainder of this document.

The set of Equations (1), (2) and (3) will be referred to in the text via the more compact formulation below, where all of the terms have been gathered in the exponential:

$$p(\mathbf{x}|\mathbf{z}) = \frac{1}{Z} \exp\Bigg(\boldsymbol{\lambda}_A \sum_i \mathbf{A}\left(x_i, \mathbf{f}_A(\mathbf{z}, x_i)\right) + \boldsymbol{\lambda}_I \sum_e \mathbf{I}\left(x_{e1}, x_{e2}, \mathbf{f}_I(\mathbf{z}, \mathbf{x}_e)\right)\Bigg). \quad (4)$$

In this paper, we assume that the random field defined by Equation (4) is homogeneous: the functions $\mathbf{A}$ and $\mathbf{I}$ are independent of the nodes at which they are instantiated. In addition, we assume that the field is isotropic, that is, the interaction potential $\mathbf{I}$ is non-directional: $I(x_{e_1}, x_{e_2}, \mathbf{f}_I) = I(x_{e_2}, x_{e_1}, \mathbf{f}_I)$.

It is important to note that CRFs are globally conditioned on the whole set of observations $\mathbf{z}$. This allows the integration of more complex observations into the model. For instance, the ROI displayed in Figure 1 overlap and will generate observations with an overlapping content. Such cases can seamlessly be integrated in a CRF while they would be more problematic in generative models.

3.3. CRF Inference

A widely used inference framework is Belief Propagation (BP) (Pearl 1988; Jordan and Weiss 2002). BP generates exact results in a graph such as trees or polytrees. However, in cyclic graphs the algorithm is only approximate and is not guaranteed to converge (Murphy et al. 1999). In the case of cyclic graphs, the algorithm is called loopy BP. A number of theoretical studies have formulated theoretical conditions with respect to the convergence of loopy BP, for instance, in terms of the Bethe free energy (Watanabe and Fukumizu 2009). In practice, loopy BP often converges to good approximates and has been successfully applied to several problems (Frey and MacKay 1997; Freeman et al. 2000). In our experiments, convergence was verified experimentally; the corresponding analysis is reported in Section 8.5.

Inference algorithms are based on the concept of marginalization. In the case of pairwise networks, the marginalization process as implemented by BP has an intuitive formulation. The sequence of computations specified by the algorithm can be interpreted as a flow of messages across the network. If the states of the nodes are discrete (such as in classification problems), a message $m_{ji}(x_i)$ from node j to node i has the following form (Jordan and Weiss 2002):

$$m_{ji}(x_i) = \sum_{x_j}\left(\phi_A(x_j)\,\phi_I(x_i, x_j) \prod_{k \in \mathcal{N}(j)\setminus i} m_{kj}(x_j)\right), \quad (5)$$

where the functions ϕ_A and ϕ_I are as defined in the previous section.

Once the messages have been propagated, the distribution over the states of a node can be recovered as follows:

$$p(x_i|\mathbf{z}) \propto \phi_A(x_i) \prod_{k \in \mathcal{N}(i)} m_{ki}(x_i). \quad (6)$$

Other types of query such as the Maximum *A Posteriori* (MAP) inference involve similar mechanisms (Jordan and Weiss 2002).

There are several other techniques for performing inference in the literature (Sudderth et al. 2003; Szummer et al. 2008). BP was chosen because it allows an intuitive interpretation of inference in terms of network messages which enables the generalization of the concepts of smoothing and filtering under the more general notion of inference in a probabilistic graph (Section 4). Other potentially faster inference techniques such as graph cuts or tree re-weighted message passing for instance (Greig et al. 1989; Wainwright et al. 2005) were not considered here.

3.4. CRF Training

To introduce general learning concepts as applied to the CRF framework we use the following formulation of a CRF:

$$p(\mathbf{x}|\mathbf{z}) = \frac{1}{Z(\mathbf{z})} \prod_{c \in \mathcal{C}} \exp\left(\mathbf{w} \cdot \mathbf{f}_c(\mathbf{x}_c, \mathbf{z})\right), \qquad (7)$$

where $\mathcal{C}$ is a set of node cliques (the dependency of $\mathbf{w}$ on $\mathbf{z}$ is not made explicit in this formulation to simplify the notation). This expression corresponds to a standard log-linear CRF and is different from that given in Equation (4): the potential functions $\phi_c(\mathbf{x}_c, \mathbf{z})$ are now defined as log-linear combinations of the feature functions $\mathbf{f}_c$, i.e. $\phi_c(\mathbf{x}_c, \mathbf{z}) = \exp(\mathbf{w} \cdot \mathbf{f}_c(\mathbf{x}_c, \mathbf{z}))$. This standard formulation is introduced to point out the benefits of the proposed model. The details of the relationship between this formulation and the formulation in Equation (4) are given in the next section.

Learning a CRF consists of defining the set of weights $\mathbf{w}$ in Equation (7), based on a labeled training set. Learning can also be performed on a set of partially labeled data; this will be further discussed in Section 3.6. Maximum Likelihood (ML) estimation provides a general framework for automatically adjusting the free parameters of a model to fit empirical data. Applied to a CRF, it requires maximizing the conditional likelihood $p(\mathbf{x}|\mathbf{z})$ of a labeled set $\{\mathbf{x}, \mathbf{z}\}$ with respect to the parameters $\mathbf{w}$. For the remainder of this section, the dependency of the model on its parameters will be made explicit by writing the conditional likelihood $p(\mathbf{x}|\mathbf{z}, \mathbf{w})$.

The model is more conveniently expressed as a log-linear combination of features. The learning problem consists then of minimizing the negative log-likelihood:

$$L(\mathbf{w}) \triangleq -\log p(\mathbf{x}|\mathbf{z}, \mathbf{w}) \qquad (8)$$

$$= -\sum_{c \in \mathcal{C}} \mathbf{w} \cdot \mathbf{f}_c(\mathbf{x}_c, \mathbf{z}) + \log Z(\mathbf{z}, \mathbf{w}). \qquad (9)$$

Such an optimization problem is NP-hard due to the term $Z(\mathbf{z}, \mathbf{w})$ (Kumar 2005) which involves summing over an exponential number of states configurations; we recall here the formulation of the partition function: $Z(\mathbf{z}, \mathbf{w}) = \sum_{\mathbf{x}} \prod_{c \in \mathbf{C}} \exp(\mathbf{w} \cdot \mathbf{f}_c(\mathbf{x}_c, \mathbf{z}))$.

To circumvent this difficulty, various techniques have been proposed to compute approximations of the partition function $Z(\mathbf{z}, \mathbf{w})$. Some approaches approximate $\log Z(\mathbf{z}, \mathbf{w})$ directly, such as by Markov chain Monte Carlo (MCMC) (Liao 2006) or variational methods (Yedidia et al. 2002). Other approaches estimate the parameters locally, that is, they replace the global normalization constant $Z(\mathbf{z}, \mathbf{w})$ by a set of local normalizations (Sutton and McCallum 2007). We now focus on one such local approach: pseudo-likelihood.

MPL (Besag 1975) is a classical training method which performs ML estimation on sub-graphs of the networks. This is illustrated in Figure 3. Formally, the pseudo-likelihood is expressed as (Liao 2006)

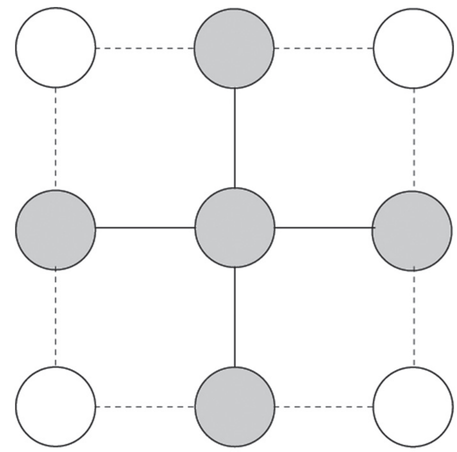


Fig. 3. Illustration of the assumption made during MPL learning. The Markov blanket of the middle node consists of the four nodes which are directly connected to it. During learning, only the neighbors in the Markov blanket are considered and the rest of the network is disregarded when processing this particular node (and a node contributes to several Markov blankets). This assumption makes the computation of the partition function Z tractable. MPL learning also assumes that the nodes in the Markov blanket are observed, that is, it requires their label to be known. Labeled nodes are indicated by their gray color.

$$PL(\mathbf{x}|\mathbf{z}, \mathbf{w}) \triangleq \prod_i p(x_i | MB(x_i), \mathbf{w}) \qquad (10)$$

$$= \prod_i \frac{1}{Z(MB(x_i), \mathbf{w})} \exp\left(\mathbf{w} \cdot \mathbf{f}_i(\{x_i, MB(x_i)\}, \mathbf{z})\right), \qquad (11)$$

where $MB(x_i)$ is the Markov blanket of node x_i: the set of nodes which are directly connected to node x_i. As this formulation suggests, the pseudo-likelihood is the product of all of the local likelihoods, $p(x_i|MB(x_i), \mathbf{w})$. The term $Z(MB(x_i), \mathbf{w})$ is equal to $\sum_{x_i'} \exp\left(\mathbf{w} \cdot \mathbf{f}_i\left(\{x_i', MB(x_i)\}, \mathbf{z}\right)\right)$ and represents the local partition function. It can be easily computed since it involves only the set of immediate neighbors of x_i rather than the whole set of nodes in the graph as was the case for the global partition function $Z(\mathbf{z}, \mathbf{w})$. As a result, evaluating the pseudo-likelihood is much faster than computing the full likelihood $p(\mathbf{x}|\mathbf{z}, \mathbf{w})$ since it only requires evaluating local normalizing functions and avoids the computation of the global partition function $Z(\mathbf{z}, \mathbf{w})$. The difference in complexity between the computation of the likelihood and the pseudo-likelihood is exponential in the number of nodes in the network.

Inference in models trained with MPL can be performed with various techniques including BP (described in the previous section) or MCMC techniques (Liao 2006).

3.5. Logitboost-based Training

While the MPL approach renders the learning problem computationally feasible, it has two drawbacks in addition to being an approximation. We review each of them and explain the solution adopted in this work.

The first limitation of MPL learning is the need for entire labeling of the network: when processing a given node, the MPL procedure requires the labels of the neighbor nodes to be known. Based on these labels, the parameters describing neighborhood interactions can be learnt. As observed in several studies, assuming the states of the neighbor nodes to be known during training might result in over-estimating the weights in the pairwise connections (Geyer and Thompson 1992; Kumar 2005). In Section 3.6 we present a novel extension of the MPL procedure which does not require the neighborhood labels to be known during training. In addition, we show that the latter version of MPL learning can handle partially labeled data.

A second limitation of the standard MPL approach is linked to the potentially large number of weights $\mathbf{w}$. As can be seen in Equation (7), in standard log-linear models, $\mathbf{w}$ directly multiplies the feature vector $\mathbf{f}$, which implies that the model requires one weight per dimension of the feature vector $\mathbf{f}$. In our application, the feature vectors have a dimensionality of 1,470. The associated model would require 1,470 weights to be learnt. Such a large number of features slows down the learning process considerably which can, in some cases, be too slow for practical deployment. As an example, with the set of networks described in Section 5 and limiting the number of parameters to be estimated to eight (by using only two parameters to represent the pairwise matrix, see Section 8), learning the model already took approximately 3 hours.

Some approaches based on L_1 regularization have shown that feature selection can be performed in conjunction with CRF training (Vail et al. 2007). In effect, these techniques find the features which are not useful for classification and drive the associated weights to zero. The resulting model is sparse. However, the optimization is still performed directly in the feature space leading to long learning times.

As a consequence, we approach the learning problem from a different angle. The specificity of our approach lies in the way a pairwise CRF is defined; the corresponding formulation is given in Equation (4). Specifically, it lies in the way the association potentials $\mathbf{A}$ are defined: *the association potentials* $\mathbf{A}$ *are classifiers*. For the reasons developed in Section 6.3, $\mathbf{A}$ is implemented as a Logitboost classifier (Friedman et al. 2000).

Using a classifier as an association potential significantly reduces the number of weights in the vector $\mathbf{w}$. This comes from the following aspect. As defined in Equation (4), the association potentials $\mathbf{A}$ are a function of the features $\mathbf{f}_A$. Unlike a standard log-linear model, the weights defining the model do not directly multiply the (potentially high-dimensional) feature vectors $\mathbf{f}_A$ but the output of $\mathbf{A}$. Since $\mathbf{A}$ is a Logitboost classifier, its output is a distribution over class labels. This distribution comes in the format of a vector whose dimensionality is equal to the number of classes. The latter is usually much lower than the dimensionality of $\mathbf{f}_A$. As a consequence, the number of weights in the vector $\boldsymbol{\lambda}_A$ is much smaller than in the original set of weights $\mathbf{w}$, which can significantly accelerates the learning procedure.

The proposed learning approach proceeds as follows. A Logitboost classifier $\mathbf{A}$ is first learnt on the set of feature vectors $\{\mathbf{f}_A\}$. This process runs through each of the dimensions of the feature vectors. However, unlike in standard learning in log-linear models, this first phase does not require the join optimization of the weights on the local and pairwise features. Once the association potential is learnt, the weights $\boldsymbol{\lambda}_A$ of the CRF model multiply the output of $\mathbf{A}$ and are learnt during a second phase via the modified version of MPL training presented in Section 3.6.

In addition, a Logitboost classifier can be made to return a *normalized* distribution over classes. This implies that $\mathbf{A}$ (which is a Logitboost classifier) does not need to be multiplied by the weights $\boldsymbol{\lambda}_A$. The reason for this is the following. The role of the weights $\boldsymbol{\lambda}_A$ and $\boldsymbol{\lambda}_I$ is to balance the influence of local and neighborhood information when computing the distribution given by Equation (4). When the output of $\mathbf{A}$ is not bounded, the weights $\boldsymbol{\lambda}_A$ re-scale $\mathbf{A}$'s output so that it is numerically comparable to $\mathbf{I}$'s output. When $\mathbf{A}$'s output is normalized, re-scaling becomes unnecessary and the weights $\boldsymbol{\lambda}_I$ suffice to balance the effect of local and neighborhood information. As a consequence, the proposed CRF formulation avoids the optimization of the local weights $\boldsymbol{\lambda}_A$. Modeling the interaction potential $\mathbf{A}$ by a classifier is in effect equivalent to performing a second feature extraction process. The first pass of feature extraction provides the feature vector $\mathbf{f}_A$. The second pass provides a normalized distribution over class labels, that is, the output of $\mathbf{A}$.

A formulation of pairwise CRFs based on association potentials implemented as classifiers had not been combined with semi-supervised MPL learning (semi-supervised MPL is detailed in Section 3.6). Similar approaches using Boosting for building local potentials have been presented, see Heitz et al. (2009) for instance. Note that not only can Boosting classifiers be used, but any classifier returning a distribution over class labels, see for instance Vallespi-Gonzalez (2008).

3.6. Semi-supervised MPL

Training very large networks (such as those presented in Section 4) with a fully supervised approach is not practical as it requires labeling every single laser return in the training set. Therefore, we resort to a novel semi-supervised extension of the MPL procedure.

The formulation of the corresponding pseudo-likelihood function is as follows:

$$PL(\mathbf{x}|\mathbf{z}) = \prod_i \frac{1}{Z(MB(x_i),\mathbf{w})} \underbrace{\phi_A(x_i)}_{\text{Local Potential}} \prod_{k\in MB(x_i)} \overbrace{\phi_I(x_i,x_k)}^{\text{Pairwise Matrix}} \underbrace{\phi_A(x_k)}_{\text{Neighbor Local Potential}} \qquad (12)$$

This equation will be explained shortly. It assumes that the CRF is in the form given in Equation (1). The terms ϕ_A and ϕ_I are defined in Equations (2) and (3), respectively.

The pseudo-likelihood formulation proposed above can be recovered from the general form of the pseudo-likelihood function. The general formulation was given in Equation (11) and is repeated here

$$PL(\mathbf{x}|\mathbf{z},\mathbf{w}) = \prod_i p(x_i|MB(x_i),\mathbf{w})$$

Using the form of a CRF model given in Equation (1), the pseudo-likelihood can be re-written as

$$p(\mathbf{x}|\mathbf{z}) = \prod_i \frac{1}{Z(MB(x_i),\mathbf{w})}\,\phi_A(x_i) \prod_{k\in MB(x_i)} \phi_I(x_i,x_k), \qquad (13)$$

The term $\phi_A(x_k)$ does not appear in the latter equation while it does in the expression we are trying to recover (Equation (12)). As explained in Section 3.5, a MPL approach requires the neighbor labels to be known during training. These labels correspond here to the term x_k. However, here we wish to relax the assumption that neighbor labels are known during training. To do so, we marginalize out x_k. This corresponds to multiplying the matrix $\phi_I(x_i,x_k)$ by the distribution over x_k, which is given by (normalizing) the term $\phi_A(x_k)$, that is,

$$\phi_I(x_i,x_k)\,\phi_A(x_k) = \overbrace{\begin{bmatrix} \phi_{I_{11}} & \dots & \phi_{I_{1L}} \\ \vdots & \ddots & \\ \phi_{I_{L1}} & & \phi_{I_{LL}} \end{bmatrix}}^{\text{Pairwise Matrix}} \underbrace{\begin{bmatrix} \phi_A(x_k^1) \\ \vdots \\ \phi_A(x_k^L) \end{bmatrix}}_{\text{Neighbor Local Potential}}. \qquad (14)$$

The additional indices ranging from 1 to L refer to the various instances of label x_k given that the classification problem involves L classes. Marginalizing out a variable is a standard inference mechanism. It is applied here to the MPL formulation to relax the requirement of having fully labeled data. This marginalization leads us to the formula we were trying to recover:

$$PL(\mathbf{x}^L,\mathbf{x}^U|\mathbf{z}) = \prod_{i\in L} \frac{1}{Z(MB(x_i),\mathbf{w})}\phi_A(x_i) \prod_{k\in MB(x_i)\cap(L\cup U)} \phi_I(x_i,x_k)\,\phi_A(x_k), \qquad (15)$$

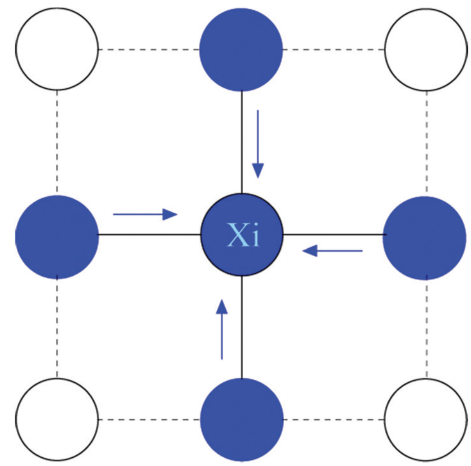


Fig. 4. Illustration of the proposed semi-supervised MPL learning. This figure should be viewed in parallel with Figure 3 which illustrates the standard MPL approach. The association potential **A** is applied to the central node x_i and generates the term $\phi_A(x_i)$ of Equation (12). The association potential is also applied to each of the four neighbors in the Markov blanket of x_i. In these cases, it generates the terms $\phi_A(x_k)$. The four neighbor potentials are "sent" across the links as indicated by the blue arrows. This operation corresponds to the multiplication of the neighbor potentials to the pairwise matrix in Equation (12). It results in marginalizing out the variables x_k. The word "sent" is used here because marginalization is an inference mechanism which is interpreted in the context of BP as sending messages. Marginalization being performed, the resulting terms are multiplied to the local potential $\phi_A(x_i)$. This sequence of operations allows the local likelihood $p(x_i|MB(x_i),\mathbf{w})$ to be computed at node x_i. The whole process is repeated at each of the labeled nodes to complete the calculation of the pseudo-likelihood; this corresponds to the outer product in Equation (12). Note that this overall process only requires a subset of nodes to be labeled. Then, the algorithm is able to exploit the information provided by the neighborhood independently of whether neighbor nodes are labeled. This is in contrast to the standard MPL approach which requires all of the nodes to be labeled.

where L and U explicitly indicate labeled and unlabeled nodes, respectively. The intuition behind the marginalization process is described in Figure 4.

When learning a CRF based on this modified MPL approach, each of the nodes indexed by i needs to be labeled. However, their neighbors, referred to as $MB(x_i)$, do not need to be labeled since they intervene via their local potentials $\phi_A(x_k)$ rather than their label. This differentiates the above formulation from the standard MPL approach which requires all of the nodes to be labeled.

In terms of implementation, the association potential **A** is learned first. As discussed in Section 3.4, the Logitboost algorithm is used for this first phase of the training. Also, since a Logitboost classifier returns a *normalized* distribution over classes, the weights $\boldsymbol{\lambda}_A$ need not be learned and are simply set to one (as discussed in Section 3.5). Then, the above semi-supervised version of MPL learning is applied to learn the terms related to the interaction potential **I**. In our implementation the pairwise matrix ϕ_I is directly learned without explicitly learning the terms λ_I and I. The optimization algorithm used for this second phase of the learning is a Broyden—Fletcher—Goldfarb–Shanno (BFGS)-based technique (Sutton and McCallum 2006) (as implemented by the Matlab function "fmincon").

Such a MPL formulation allows us to investigate the performance of the proposed spatio-temporal model applied to large-scale networks (presented in the following sections). Equation (12) corresponds to a simple yet efficient extension of MPL learning to partially labeled data. With this formulation it is possible to exploit the connections between labeled nodes and all of their neighbors, independently of whether the latter are labeled. Note that the derivations presented here corresponds to the case of a constant pairwise potential matrix, that is, independent of the observations unlike what is shown in the CRF definition in Equation (4). Section 5.2 also presents an approach by which the pairwise potentials can be made dependent on the observations. The learning procedure remains the same but the term $\mathbf{f}_I(\mathbf{z}, \mathbf{x})$ in Equation (4) plays the role of a switch (implemented as a classifier) which allows the use of several pairwise matrices and leads to a more accurate modeling of the network links.

4. From Laser Scans to Conditional Random Fields

This section describes how the graph structure of a CRF can be generated from laser data. Each node of the resulting network corresponds to a laser return whose hidden state corresponds to object types: car, trunk, foliage, people, wall, grass and other (the class "other" representing any other type of object). These classes were chosen because they cover the set of typical objects encountered in the datasets used in this paper. The choice of classes is task specific, e.g. for the task of identifying moving objects the classes chosen would be cars, pedestrians and bicycles (Katz et al. 2008a).

This section is organized according to the increasing complexity of the presented networks. The representation of spatial relationships is first introduced by modeling single laser scans as chain CRFs. Then, consecutive scans are connected according to their alignment to model temporal relationships and effectively implement operations such as filtering and smoothing.

4.1. Spatial Reasoning

CRFs were selected as the basis for the proposed model owing to their ability to encode spatial and temporal dependencies in the classification process. Spatial dependencies come from the natural organization of the laser data into clusters of returns: spatially close samples are likely to have the same label. Temporal dependencies come from overlapping observations performed at successive times: samples generated by the same object and acquired at successive times are likely to be dependent. In the context of a CRF network, these two types of dependencies are represented by two sets of links.

In a given laser scan, spatial dependencies can be represented by the CRF model displayed in Figure 5(a). This model is a chain network connecting the successive returns in the scan. Laser returns which are separated by more than a few meters from each other are not likely to be dependent. As a result, network links are instantiated only between returns separated by a distance inferior to a certain threshold. In our implementation, this threshold was set to 1 m.

By performing probabilistic inference, the classes of the laser returns connected in the model are jointly estimated. Local observations $\mathbf{z}_i$ are passed onto each node via the association potentials **A** and the resulting local estimates are propagated in the network via the interaction potentials **I**.

Since this first type of network is a chain, inference is exact and can be performed with BP (introduced in Section 3.3). The tests with this model (Section 7) are performed with fully labeled data to first verify performance gains. Training is based on a standard CRF learning procedure: VEB (Liao et al. 2007). Experiments involving the proposed semi-supervised MPL procedure and partially labeled data are reported in Section 8.

4.2. Temporal Reasoning

Owing to the sequential nature of robotics applications, a substantial amount of information can be gained by taking into account temporal dependencies. Using the same elementary components of CRFs, i.e. nodes and links, we now build a model achieving temporal smoothing in addition to exploiting the geometric structure of laser scans. This model is illustrated in Figure 5(b).

In this work, the links modeling the temporal dependencies are instantiated such that they represent the associations obtained by the Iterative Closest Point (ICP) matching algorithm (Zhang 1994). The resulting network connects successive chain networks and is characterized by a cyclic topology. This network models spatial correlations via links connecting the nodes within one scan and temporal correlations via links connecting the successive chain networks.

Corresponding to different variants of temporal state estimation, our spatio-temporal model can be deployed to perform three types of inference:

- Off-line smoothing: all scans in a temporal sequence are connected using ICP. Loopy BP is then run in the whole network to estimate the class of each laser return in the sequence. During loopy BP, each node sends

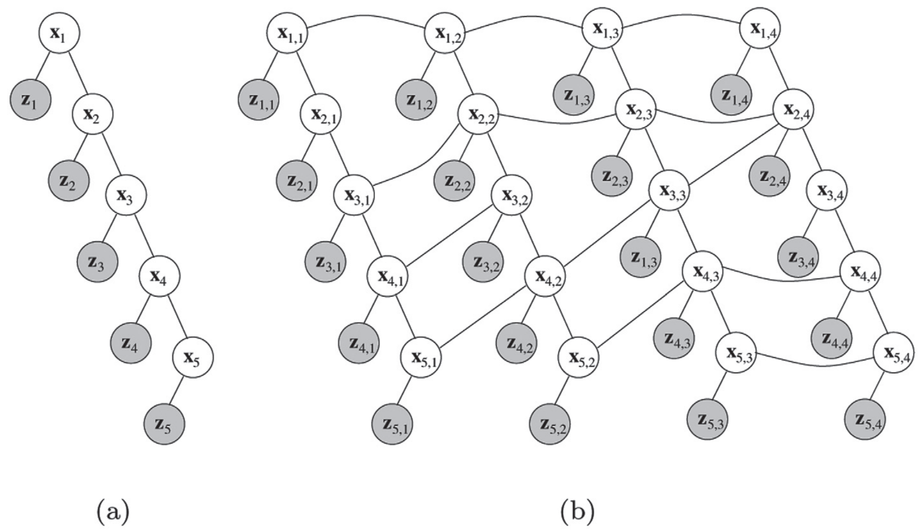


Fig. 5. (a) Graphical model of a chain CRF for single time slice object recognition. Each hidden node x_i represents one (non-out-of-range) return in a laser scan. The nodes $\mathbf{z}_i$ represent the features extracted from the laser scan and the corresponding image. (b) Graphical model of the spatio-temporal CRF. Nodes $x_{i,j}$ represent the ith laser return observed at time j. Temporal links are generated between time slices based on the ICP matching algorithm.

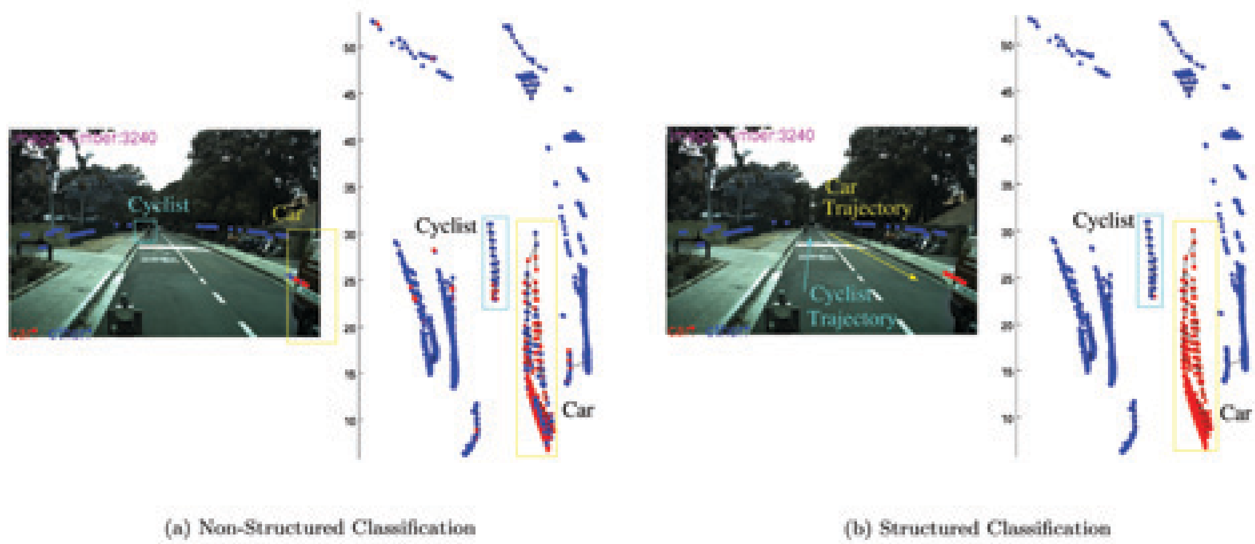


Fig. 6. Example of classification improvements obtained with a spatio-temporal CRF. (a) The estimates obtained with local classification (i.e. using only the **A** functions in Equation (4)). (b) The estimates obtained using a CRF as the model displayed in Figure 5(b). The right part of each figure shows a sequence of laser scans projected in a global frame. The units of the axes are meters. The estimates are indicated by the color of each return: red for car and blue for other. The black links represent the temporal edges of the underlying network. The left part of each figure displays the last image of the sequence as well as the projection in the image of the corresponding laser returns. In the sequence used to generate this figure, a car is moving toward our vehicle and a cyclist is moving away from our vehicle. Based on local classification (a), some of the returns are mis-classified since all of the returns associated to the cyclist should be blue and all of the returns associated to the car should be red. Based on structured classification (b), almost all returns are classified correctly.

messages to its neighbors through structural and temporal links (vertical and horizontal links in Figure 5(b), respectively).

- On-line fixed-lag smoothing: here, scans are added to the model in an on-line fashion. To label a specific scan, the system waits until a certain number of additional

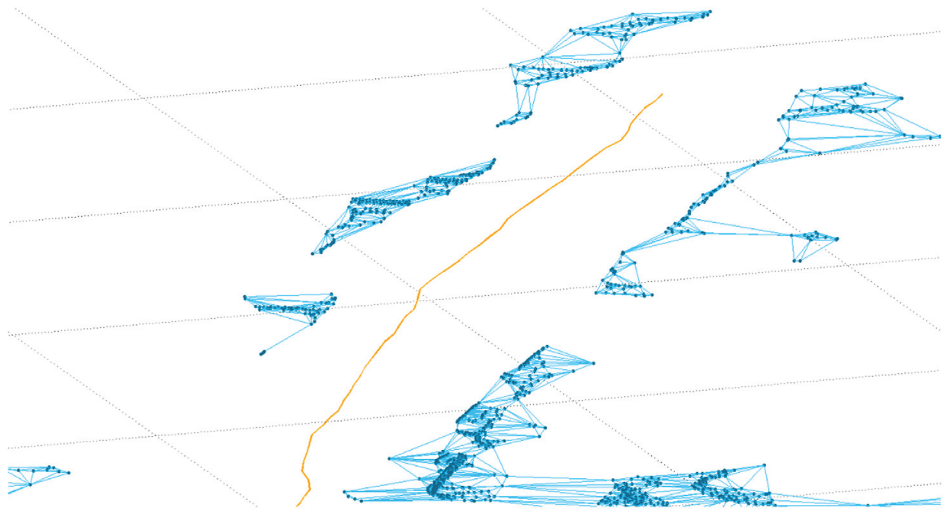


Fig. 7. Representation of a Delaunay CRF generated from urban data (the dataset is described in Section 7.1). The trajectory of the vehicle is displayed in orange. Laser returns are assembled into a mesh by means of the Delaunay triangulation. Returns and triangulation links are plotted in dark and light blue, respectively. For this display the maximum link length is set to 2 m instead of 50 cm as in the deployed version of the system.

scans become available. It then runs loopy BP which combines past and future observations to estimate the network's labels.

- On-line filtering: in this case the spatio-temporal model includes scans up to the current time slice resulting in an estimation process which integrates prior estimates.

An example of on-line fixed-lag smoothing is presented in Figure 6. It can be seen in this figure that the sets of nodes corresponding to the car and the cyclist are classified correctly when a CRF is used to integrate spatial and temporal information. The estimates given by local estimation, that is, estimation which does not take into account the information provided by the network links, are only partially correct.

Since spatio-temporal networks contain cycles, inference is based on loopy BP and is as a result only approximate. Alternatives to approximate techniques are discussed in Section 5.3. The tests with this model (Section 7) are performed with fully labeled data to first verify performance gains. Training is based on a standard CRF learning procedure: VEB (Liao et al. 2007). Experiments involving the proposed semi-supervised MPL procedure and partially labeled data are reported in Section 8.

5. 2D Semantic Mapping

We now show how a larger scale CRF network can be built to generate a semantic map. The proposed map building approach requires as an input a set of already aligned 2D laser scans. In our implementation, the ICP algorithm was used to perform scan registration. However, in spatially more complex datasets containing loops, consistently aligned scans can be generated using various existing SLAM techniques (Williams 2001; Thrun et al. 2005; Bosse and Zlot 2008).

In this section, we present three types of CRFs which will be compared to better understand how to model spatial dependencies. We explain how the three different models can be instantiated from aligned laser data and indicate which inference and learning techniques are used in each case. As in the previous models, the hidden states represent the object types of the laser returns.

5.1. Delaunay CRF

In this first type of network, the connections between the nodes are obtained using the Delaunay triangulation procedure (De Berg et al. 2000) which efficiently finds a triangulation with non-overlapping edges. The system then removes links which are longer than a pre-defined threshold (50 cm in our application) since distant nodes are not likely to be strongly correlated. An example of Delaunay CRF graph is shown in Figure 7.

Since a Delaunay CRF contains cycles, inference is performed with loopy BP. To train a Delaunay CRF, the semi-supervised version of MPL learning detailed in Section 3.6 is used.

Structured classification as performed by CRFs should improve on classification results since neighborhood dependencies are accounted for by interaction potentials. However, as will be illustrated by the experimental results, the Delaunay CRF does not in fact improve the classification by much. This is due to spatial correlation modeling being too coarse. In the Delaunay CRF, the terms ϕ_I in

Equation (12) are learned as a constant matrix instantiated at each of the links. This gives the network a smoothing effect on top of the local classification. Since all of the links are represented with the same matrix, only one type of node-to-node relationship is encoded. In our application, the learning results in a pairwise matrix close to the identity matrix which means that it models the following type of correlation: “two neighbor nodes are likely to have the same label”. While this type of link may be appropriate for modeling a single scan or very structured parts of the environment, it may over-smooth the estimates in areas where the density of objects increases.

5.2. Delaunay CRF with Link Selection

To model more than one type of node-to-node relationship, a second type of network is introduced in which interaction potentials $\mathbf{f}_I$ are function of the observations. This means that the function $\mathbf{f}_I$ in Equation (4) is now modeled while it was not used in the previous types of networks. In particular, it implements a Logitboost binary classifier which plays the role of a switch and allows different pairwise matrices to be used to represent the network links.

Depending on the output of the interaction potential $\mathbf{f}_I$ (the Logitboost binary classifier) the pairwise potential ϕ_I takes on different values, that is, the value of $\mathbf{f}_I$ dictates the selection of one pairwise matrix amongst a set of them. In this way, the type of pairwise relationship instantiated is changed depending on the observations at the two ends of a link.

The Logitboost binary classifier estimates the similarity of two nodes and is trained using the difference of observations $\mathbf{d}_{ij} = |\mathbf{z}_i - \mathbf{z}_j|$ between the two nodes i and j at the ends of a link. The operator $|\cdot|$ refers to the absolute value and is applied to each dimension of the vector. Here $\mathbf{d}_{ij}$ is given the label 1 if the two nodes have the same label, otherwise it is given the label 0. The training of this classifier is performed before running MPL learning, as is done for the Logitboost classifier modeling the association potential $\mathbf{A}$. Since this second type of network contains loops, inference is also performed using loopy BP.

As will be shown by the experimental analysis in Section 8.2, the accuracy of this second type of network improves over local classification which confirms our analysis of the role played by network links: link instantiation must be determined on a case-by-case basis not to over-smooth the estimates. This analysis will be further developed in Section 9.

5.3. Tree CRF

The previous two types of networks contain cycles, which implies the use of an approximate inference algorithm. We now present a third type of network which is cycle free. To design non-cyclic networks we start from the following observation: laser returns in a scan map are naturally organized into clusters. These clusters can be identified by analyzing the connectivity of the Delaunay graph and finding its disconnected sub-components. Disconnected sub-components appear when removing longer links in the original triangulation. In Figure 8, the extracted clusters are indicated by green rectangles. The Delaunay triangulation is used here to cluster the data which leads to the definition of a graph. Edges could also be defined using k-nearest neighbors or by connecting all of the neighbors within a fixed radius.

Once the clusters are identified, the nodes of a particular cluster are connected by a tree of depth one. To accomplish this, a root node is instantiated for each cluster and each node in the cluster becomes a leaf node. The root node does not have an explicit state. From the point of view of BP, it is neutral since its local potential is maintained uniform. Such a root node has in fact no physical meaning but simply allows a tree structure to be created: it provides a node which all of the cluster node can attach to. This results in a tree-like topology which is cycle free and, as a consequence, permits the use of an exact inference technique. With this third type of network, BP is used for inference. A tree CRF does not encode node-to-node smoothing but rather performs smoothing in a whole cluster at once. The trees associated with the clusters in Figure 8 are represented by green volumes. Computing the minimum spanning tree of the points in each cluster would be another way to build trees but the computational cost would be higher, of the order of $O(V \log E)$, where V is the number of points or vertices and E the number of edges (Cormen et al. 2001). The proposed approach has a complexity of $O(V)$.

The possibility of using exact inference is a strong advantage since in the case of approximate inference (based on loopy BP for example) the convergence of the algorithm is not guaranteed. As suggested by Murphy et al. (1999), while convergence of loopy BP in cyclic networks is not proven, it can be checked experimentally. To evaluate the convergence of the inference procedure in the two previous networks, an empirical convergence analysis is presented in Section 8.5. The tree CRFs are learnt with the semi-supervised MPL approach proposed in Section 3.6.

6. Features

The CRF model used in this work as defined in Equation (4) involves the feature functions $\mathbf{f}_A$ and $\mathbf{f}_I$. This section provides details of the features generated by the function $\mathbf{f}_A$. As discussed in the previous section, $\mathbf{f}_I$ is either not used (for instance, in the cases of Delaunay CRFs without link selection and tree CRFs) or implemented as a Logitboost binary classifier (as in the case of Delaunay CRFs with link selection).

For clarity, in this section the output of $\mathbf{f}_A$ will be referred to as $\mathbf{f}$. Here $\mathbf{f}$ is a high-dimensional vector computed for each laser return in a scan. Its dimensionality is 1,470.

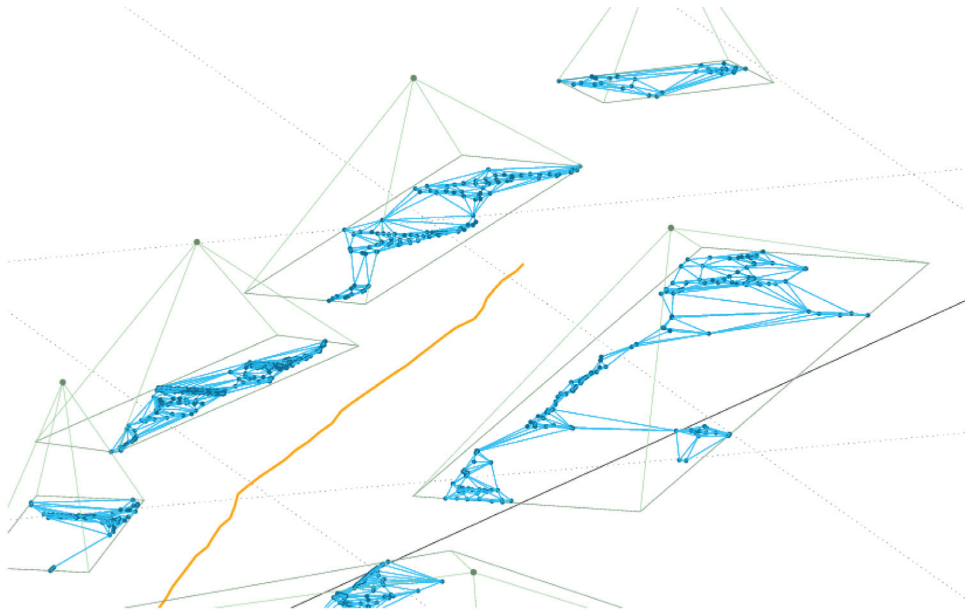


Fig. 8. Representation of a tree CRF in one region of a graph generated from data (the same scene as shown in Figure 7). The trajectory of the vehicle is displayed in orange. Laser returns are first assembled into a mesh by means of the Delaunay triangulation. Returns and triangulation links are plotted in dark and light blue, respectively. By analyzing the connectivity structure of the graph in blue, clusters of returns are extracted. Identified clusters are indicated by the green rectangles. Once the clusters of returns have been formed, the triangulation links are disregarded. A root node is then created for each cluster and linked to all of the returns in the clusters. The root nodes are plotted as green nodes above the ground. For clarity all of the pairwise connections between the root nodes and the nodes in the corresponding cluster are not displayed. However, the overall tree structures are represented by the volume materialized with the green edges.

It results from the concatenation of geometric and visual features:

$$\mathbf{f} = [\mathbf{f}_{\text{geo}}, \mathbf{f}_{\text{visu}}]. \tag{15}$$

Geometric features are described first. We then show how visual features can be extracted via registration of the laser data with respect to the imagery. Finally, we explain how the use of Logitboost allows the selection of effective features for classification.

The offset between the positions of the laser and the camera on the vehicle generates projection artifacts (Douillard 2009). A simple heuristic is applied to filter out returns which are potentially mis-projected. This heuristic consists in working through the projected scan, from the far right to the center of the image. We only keep the returns which are closer to the center than the previous returns in the scans. A second pass is run from the left to the center of the image. The selected returns form a scan whose projection in the image is concave, which has the effect of filtering out projection artifacts.

6.1. Laser Features

Geometric features capture the shape of objects in a laser scan. The geometric feature vector computed for one laser return has a dimensionality of 231 and results from the concatenation of 38 different multi-dimensional features. Only the features which are the most useful for classification are presented here. In Section 6.3, it is explained how features can be ranked according to their usefulness. Some of these 38 features are as follows:

$$\mathbf{f}_{\text{geo}} = \left[\mathbf{f}_{\text{nAngle}}, \mathbf{f}_{\text{minAngle}}, \mathbf{f}_{\text{cSplineFit}}, \mathbf{f}_{\text{cEigVal1}}, \mathbf{f}_{\text{maxFilter}}, \dots\right]. \tag{16}$$

The features $\mathbf{f}_{\text{nAngle}}$ and $\mathbf{f}_{\text{minAngle}}$ respectively refer to the norm and the minimum of a multi-dimensional angle descriptor $\mathbf{f}_{\text{angle}}$ which has been designed for this application. Its kth dimension is computed as follows:

$$\mathbf{f}_{\text{angle}}(k) = |\angle(\mathbf{r}_{i-k} - \mathbf{r}_i, \mathbf{r}_{i+k} - \mathbf{r}_i)|, \tag{17}$$

where $\angle(\mathbf{a}, \mathbf{b})$ represents the angle formed by two vectors $\mathbf{a}$ and $\mathbf{b}$; in our implementation, an angle is expressed modulo π. The vector $\mathbf{r}_i$ refers to the 2D position of the ith return in the scan being processed, and k varies from -10 to $+10$. The dimensionality of both $\mathbf{f}_{\text{nAngle}}$ and $\mathbf{f}_{\text{minAngle}}$ features is one. In the various models learned across the experiments, features computed from the descriptor $\mathbf{f}_{\text{angle}}$ were amongst the best for the recognition of tree trunk and pedestrian classes. In these two cases, features capture typical curvilinear shapes when, for example, the scan hits these objects at about 1 m above the ground.

The features $\mathbf{f}_{\text{cSplineFit}}$ and $\mathbf{f}_{\text{cEigVal1}}$ characterize the shape of a cluster of returns. Clusters are extracted within one scan based on a simple distance criteria: returns closer than a threshold (we used 1 m in our applications) are associated with the same cluster. Based on the identified clusters, various quantities are computed. Feature $\mathbf{f}_{\mathrm{cSplineFit}}$ is obtained as the error of the fit of a spline to the curve formed by the cluster of 2D returns. Feature $\mathbf{f}_{\mathrm{cEigVal1}}$ is the largest eigenvalue of the covariance matrix describing the cluster. While not being ranked amongst the most important features, cluster-based features are useful in classifying all of the seven classes considered in this work. Note that all of the returns within one cluster receive the same cluster features.

The feature $\mathbf{f}_{\mathrm{maxFilter}}$ is obtained as the maximum response of a filter run in a window centered on a given return. This filter is essentially a low-pass discrete filter processing a scan represented as a sequence of angles. This filter provides a multi-dimensional representation whose various dimensions have proven useful in detecting the class car and the class pedestrian.

6.2. Vision Features

A CRF learned with a Logitboost-based algorithm can integrate both geometric information and any other type of data, in particular, visual features extracted from monocular color images. Visual features are extracted as follows. A ROI is defined around the projection of each laser return in the image and a set of features is computed within this ROI. The parameters required to perform the projection are defined through the camera laser calibration procedure developed by Zhang and Pless (2004). The size of the ROI is changed depending on the range of the return. This provides a mechanism to deal with changes in scales across images. It was verified that the use of size varying ROIs improves classification accuracy by 4%. Examples of ROIs generated by the system are shown in Figure 9.

To obtain a visual feature vector $\mathbf{f}_{\mathrm{visu}}$ of constant dimensionality despite size varying ROIs, vision features are designed which are independent of patch size. This is achieved by using distribution-like features (e.g. a histogram with a fixed number of bins) and whose dimensionality is constant (e.g. equal to the number of bins in the histogram). A larger ROI leads to a better sampled distribution (e.g. a larger number of samples in the histogram) while the actual feature dimensionality remains invariant.

The overall visual feature vector $\mathbf{f}_{\mathrm{visu}}$ associated with each return has a dimensionality of 1,239 and results from the concatenation of 51 multi-dimensional features computed in the ROI. Only the most useful subset of features are described here. The presentation follows the ranking of the features obtained as explained in Section 6.3:

$$\mathbf{f}_{\mathrm{visu}} = \left[\mathbf{f}_{\mathrm{pyr}}, \mathbf{f}_{\mathrm{hsv}}, \mathbf{f}_{\mathrm{rgb}}, \mathbf{f}_{\mathrm{hog}}, \mathbf{f}_{\mathrm{haar}}, \mathbf{f}_{\mathrm{lines}}, \mathbf{f}_{\mathrm{sift}}, \ldots\right]. \tag{18}$$

The feature $\mathbf{f}_{\mathrm{pyr}}$ contains texture information encoded as the steerable pyramid (Simoncelli and Freeman 1995) coefficients of the ROI as well as the minimum and the maximum of these coefficients. These extrema are useful in classifying cars which from most point of views have a relatively low texture maximum due to their smooth surface.

The features $\mathbf{f}_{\mathrm{hsv}}$ and $\mathbf{f}_{\mathrm{rgb}}$ contain a 3D histogram of the RGB and HSV data in the ROI, respectively. A 3D histogram is built as follows. The RGB or the HSV space defines a 3D space which is discretized to form a 3D grid. Each cell of the grid is a bin in the histogram. Based on the RGB or HSV coordinates of a pixel, a sample is added to the appropriate bin. HSV and RGB histograms were selected in the representation of each of the seven classes. On average, HSV histogram feature received a better rank than RGB-based features. This confirms the analysis made in various studies (Douillard 2009). An example of RGB histogram is shown in Figure 1.

The features $\mathbf{f}_{\mathrm{hog}}$ are histograms of gradients (see http://www. robots. ox. ac. uk/~vgg/research/caltech/phog.html). These features are selected by the learning algorithm for the modeling of the classes car, pedestrian and grass.

The feature $\mathbf{f}_{\mathrm{haar}}$ contains Haar features computed in the return's ROI according to the integral image approach proposed in (Viola and Jones 2004). Haar features are useful in classifying the classes tree trunk and foliage.

The feature $\mathbf{f}_{\mathrm{lines}}$ contains a set of quantities describing the lines found by a line detector (see http://www.cs.uiuc.edu/homes/dhoiem/) in the ROI. These quantities include the number of extracted lines, the maximum length of these lines and a flag which indicates whether the line of maximum length is vertical. These features have been useful in classifying all of the seven considered classes.

The feature $\mathbf{f}_{\mathrm{sift}}$ contains the Sift descriptor (Lowe 2004) of the ROI's center as well as the number of Sift features found in the ROI. Sift features were selected during the training of various models to represent the classes grass and other.

6.3. Feature Selection and Dimensionality Reduction

The learning procedure described in Section 3.6 is based on a version of Logitboost which uses decisions stumps as weak classifiers. With the latter algorithm, the dimensions of the feature vector can be ranked according to their ability to discriminate between the various classes. This ranking is obtained once the algorithm has processed each dimension of the feature vector. For each dimension, it attempts to separate two classes based on a simple threshold. Once the threshold has been computed, the algorithm estimates the quality of the separation provided by this threshold. This is referred to as the quality estimate q_k, where k is the index of the associated dimension. Once the algorithm has inspected all dimensions of the feature vector, it selects the dimension associated with the best q_k and augments the model accordingly. This completes one iteration of Logitboost. The same process is repeated until a pre-defined number of iterations is reached. Effectively, the algorithm implements a greedy search by finding the best feature at each iteration. This

Fig. 9. Examples of ROIs generated by the system. The ROIs are indicated by the yellow rectangles, the laser returns are indicated by the yellow crosses at the center of the rectangles. It can be seen that the size of the ROI is decreased for longer ranges. As discussed in Section 3.2, the fact that these ROIs overlap and generate feature vectors with an overlapping content is not a problem from the point of view of a CRF. Since a CRF is globally conditioned on the set of observations, it can readily integrate the content of overlapping feature vectors.

results in an explicit ranking of the features where the rank of a feature is the iteration at which it was selected. As illustrated in Sections 6.1 and 6.2, such a ranking is crucial in the design process since it explicitly indicates which aspect of the data is useful (it allows us to focus the design on features improving the top of the ranking).

Feature selection as performed by Logitboost based on decision stumps can also be seen as a dimensionality reduction procedure. One hundred rounds of Logitboost will result in the selection of 100 dimensions of the original feature vector. This implies that during the testing phase only these 100 selected features need to be computed allowing real-time implementation; see Table 6 later in this article. In addition, since the dimensions of the feature vector are processed one at a time, no overall normalization of the feature vector is required which is an advantage with respect to more standard dimensionality reduction techniques such as those introduced by Fisher (1936), Hotelling (1933), Roweis and Saul (2000), and Tenenbaum et al. (2000).

Another interesting aspect of Logitboost is linked to its ability to process multi-modal data. Features computed from an additional modality can be concatenated to the overall feature vector in the same manner as laser and vision features in Sections 6.1 and 6.2. The feature vector in this sense plays the role of a proxy between the various modalities and the learning algorithm.

7. Experimental Results: Spatial and Temporal Reasoning

7.1. Experimental Setup

Experiments were performed using outdoor data collected with a modified car traveling at a speed of 0–40 km h^{-1} on a university campus and surrounding urban areas. The scenes typically contain buildings, walls, cars, bushes, trees, and lawns. Results are presented using two different datasets to illustrate how the model can be applied to different urban environments. One dataset was acquired in Sydney, Australia, and will be referred to as the Sydney dataset. The other was acquired in Boston, MA, USA and will be referred to as the Boston dataset. Each of the two datasets approximately corresponds to 20 minutes of logging with a monocular color camera and 2D laser scanners. To acquire the two datasets, different vehicles and different sensor brands were used.

Table 1. Classification Accuracy (%) for a Car Detection Problem (Sydney Dataset)

Training set	geo only	visu only	geo+visu	geo+visu
Number of time slices in the model	1	1	1	∓ 10
CRF	68.9	81.8	83.3	88.1
Logitboost	67.6	81.5	83.2	×

The evaluations of the various classifiers are performed using K-fold cross-validation (K being either 5 or 10 depending on the experiments).

7.2. Sydney Dataset

In this first set of experiments we consider two classes: car and other. Seven-class results are presented in Section 8. Table 1 summarizes the experimental results in terms of classification accuracy. The accuracies are given in percentages and computed using 10-fold cross-validation on a set of 100 manually labeled scans selected in the Sydney dataset. In this dataset every return is labeled.

For each cross-validation, different models were trained with 200 iterations of VEB. This number of iterations is an upper limit rather than a fixed number of rounds since VEB modifies the model only if it can find a feature which improves the accuracy. It keeps running until the prescribed number of iterations but may, in effect, stop selecting features before halting. Typically, in our application, VEB stopped selecting features at about iteration 130.

VEB models were computed allowing learning of pairwise relationships only after iteration 100. It was found that this procedure increases the weights of local features and improves classification results.

The first line of Table 1 indicates the types of features used to learn the classifier. Four different configurations were tested: first using geometric features only, second using visual features only, third using both geometric and visual features, and fourth with geometric and visual features integrated over a period of 21 times slices. The second line of Table 1 indicates the number of time slices in the network: “1” means that a network as presented in Figure 5(a) was used; “∓ 10” refers to the classifier shown in Figure 5(b) instantiated with 10 unlabeled scans prior and posterior to the labeled scan.

Two types of classifiers were used: CRFs and Logitboost classifiers. While a CRF takes into account the neighborhood information to perform classification, Logitboost learns a classifier that only supports independent classification, that is, which does not use neighborhood information. This is equivalent to using only the **A** functions in Equation (4) and not modeling the term **I**. Logitboost is used here for comparison purposes in order to investigate the gain in accuracy obtained with a classifier that takes into account the structure of the scan.

The first three columns of Table 1 show that classification results are improving as richer features are used for learning. It can also be seen that the CRF models consistently lead to slightly more accurate classification. In addition, as presented in Section 4.2, a CRF model can readily be extended into a spatio-temporal model. The latter leads to an improvement of almost 5% in classification accuracy (right-most column of Table 1). This shows that the proposed spatio-temporal model, through the use of past and posterior information, performs better. The cross in the bottom right of the table refers to the fact that Logitboost does not allow the incorporation of temporal information in a straightforward manner.

To evaluate the difficulty of the classification task, we also performed Logitboost classification using visual Haar features, which results in the well-known approach proposed by Viola and Jones (2004). The accuracy of this approach is 77.09%, which shows that even the single time slice approach (83.26%) outperforms the reference work of Viola and Jones. The improvement in accuracy obtained in our tests comes from the use of richer features as well as the ability of a CRF to capture neighborhood relationships.

Figure 10 shows four examples of classification results. It can be seen that the spatio-temporal model gives the best results. While the Logitboost classifier tends to alternate correct and incorrect classifications across one scan, the ability of the CRF classifiers to capture the true arrangement of the labels (that is, their structure) is illustrated by the block-like distribution of the inferred labels. Figure 10(b) shows the three classifiers failing in a very dark area of the image (right of the image). In the rest of the image which is still quite dark, as well as in images with various lighting conditions (Figures 10(a), 10(c) and 10(d)) the spatio-temporal model does provide good classification results.

7.3. Boston Dataset

The comparisons between the different setups described in Table 1 were also performed using the Boston dataset. The corresponding results are indicated in Table 2 and were obtained with five-fold cross-validation on a set of 400 manually labeled scans. Each of these scans were fully labeled. For this second set of tests, the classes of interest were also car and other.

Figure 11 shows an example of image extracted from the Boston dataset. The laser scanner used to acquire this data is a Velodyne sensor (see http://www.velodyne.com/lidar/) which is a 3D LIDAR (Light Detection and Ranging) unit composed of 64 2D laser scanners positioned on the device with increasing pitch angle. To perform this set of experiments we used the data provided by 6 of these 64 lasers. Unlike in the Sydney dataset, these lasers are downward

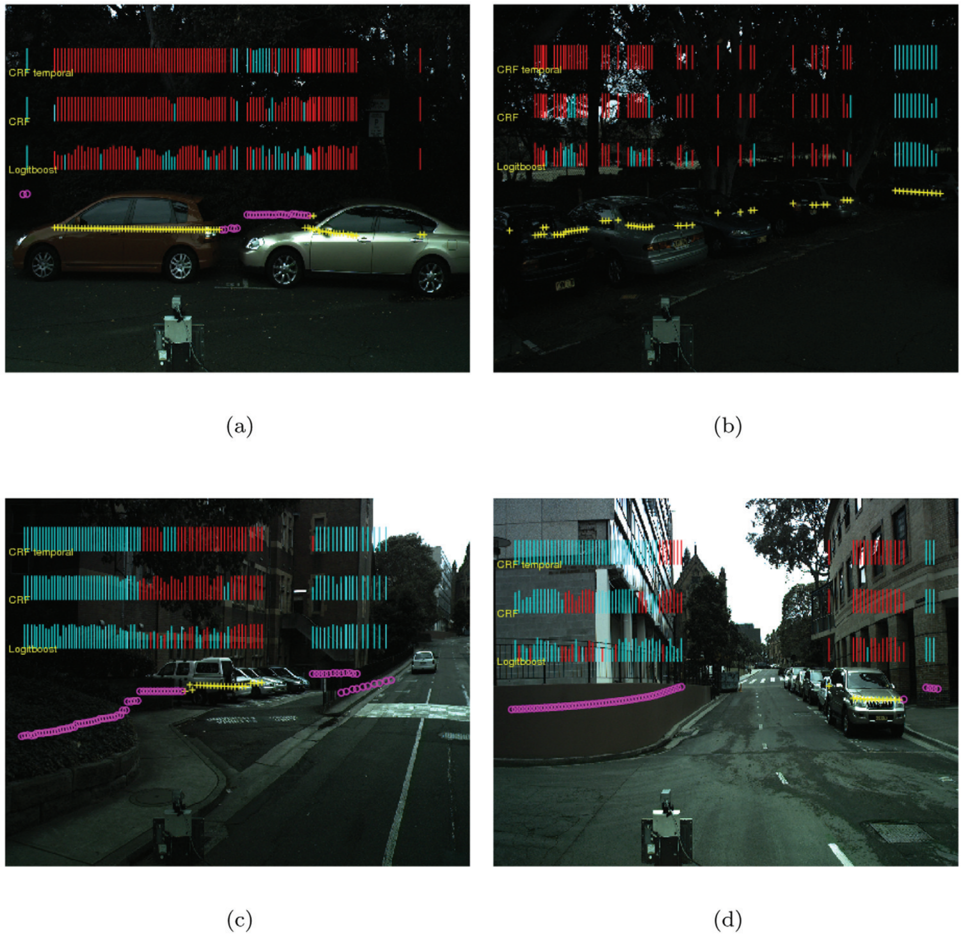


Fig. 10. Examples of classification results. The laser returns are projected into the images and indicated by two markers. Each marker corresponds to one class of object. The markers + in yellow corresponds to the class “car” and the marker ○ in magenta corresponds to the class “other”. The color of the bars above each return indicates the inferred label: red means that the inferred label is car and cyan refers to the label other. The height of the bars represents the confidence associated with the inferred label which is obtained here as its associated probability. The classifiers used to generate the different estimates are specified on the left.

looking. Examples of scans generated by these six lasers are displayed in Figure 11.

The six selected lasers are characterized by a slightly different pitch angle which allows networks to be built from laser returns such as that displayed in Figure 11. While the scan-to-scan links in these networks do not strictly correspond to temporal links (since the Velodyne unit fires the six lasers at the same time) these networks can be thought of as belonging to the category “on-line filtering” described in Section 4.2. Having six laser scanners looking downwards, each of them with a slightly larger pitch angle than the previous one, is approximatively equivalent to using one downward-looking sensor scanning at six consecutive time steps. As a consequence, this setup provides networks of the type “on-line filtering”.

The results in Table 2 show the same trends as Table 1. As more features are added (moving from the left column to the right column of the table), the classification accuracy increases. Classification accuracy is also increased when using CRFs, which unlike Logitboost, enforces consistency in the sequence of estimates. The value “−5” in the right-most column refers to the “on-line filtering” networks which are built by connecting five unlabeled scans before each labeled scan. As with the Sydney dataset, temporal information further improves the performance.

Table 2. Classification Accuracy (%) for a Car Detection Problem (Boston Dataset)

Training set	geo only	visu only	geo+visu	geo+visu
Number of time slices in the model	1	1	1	−5
CRF	81.8	85.0	88.5	90.0
Logitboost	81.4	82.6	88.0	×

It is interesting to note that the classification accuracies achieved on this second dataset for the car detection problem are similar to those achieved on the Sydney

Fig. 11. An example image from the Boston dataset displayed with the associated projected laser returns (in yellow). A part of the CRF network built from these laser returns is displayed in blue in the inset in the top left corner. The labeled scan is that forming the upper side of the network. The image in this inset corresponds to a magnification of the area indicated by the arrow. This figure also illustrates the low resolution of the image, notably with respect to the images in the Sydney dataset (see Figure 10).

dataset: the overall accuracy is about 90% in the Boston dataset and 88% in the Sydney dataset. The resolution of the imagery as well as the density of the laser returns were quite different between the two datasets: the image size is [240 × 376] pixels in the Boston dataset and [756 × 1,134] pixels in the Sydney dataset; on average 300 laser returns were available per image in the Boston dataset against 100 in the Sydney dataset. In spite of these differences, the proposed model provides comparable results which demonstrates its applicability to different types of lasers and cameras.

With respect to the first experiments, the lower resolution of the vision data and the larger number of returns available per image lead to a vision classifier with an accuracy (82.6%) only slightly above that obtained with the laser classifier (81.4%). In the Sydney dataset, a much richer imagery resulted in 13.9% difference in accuracy between the vision-only and the laser-only classifiers. However, in both cases, the model is able to exploit the best of each modality to maintain overall accuracy. This is made possible by the Logitboost algorithm which selects the most discriminative features during learning.

8. Experimental Results: 2D Semantic Maps

This section presents the classification performance obtained with the three models introduced in Section 5. For these three networks, the hidden state of each node ranges over the seven object types: car, tree trunk, foliage, people, wall, grass, and other ("other" referring to any other object type). Results for local classification are first presented in order to provide a baseline for comparison. All of the evaluations were performed using 10-fold cross-validation and the models trained with the semi-supervised MPL learning proposed in Section 3.6.

Table 3. Characteristics of the training and testing sets. These numbers are averaged over the 10 tests used for cross-validation. The number of nodes does not correspond to the number of returns per scan since some returns are disregarded when creating the delaunay triangulation.

	Length of vehicle trajectory	Number of scans *total* labeled	Number of nodes *total* labeled
Training set	2.6 km	*3,843* 72	*67,612* 5,168
Testing set	290 m	*4,27* 8	*7,511* 574

The characteristics of the training and testing sets averaged over the 10-fold cross-validation sets are provided in Table 3. The Sydney dataset was used for these experiments since it contains horizontal 2D laser scans which can be registered using ICP. The registration of downward looking scans is a more complex problem (successive downward-looking scans do not hit objects at the same location requiring the use of a different approach or a full 3D ICP) precluding these mapping experiments using the Boston dataset.

Table 4. Local Classification: Confusion Matrix (Corresponding Accuracy: 90.4%)

	Inferred						
Truth	Car	Trunk	Foliage	People	Wall	Grass	Other
Car	1,967	1	7	10	3	0	48
Trunk	4	165	18	0	4	0	11
Foliage	25	18	1,451	0	24	0	71
People	6	2	2	145	0	0	6
Wall	6	6	21	0	513	1	39
Grass	0	0	1	1	1	146	4
Other	54	5	123	3	24	0	811

Table 5. Local Classification (%): Precision and Recall

	Car	Trunk	Foliage	People	Wall	Grass	Other
Precision	96.6	81.7	91.3	90.1	87.5	95.4	79.5
Recall	97.9	99.3	96.4	99.7	98.5	99.9	95.4

8.1. Local Classification

A seven-class Logitboost classifier is learned and instantiated at each node of the network as the association potential **A** (Equation (4)). Local classification, that is classification which does not take neighborhood information into account, is performed and leads to the confusion matrix presented in Table 4. This confusion matrix displays a strong diagonal which corresponds to an accuracy of 90.4%. A compact characterization of the confusion matrix is given by precision and recall values (for a definition of precision and recall values, see Douillard (2009)). These are presented in Table 5. Averaged over the seven classes, the classifier achieves a precision of 89.0% and a recall of 98.1%.

To obtain these results an additional set of features was used. The original set $\mathbf{f} = [\mathbf{f}_{\text{geo}}, \mathbf{f}_{\text{visu}}]$ described in Section 6 was augmented with the set $\mathbf{f}_{\text{binary}}$. The latter features are generated with Logitboost binary classifiers. For each of the seven classes, a binary classifier is learned using the set $\{\mathbf{f}\}$. This is then run on the training and testing sets and produces a one-dimensional binary output. This output is an estimated class label but is used here as an additional feature concatenated to **f**. The overall operation results in a **f** vector augmented with seven binary-valued dimensions. For this experiment such features are key to the performance of the classifier, resulting in an increase in accuracy of 8.4%. The critical role of the $\mathbf{f}_{\text{binary}}$ features in the Sydney dataset is related to the resolution of the imagery. The Sydney dataset contains the images with the highest resolution which significantly improves local classification. Given this amount of visual data, each binary classifier can, in the ROI associated with a laser return, find the information specific to a class. When the image resolution is low, as in the Boston dataset, the information content of a ROI is blurred and the binary features do not make a difference.

8.2. Delaunay CRF Classification

8.2.1. CRF Without Link Selection The accuracy achieved is 90.3% providing no improvements on local classification. As described in Section 5.2, spatial correlation modeling is too coarse, consisting of only one type of link which cannot accurately model the relationships between all neighbor nodes. Consequently the links represent the single predominant relationship in the data. In the Sydney dataset this neighborhood relationship is “neighbor nodes have the same label”. The resulting learnt links thus enforce this “same-to-same” relationship across the network leading to over-smoothed class estimates. To verify that a better modeling of the CRF links improves the classification performance, we now present results generated by Delaunay CRFs equipped with link selection capabilities.

8.2.2. CRF With Link Selection The accuracy achieved by CRF models with link selection is 91.4%, a 1.0% improvement in accuracy. Since the local accuracy is already high, the improvement provided by the network is better articulated by the reduction in the error rate of 10.4%. This result validates the claim that a set of link types encoding a variety of node-to-node relationships is required to exploit the spatial correlations in the laser map.

8.3. Tree-based CRF Classification

The two types of networks evaluated in the previous section contain cycles and require the use of an approximate inference algorithm. The tree-based CRFs presented in Section 5.3 avoid this issue and allow the use of an exact inference procedure using BP.

This tree network achieves an accuracy of 91.1% which is slightly below the accuracy given by a Delaunay CRF with link selection while still improving on local classification. However, the major improvement brought by this third type of network is in terms of computational time. Since the network has the complexity of a tree of depth one, learning and inference, in addition to being exact, can be implemented very efficiently. As shown in Table 6, a tree-based CRF is 80% faster at training and 90% faster at testing than a Delaunay CRF. Both network types use the same image and scan features which are extracted in 1.2 seconds on average. These quantities are based on a Matlab implementation run on a 2.33 GHz machine. As shown in Table 3, the test set contains 7,511 nodes on average which suggests that the tree-based CRF approach is in its current state close to real time, with feature extraction being the main bottleneck.

8.4. Map of Objects

This section presents a visualization of some of the mapping results. It follows the layout of Figure 12 in which the vehicle is traveling from right to left.

At the location of the first inset, the vehicle is going up a straight road with a fence on the left and right and, from the

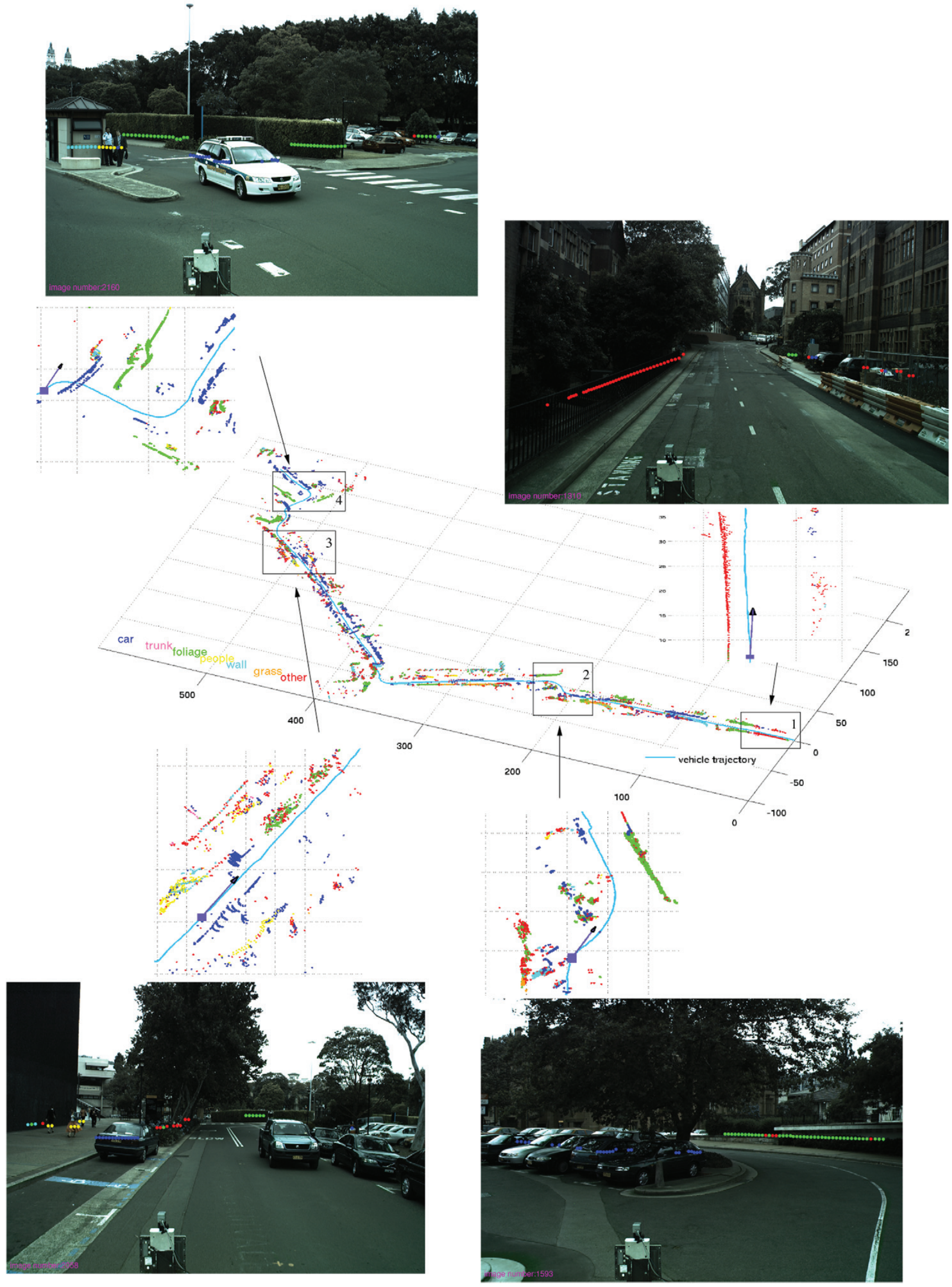


Fig. 12. Visualization of 750 m long portion of the estimated map of objects which has a total length of 3 km. The map was generated using the tree-based CRF model. The legend is indicated in the bottom left part of the 2D plane. The color of the vehicle's trajectory is specified in the bottom right part of the same plane. The coordinates in the plane of the map are in meters. Each inset along the trajectory is magnified and associated with an image displayed with the inferred labels indicated by the colors of the returns. The location of the vehicle is shown in each magnified patch with a square and its orientation indicated by the arrow attached to it. The laser scanner mounted on the vehicle can be seen in the bottom part of each image.

foreground to the background, another fence, a car, a parking meter and a bush. All of these objects were correctly classified (with the fences and the parking meter identified as "other").

In the second inset, the vehicle is coming into a curve facing a parking lot with a bush on the side of the road. Four returns seen in the background of the image are misclassified as "other". The class "other" regularly generates false positives due to the large number of training samples in this class. Various ways of re-weighting the training samples or balancing the training set were tried without significant improvements.

Table 6. Computation Times Averaged Over the 10 Tests Involved in Cross-validation

	Feature Extraction (per scan)	Learning (training set)	Inference (test set)
Delaunay CRF (with link selection)	1.2 s	6.7 min	1.5 min
Tree-based CRF	1.2 s	1.5 min	10.0 s

On reaching the third inset, a car driving in the opposite direction came into the field of view of our vehicle's sensors. The trace left by this car in the map appears in the magnified inset as a set of blue dots along side our vehicle's trajectory. Dynamic objects are not considered explicitly in this work. They are assumed to move slowly enough for ICP to produce correct registrations. In the campus area where this data was acquired, this assumption has proven to be valid. In spite of a few mis-classifications in the bush on the left side of the road, the pedestrians on the footpath as well as the wall of the building are identified correctly.

Entering the fourth inset, the vehicle is facing a second car which appears in the map as a blue trace intersecting the vehicle's trajectory. Apart from one misclassified return on one of the pedestrians, and one misclassified return on the tree in the right of the image, the inferred labels are accurate. Note that the first right return is classified correctly illustrating the accuracy of the model at the border between objects.

An additional set of visualizations of the classification results generated by a semantic map is provided in Figure 13.

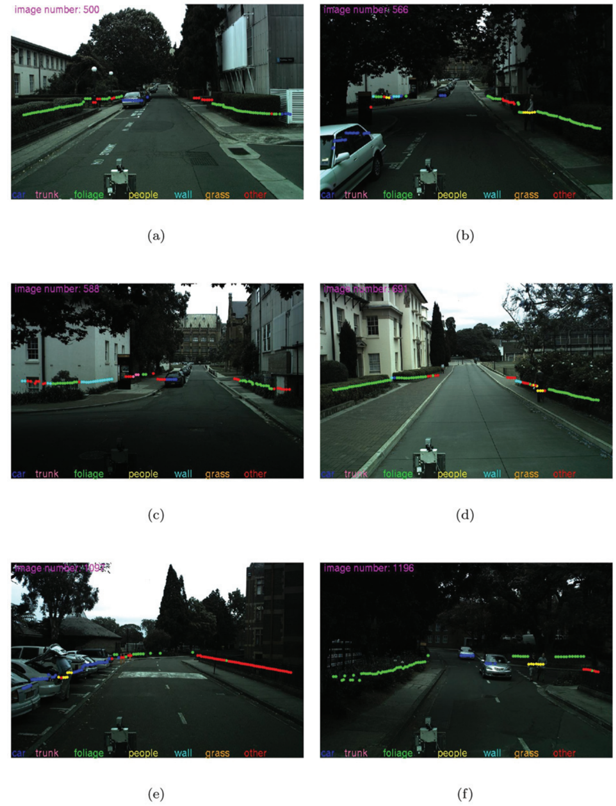


Fig. 13. Examples of classification results extracted from a semantic map such as that shown in Figure 12. Each image presents estimated class labels which are indicated by the colors of the laser returns. The legend is indicated at the bottom of each image. (a) Apart from the few blue returns on the right and the following red return, the classification is accurate. (b) The pedestrian on the right as well as the other pedestrian in the background on the left are identified correctly. The rest of the classification is correct. (c) The wall in the background is correctly classified. The other inferred labels are correct. (d) The estimation of the foliage on the left and the pedestrian on the right is correct. The other estimates are also correct. (e) The overall classification is correct. When zooming into the left of the image, it can be checked that the red return between the yellow returns corresponds to the gap between the leg and the arm of the person; as a result, these inferred labels are correct. (f) From left to right, the vegetation, the cars, the bush, the pedestrians and the fence are classified correctly (apart from one green return on the fence).

8.5. *Convergence Analysis of Inference*

As discussed in Section 3.2, convergence in graphs with cycles is not guaranteed but can be checked experimentally. In this section, the converge of loopy BP is explored. The Boston dataset was used for this last set of experiments. The behavior of loopy BP in a cyclic network was analyzed using a set of 400 manually labeled scans and five-fold cross-validation.

The evaluation is summarized in Figure 14. Inference is performed in each of the networks involved in cross-validation with a varying number of loopy BP iterations. The accuracies provided correspond to the classification of the two classes car and other. The networks used for these tests are those described in Section 7.3.

The left plot of Figure 14 shows that on average loopy BP converges after about five iterations where the accuracy reaches a plateau and is higher than the accuracy obtained with local classification. The right plot of Figure 14 shows that, as expected, the inference time increases linearly with the number of loopy BP iterations.

9. Discussion

This section discusses the effect of the pairwise connections as encoded by the proposed CRF models. After introducing the analysis via a thought experiment, the limitations related to smoothing behaviors are discussed. The benefit of the proposed model is also to allow the type of analysis which is now developed thus providing insights into the modeling of spatio-temporal correlations.

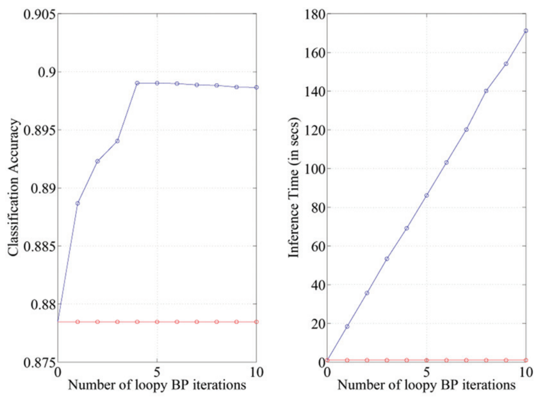


Fig. 14. Empirical analysis of the convergence of loopy BP. On the left, classification accuracies obtained on a car detection problem are plotted as a function of the number of loopy BP iterations. On the right, the corresponding computation times are shown. The red plots refer to local classification. All of the points in the plots are averaged over five-fold cross-validation and correspond to a car detection problem.

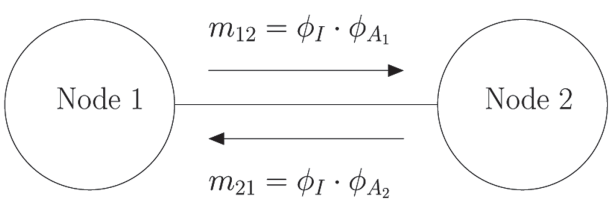


Fig. 15. A simple two-node network. Here ϕ_{A_1} and ϕ_{A_2} are the local potentials on node 1 and 2, respectively, m_{12} and m_{21} are the messages sent across the link during BP, and ϕ_I is the pairwise potential matrix representing the link. The functions ϕ_A and ϕ_I are defined in Equations (2) and (3), respectively.

9.1. A Thought Experiment

Assume that a network contains only two nodes and assume that this network is a classifier so that the states of the nodes belong to a discrete set. The two nodes are linked and this two-node network contains only one link, as shown in Figure 15.

Recall that the pairwise potential (ϕ_I in Figure 15), which quantifies the relationship represented by the link, is learnt as one or several matrices. Assume for now that the model contains only one pairwise matrix. The size of this matrix is $L \times L$, where L is the dimensionality of the state space, here, the number of classes. Performing inference with BP involves multiplying the local potential of each node by the pairwise potential matrix. This is illustrated in Figure 15.

The experiment consists of defining the relationship a link *should* encode given that the true state of each node is known. Two cases are considered: (1) the two nodes have the same state; (2) the two nodes have a different state. In the context of this thought experiment, defining the *true* relationship between two nodes (that is, the relationship a link *should* encode) is equivalent to solving the following system of linear equations:

$$\begin{aligned} \phi_{A_1} &= \phi_{I_L} \times \phi_{A_2}, \\ \phi_{A_2} &= \phi_{I_L} \times \phi_{A_1}, \end{aligned} \tag{19}$$

where the unknowns are the elements of the matrix ϕ_{I_L} (encoding the *true* relationship between the nodes 1 and 2), and the index L refers as above to the size of the state space. The ϕ_A vectors are as in Table 7: they contain only zeros except in the dimension corresponding to the label of the node. Note that solving this system is in general not part of a learning or an inference procedure. However, the mechanism encoded by each of these equations corresponds to the propagation of messages in BP (see Equation (5)), which makes this discussion applicable to system using BP for inference.

The first scenario involves two nodes with the same state. In this case, the true relationship between the two nodes is encoded by an identity matrix. It can be readily verified that the identity matrix satisfies the system of Equations (19).

In the second scenario, the two nodes have a different state. First consider the case in which the size of the state space is two, that is $L = 2$. The pairwise matrix which encodes the true relationship between the two nodes is then

$$\phi_{I_2} = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}. \tag{20}$$

It can be verified that this matrix satisfies Equations (19).

Since the ϕ_{I_L} matrix is symmetric, the number of unknown is $L(L+1)/2$. If all of the diagonal terms take the same value, the number of unknown becomes $L(L+1)/2-(L-1)$. This last term is equal to two when $L = 2$. This means that number of unknowns is two for a system of two equations and confirms that the set of equations describing ϕ_{I_L} can be solved uniquely in this case.

When $L = 3$, the true relationship between the two nodes becomes *dependent* on the state of each node. That is, a different matrix ϕ_{I_3} needs to be used for each pair of labels $\{$ label$_1$, label$_2$ $\}$. Table 7 illustrates this point. In addition, when the pair of labels $\{$ label$_1$, label$_2$ $\}$ is fixed, several ϕ_{I_3} matrices satisfy Equations (19).

When $L \geq 3$, there are more unknowns than equations; the number of elements in ϕ_{I_L} increases with L but there are still only two equations. This means that the system of linear equations, Equations (19), have multiple solutions and several ϕ_{I_L} matrices can be proposed in Table 7.

The outcome of this simple thought experiment is the following: as L becomes larger, a growing set of ϕ_{I_L} matrices is required to encode the true relationship between two nodes. This means that accurately linking network nodes requires a number of relationships to be modeled.

Some of the evaluations presented in the previous sections involved seven classes and can be used here to illustrate the conclusion we have just formulated. A link between two different nodes may represent a transition from

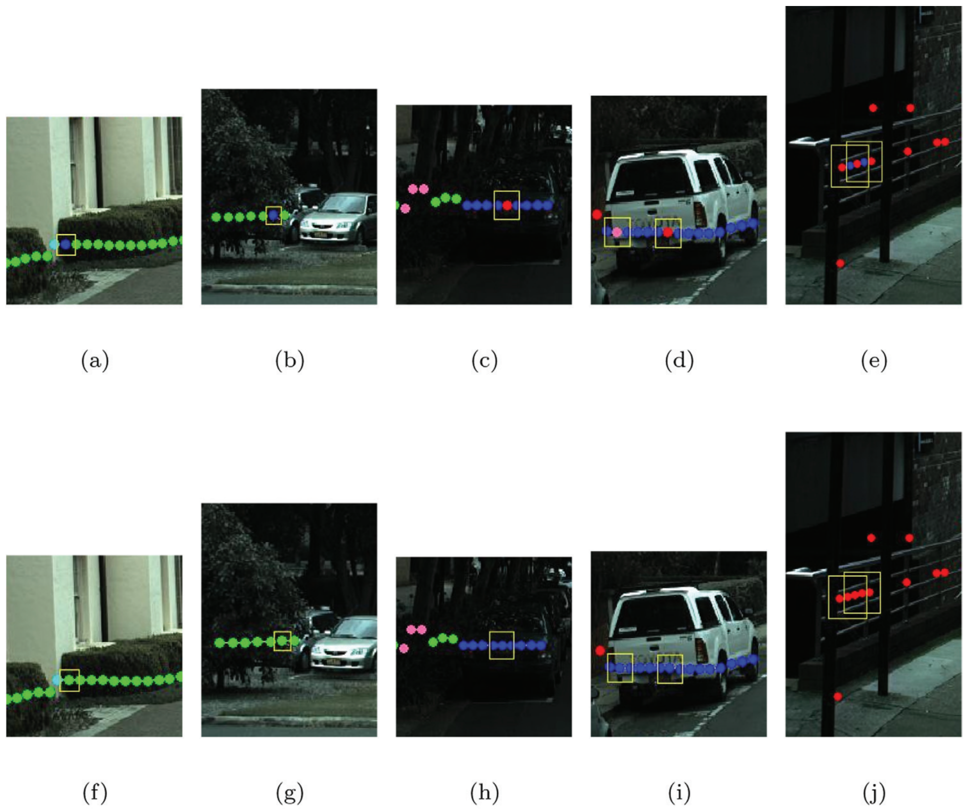


Fig. 16. Illustration of the smoothing effect provided by the one time slice networks tested in Section 7.2. Top row of images: classification before running BP. Bottom row of images: classification after running BP. All of the laser returns falling on the same objects should have the same estimated labels (that is, same color) but some are mis-classified and are indicated in the top row by a yellow rectangle. After performing inference in the various networks, all of the mis-classifications are corrected. The corrected estimates are indicated in the bottom row by a rectangle. BP allows each estimate to take into account the state of its neighbors and leads to improved classification. This display illustrates that a smoothing process is at the core of the correction mechanism provided by joint classification.

Table 7. Examples of possible ϕ_{I_3}. Here label$_1$ and label$_2$ refer to the true label or state of nodes 1 and 2, Respectively; ϕ_{A_1} and ϕ_{A_2} refer to the "ideal" local potential on nodes 1 and 2, "ideal" in the sense that they are non-zero only on the dimension corresponding to the true label. The various ϕ_{I_3} matrices are such that $\phi_{A_1} = \phi_{I_3} \times \phi_{A_2}$ and $\phi_{A_2} = \phi_{I_3} \times \phi_{A_1}$, that is, they verify Equations (19). This table illustrates the fact that when $L \geq 3$, the Value of the pairwise matrix ϕ_{I_L} becomes dependent on the State of the two connected nodes.

label$_1$	label$_2$	ϕ_{A_1}	ϕ_{A_2}	ϕ_{I_3}	
1	2	$\begin{bmatrix}1\\0\\0\end{bmatrix}$	$\begin{bmatrix}0\\1\\0\end{bmatrix}$	$\begin{bmatrix}0&1&0\\1&0&0\\0&0&0\end{bmatrix}$	$\begin{bmatrix}0&1&1\\1&0&0\\1&0&0\end{bmatrix}$
1	3	$\begin{bmatrix}1\\0\\0\end{bmatrix}$	$\begin{bmatrix}0\\0\\1\end{bmatrix}$	$\begin{bmatrix}0&0&1\\0&0&0\\1&0&0\end{bmatrix}$	$\begin{bmatrix}0&0&1\\0&1&0\\1&0&0\end{bmatrix}$
⋮	⋮				

the class car to the class foliage, or from the class trunk to the class person, and so on. An accurate model would need to represent all of these types of links, each one being encoded with one ϕ_{I_7} matrix.

9.2. Pairwise Potentials as Smoothers

The strategy consisting in using a limited number of link types is an alternative to the problem of accurate link instantiation. Pushed to the extreme, this strategy results in using only one type of pairwise potential across the whole network.

In this paper, this problem has been partially avoided through various strategies. In Section 4.1, only laser returns spatially close are linked leading to models representing only one type of similarity link. In Section 4.1, temporal links given by the ICP algorithm are all modeled by the same pairwise potential, again avoiding the link instantiation problem. In Section 8.2, the Delaunay CRF with link selection also represents only similarity relationships.

The pairwise relationships employed in this study effectively behave as smoothers. "Smoothers" can be understood by analogy with interpolation procedures. When a linear interpolation algorithm is run over a grid, the resulting values in the grid are a weighted combination of the values in

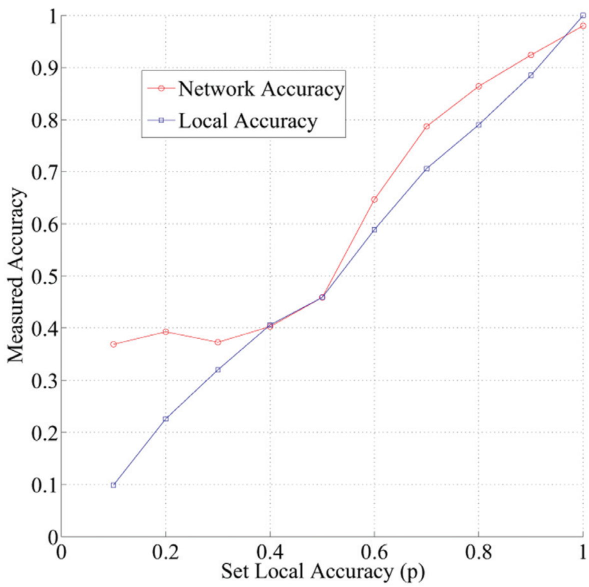


Fig. 17. Simulation of the accuracy gained by modeling pairwise relationships. The simulator uses chain networks such as that displayed in Figure 3.6. Each node has a binary state. An observation o of a node is generated according to a binomial distribution: $P(o = k) \propto p^k(1-p)^{n-k}$, with $n = 2$ since the states are binary. The parameters p corresponds to the x-axis. The blue curve represents the measured local accuracy which is equal to the accuracy of the set of observations $\{o\}$. The red curve is the accuracy obtained after feeding the observations $\{o\}$ to a chain network. For each value of p, the network model is trained with ML on a 1,000 node long network. The true labels of this network are assigned by blocks of variable length to simulate some structure in the chain. The chain network is then tested on a second network of equal length. BP is used to perform inference. This plot shows that above random, that is for $p > 0.5$, the bulk of the accuracy is achieved by local classification since the red curve is above but close to the blue curve. It can also be seen that for a local accuracy of 90%, this simulation predicts a network accuracy of 92%, which matches the results obtained in Sections 8.1 and 8.2.

a local neighborhood. Using network links encoding similarity produces the same smoothing effect. The smoothing effect at one time slice for models used in Section 7.2 is illustrated in Figure 16.

Recognizing the smoothing behavior of spatio-temporal networks helps to understand the benefits of such networks. Figure 17 shows the network accuracy as a function of the local classification accuracy. An interesting behavior can be noticed: the accuracy of the network only slightly improves on local accuracy which shows that the local accuracy drives the network accuracy.

This last remark indicates the fundamental importance of local features. Local features are what allow the classifier to achieve most classification accuracy. This is confirmed by the results presented in Section 8.1: an accuracy of 91% was achieved after local classification; this accuracy was improved by about 2% after running inference in the networks. It is also confirmed by the simulation presented in Figure 17 and the display of Figure 16.

10. Conclusion

A 2D probabilistic representation for multi-class multi-sensor object recognition has been presented. This representation is based on CRFs which are used as a flexible modeling tool to automatically select the relevant features extracted from the various modalities and represent different types of spatial and temporal correlations.

Based on two datasets acquired in two different cities with different sensors, eight different sets of results have been presented. The benefits of modeling spatial and temporal correlations were first evaluated on a car detection problem where an increase in accuracy of up to 5% was measured.

Three different types of networks have been introduced to build semantic maps. These were evaluated on a seven-class classification problem where an accuracy of 91% was achieved. The mapping experiments yielded some insight into the smoothing role of pairwise links. It has been demonstrated how over-smoothing can be partially avoided by creating networks which automatically select the types of links to be used. Computation times were evaluated showing that the larger networks involved in our study are close to being real-time requiring about 11 seconds for inference on a set of 7,500 nodes. Finally, by means of an empirical study, the convergence of the inference algorithm used in cyclic networks has been verified. Convergence is in general observed in about five iterations.

The discussion concluding this publication has described the limitations of the proposed networks in terms of their smoothing effect. The fundamental role of features in achieving high classification accuracies has also been highlighted.

Acknowledgements

The authors would like to thank Albert Huang for providing software and data, and sharing his expertise, and Roman Katz for useful discussions. This work is supported by the ARC Center of Excellence programme, the Australian Research Council (ARC), the New South Wales (NSW) State Government, the University of Sydney Visiting Collaborative Research Fellowship Scheme, and DARPA's ASSIST and CALO Programmes (contract numbers: NBCH-C-05-0137, SRI subcontract 27-000968).

Notes

1. Note that another sensor setup is also used in Section 7.3 involving a 3D range scanner (a Velodyne sensor) and monocular color imagery. The corresponding set of experiments demonstrate the applicability of the proposed framework to 3D data and as a consequence its applicability to the generation of full 3D semantic models.

2. The number of returns acquired by a 2D laser scanner over a 3 km long trajectory is larger than 7,500 (more details are provided in Table 3 later in the article). Here, 7,500 corresponds to the average number of nodes in the testing sets generated for 10-fold cross-validation.
3. This was, in fact, the first evidence pointing us to the analysis developed in Section 9 which emphasizes the predominant role of local features.
4. More evidence leading to the conclusions in Section 9 which emphasize local features as opposed to network connections.
5. In particular, the computation of an additional set of weights was implemented in the Logitboost algorithm. The latter multiply the original Logitboost weight of each sample in such way that each class receives, on the average, the same mass. This did not improve the classification results. The Boosting technique developed by (Leskovec and Taylor 2003) for unbalanced datasets is another alternative.

References

Anguelov, D., Koller, D., Parker, E. and Thrun, S. (2004). Detecting and modeling doors with mobile robots. *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*.

Anguelov, D., Taskar, B., Chatalbashev, V., Koller, D., Gupta, D., Heitz, G. and Ng, A. (2005). Discriminative learning of Markov random fields for segmentation of 3D scan data. *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR)*.

Besag, J. (1975). Statistical analysis of non-lattice data. *The Statistician*, 24: 179–195.

Bishop, C. (2006). *Pattern Recognition and Machine Learning*. Berlin, Springer.

Bosse, M. and Zlot, R. (2008). Map matching and data association for large-scale two-dimensional laser scan-based slam. *The International Journal of Robotics Research*, 27(6): 667–691.

Cormen, T., Leiserson, C., Rivest, R. and Stein, C. (2001). *Introduction to Algorithms*. New York, MIT Press and McGraw-Hill.

De Berg, M., Van Kreveld, M., Overmars, M. and Schwarzkopf, O. (2000). Berlin, Springer.

Douillard, B. (2009). *Vision and Laser Based Classification in Urban Environments*. PhD thesis, School of Aerospace and Mechanical Engineering, The University of Sydney.

Douillard, B., Fox, D. and Ramos, F. (2007a). A spatio-temporal probabilistic model for multi-sensor multi-class object recognition. *Proceedings of the International Symposium of Robotics Research (ISRR)*.

Douillard, B., Fox, D. and Ramos, F. (2008). Laser and vision based outdoor object mapping. *Proceedings of Robotics: Science and Systems*.

Douillard, B., Upcroft, B., Kaupp, T., Ramos, F. and Durrant-Whyte, H. (2007b). Bayesian filtering over compressed appearance states. *Proceedings of the Australasian Conference on Robotics and Automation (ACRA)*.

Duda, O., Hart, P. and Stork, D. (2001). *Pattern Classification*, 2nd edition. New York, Wiley-Interscience.

Durrant-Whyte, H. and Bailey, T. (2006). Simultaneous Localisation and Mapping (SLAM): Part I The essential algorithms. *Robotics and Automation Magazine*, 13(2): 99–110.

Fei-Fei, L. and Perona, P. (2005). A Bayesian hierarchical model for learning natural scene categories. *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR)*.

Felzenszwalb, P., McAllester, D. and Ramanan, D. (2008). A discriminatively trained multiscale deformable part model. *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR)*.

Fisher, R. A. (1936). The use of multiple measurements in taxonomic problems. *Annals of Eugenics*, 7: 179–188.

Freeman, W., Pasztor, E. and Carmichael, O. (2000). Learning low-level vision. *Proceedings of the International Conference on Computer Vision (ICCV)*, Vol. 1, pp. 25–47.

Frey, B. and MacKay, D. (1997). A revolution: belief propagation in graphs with cycles. *Advances in Neural Information Processing Systems (NIPS)*. Cambridge: MIT Press.

Friedman, J., Hastie, T. and Tibshirani, R. (2000). Additive logistic regression: a statistical view of boosting. *The Annals of Statistics*, 28(2): 337–407.

Friedman, S., Fox, D. and Pasula, H. (2007). Voronoi random fields: Extracting the topological structure of indoor environments via place labeling. *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI)*.

Geyer, C. and Thompson, E. (1992). Constrained Monte Carlo maximum likelihood for dependent data. *Journal of the Royal Statistical Society Series B (Methodological)*, 54: 657–699.

Greig, D., Porteous, B. and Seheult, A. (1989). Exact maximum a posteriori estimation for binary images. *Journal of the Royal Statistical Society. Series B (Methodological)*, 51(2): 271–279.

Heitz, G., Elidan, G., Packer, B. and Koller, D. (2009). Shape-based object localization for descriptive classification. *International Journal of Computer Vision*, 84(1): 40–62.

Hoiem, D., Efros, A. and Hebert, M. (2006). Putting objects in perspective. *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR)*.

Hotelling, H. (1933). Analysis of a complex of statistical variables into principal components. *Journal of Educational Psychology*, **24**: 417–441.

Jordan, M. and Weiss, Y. (2002). *Probabilistic Inference in Graphical Models*. Technical Report, EECS Computer Science Division, University of California Berkeley.

Katz, R., Douillard, B., Nieto, J. and Nebot, E. (2008a). A self-supervised architecture for moving obstacles classification. *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*.

Katz, R., Nieto, J. and Nebot, E. (2008b). Probabilistic scheme for laser based motion detection. *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*.

Kaupp, T., Douillard, B., Ramos, F., Makarenko, A. and Upcroft, B. (2007). Shared environment representation for a human–robot team performing information fusion. *Journal of Field Robotics*, 24(11–12): 911–942.

Kumar, S. (2005). *Models for Learning Spatial Interactions in Natural Images for Context-based Classification*. PhD thesis, Robotics Institute, Carnegie Mellon University, Pittsburgh, PA.

Kumar, S., Ramos, F., Douillard, B., Ridley, M. and Durrant-Whyte, H. (2007). A novel visual perception framework. *Proceedings of the International Conference on Control, Automation, Robotics and Vision*.

Kumar, V., Rus, D. and Singh, S. (2004). Robot and sensor networks for first responders. *IEEE Pervasive Computing*, 3(4):24–33.

Lafferty, J., McCallum, A. and Pereira, F. (2001). Conditional random fields: probabilistic models for segmenting and labeling sequence data. *Proceedings of the International Conference on Machine Learning (ICML)*.

Leskovec, J. and Taylor, J. (2003). Linear programming boost for uneven datasets. *Proceedings of the International Conference on Machine Learning (ICML)*.

Liao, L. (2006). *Location-based Activity Recognition*. PhD thesis, University of Washington, Department of Computer Science and Engineering, Seattle, WA.

Liao, L., Choudhury, T., Fox, D. and Kautz, H. (2007). Training conditional random fields using virtual evidence boosting. *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI)*.

Limketkai, B., Liao, L. and Fox, D. (2005). Relational object maps for mobile robots. *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI)*.

Lowe, D. (2004). Discriminative image features from scale-invariant keypoints. *International Journal of Computer Vision*, 60(2): 91–110.

Luber, M., Arras, K., Plagemann, C. and Burgard, W. (2008). Classifying dynamic objects: an unsupervised learning approach. *Proceedings of Robotics: Science and Systems*, Zurich, Switzerland.

Martinez-Mozos, O., Stachniss, C. and Burgard, W. (2005). Supervised learning of places from range data using Adaboost. *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*.

Martinez-Mozos, O., Triebel, R., Jensfelt, P., Rottmann, A. and Burgard, W. (2007). Supervised semantic labeling of places using information extracted from sensor data. *Robotics and Autonomous Systems*, 55(5): 1742–1747.

Monteiro, G., Premebida, C., Peixoto, P. and Nunes, U. (2006). Tracking and classification of dynamic obstacles using laser range finder and vision. *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*.

Murphy, K., Torralba, A. and Freeman, W. (2003). Using the forest to see the trees: a graphical model relating features, objects and scenes. *Advances in Neural Information Processing Systems (NIPS)* Cambridge: MIT Press.

Murphy, K., Weiss, Y. and Jordan, M. (1999). Loopy belief propagation for approximate inference: an empirical study. *Proceedings of the Conference on Uncertainty in Artificial Intelligence (UAI)*.

Ng, A. and Jordan, M. (2002). On discriminative vs. generative classifiers: A comparison of logistic regression and naive bayes. *Advances in Neural Information Processing Systems (NIPS)*. Cambridge: MIT Press.

Pantofaru, C., Unnikrishnan, R., and Hebert, M. (2003). Toward generating labeled maps from color and range data for robot navigation. *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*.

Pearl, J. (1988). *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference*. San Francisco, CA, Morgan Kaufmann Publishers, Inc.

Posner, I., Cummins, M. and Newman, P. (2009). A generative framework for fast urban labeling using spatial and temporal context. *Autonomous Robots*, 26: 153–170.

Posner, I., Cummins, M. and Newman, P. (2008). Fast probabilistic labeling of city maps. *Proceedings of Robotics: Science and Systems*.

Posner, I., Schroeter, D. and Newman, P. M. (2007). Using scene similarity for place labeling. *Proceedings of the International Symposium on Experimental Robotics (ISER)*.

Roweis, S. T. and Saul, L. K. (2000). Nonlinear dimensionality reduction by locally linear embedding. *Science*, 290: 2323–2326.

Schapire, R. and Singer, Y. (1999). Improved boosting algorithms using confidence-rated predictions. *Machine Learning*, 37(3): 297–336.

Schulz, D. (2006). A probabilistic exemplar approach to combine laser and vision for person tracking. *Proceedings of Robotics: Science and Systems*.

Simoncelli, E. and Freeman, W. (1995). The steerable pyramid: a flexible architecture for multi-scale derivative computation. *Proceedings of the International Conference on Image Processing*.

Sudderth, E., Ihler, A., Freeman, W. and Willsky, A. (2003). Nonparametric belief propagation. *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR)*.

Sudderth, E., Torralba, A., Freeman, W. and Willsky, A. (2008). Describing visual scenes using transformed objects and parts. *International Journal of Computer Vision*, 77(1–3): 291–330.

Sutton, C. and McCallum, A. (2006). An introduction to conditional random fields for relational learning. *Introduction to Statistical Relational Learning*, Getoor, L. and Taskar, B. (eds). Cambrdige, MA, MIT Press.

Sutton, C. and McCallum, A. (2007). Piecewise pseudolikelihood for efficient training of conditional random fields. *Proceedings of the International Conference on Machine Learning (ICML)*.

Szummer, M., Kohli, P. and Hoiem, D. (2008). Learning CRFs using graph cuts. *European Conference on Computer Vision (ECCV)*.

Tenenbaum, J., DeSilva, V. and Muller, K. R. (2000). A global geometric framework for nonlinear dimensionality reduction. *Science*, 290: 2319–2323.

Thrun, S., Burgard, W. and Fox, D. (2005). *Probabilistic Robotics*. Cambridge, MA, MIT Press.

Torralba, A., Murphy, K., and Freeman, W. (2004). Contextual models for object detection using boosted random fields. *Advances in Neural Information Processing Systems (NIPS)*. Cambridge: MIT Press

Toyama, K. (2001). Probabilistic tracking in a metric space. *Proceedings of the International Conference on Computer Vision (ICCV)*, 50–59.

Triebel, R., Kersting, K. and Burgard, W. (2006). Robust 3D scan point classification using associative Markov networks. *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*.

Vail, D., Lafferty, J. and Veloso, M. (2007). Feature selection in conditional random fields for activity recognition. *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*.

Vallespi-Gonzalez, T. S. C. (2008). Prior data and kernel conditional random fields for obstacle detection. *Proceedings of Robotics: Science and Systems IV*.

Vapnik, V. (2000). *The Nature of Statistical Learning Theory*. Berlin, Springer.

Vedaldi, A., Favaro, P. and Grisan, E. (2007). Boosting invariance and efficiency in supervised learning. *Proceedings of the International Conference on Computer Vision (ICCV)*.

Viola, P. and Jones, M. (2004). Robust real-time object detection. *International Journal of Computer Vision*, 57(2) 137–154.

Wainwright, M., Jaakkola, T. and Willsky, A. (2005). Map Estimation Via Agreement on (Hyper) Trees: Message-passing and Linear Programming. *IEEE Transactions on Information Theory*, 51: 3697–3717.

Watanabe, Y. and Fukumizu, K. (2009). Graph zeta function in the Bethe free energy and loopy belief propagation. *Advances in Neural Information Processing Systems (NIPS)*, Cambridge: MIT Press.

Williams, S. (2001). *Efficient Solutions to Autonomous Mapping and Navigation Problems*. PhD thesis, University of Sydney, Australian Centre for Field Robotics.

Yedidia, J., Freeman, W., and Weiss, Y. (2002). *Constructing Free Energy Approximations and Generalized Belief Propagation Algorithms*. Technical Report, MERL.

Zhang, Q. and Pless, R. (2004). Extrinsic calibration of a camera and laser range finder (improves camera calibration). *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Sendai, Japan.

Zhang, Z. (1994). Iterative point matching for registration of free-form curves and surfaces. *International Journal of Computer Vision*, 13(2): 119–152.