Onboard Contextual Classification of 3-D Point Clouds with Learned High-order Markov Random Fields

Daniel Munoz, Nicolas Vandapel, and Martial Hebert
The Robotics Institute
Carnegie Mellon University
{dmunoz,vandapel,hebert}@ri.cmu.edu

***Abstract*— Contextual reasoning through graphical models such as Markov Random Fields often show superior performance against local classifiers in many domains. Unfortunately, this performance increase is often at the cost of time consuming, memory intensive learning and slow inference at testing time. Structured prediction for 3-D point cloud classification is one example of such an application. In this paper we present two contributions. First we show how efficient learning of a random field with higher-order cliques can be achieved using subgradient optimization. Second, we present a context approximation using random fields with high-order cliques designed to make this model usable online, onboard a mobile vehicle for environment modeling. We obtained results with the mobile vehicle on a variety of terrains, at 1/3 Hz for a map 25 × 50 meters and a vehicle speed of 1-2 m/s.**

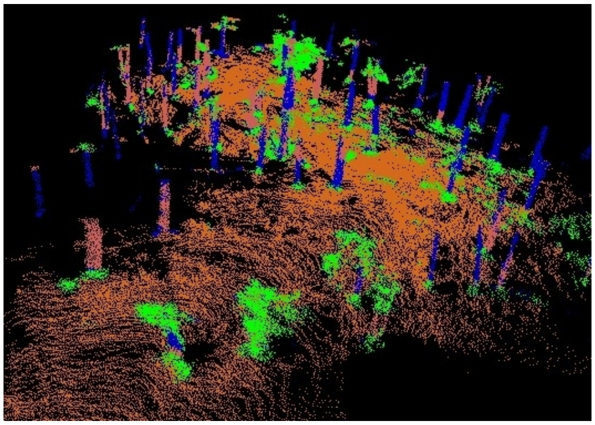

Fig. 1. Example onboard terrain classification result.

I. INTRODUCTION

Accurate perception of environments is critical for autonomous ground vehicles. An improved understanding/classification of the world is beneficial for many applications such as autonomous navigation and environment modeling. However, there exists a trade-off between processing time and the quality of the classification. In this paper, we address both these challenges for the task of contextual 3-D point cloud classification onboard of a moving vehicle. An example classification result from our onboard experiments is presented in Figure 1; the different colors represent interesting labels such as vegetation (green), large (red) and small (blue) tree trunks, and ground (orange).

A common classification approach for this problem is to learn a classifier that assigns a label to each point, independently of its neighbors' assignments. This approach is computationally fast [1] and can produce good results when the extracted features are discriminative. However, when the extracted features are noisy, this approach can produce noisy classifications where points' labels are not locally consistent. Local consistency can be achieved by considering the local context around each point when performing classification. The label of the points are then determined jointly, not independently. The Markov Random Field framework is a popular choice because it models the spatial interactions present in the scene. A Conditional Random Field (CRF) is a popular discriminative model to learn such interactions from training data. In this paper, we are interested in a CRF variant called an Associative Markov Network (AMN). The authors in [2], [3] show that modeling the interactions with AMNs outperform standard Support Vector Machine (SVM) algorithms for the application of 3-D point cloud classification.

While the benefits of modeling the interactions are apparent, they come at a cost. Due to combinatorial issues, these interactions are typically considered at most between pairs of points at a time, that is, a *pairwise* model is used. Thus, in order to model context, for each point it is necessary to consider the interactions with many neighbors during classification. Because of this interdependency, classification is slower than local independent classifiers such as SVMs.

In [4], the authors present a new method that very efficiently models the interactions of a *group* of points instead of normally considering the *many linked pairs* of points within that same group, that is, it models *high-order* interactions. The parameters for this high-order contextual model are determined through cross-validation, a time-consuming task for high-dimensional parameters. The first contribution of this paper is to show how to efficiently *learn* high-dimensional model parameters from very large training datasets using a recently proposed optimization technique [5]. The result of this formulation directly translates into efficiently learning a high-order AMN. Our second contribution analyzes the feasibility and presents a context approximation method to perform contextual classification onboard of a moving vehicle.

This paper is structured into six sections. In the next, various notations are introduced and the background of the original pairwise AMN formulation is reviewed. In Section III, we present our first contribution: how to efficiently learn the model parameters for high-order AMNs. In Section IV, we propose and analyze the performance of our context approximation in the off-board, batch scenario. Section V presents our second contribution where we utilize a high-order AMN and perform classification onboard of a moving robot with our proposed context approximation.

II. PRELIMINARIES

A. Problem

Our classification task can be formalized as follows. Given a set of N random variables $\mathbf{Y} = \{Y_1, \ldots, Y_N\}$, where each variable takes a value $Y_i \in \{l_1, \ldots, l_K\}$, find the assignment of values of $\mathbf{y} = \{y_1, \ldots, y_N\}$ to $\mathbf{Y}$ that maximizes some score function. In the context of 3-D point classification, each random variable represents a 3-D point and its value corresponds to one of K labels; in our experiments we classify {wires, pole/tree trunks, load bearing surfaces, facades, vegetation/scattered points} from data. Formulating the classification task as a supervised learning problem, we want to learn a discriminative model that conditions the joint distribution of labels on the data $\mathbf{x}$ that we can extract from the scene $P_{\mathbf{w}}(\mathbf{y}|\mathbf{x})$, where $\mathbf{w}$ are the model parameters. The classification procedure is then broken into two steps: (1) learning the model parameters given labeled data $(\mathbf{x}, \hat{\mathbf{y}})$ and then (2) inferring the best assignments of a novel scene given its features.

B. Conditional Random Fields

In this section we give a brief overview of the CRF model in general and then proceed in more details about our model in the next section. A CRF is a popular MRF variant used in computer vision that models a conditional distribution $P(\mathbf{y}|\mathbf{x})$ [6]. The distribution is defined by the dependencies of the random variables represented in an undirected graph (V, E) with edges $E = \{(ij)\}|(i < j)$ where each vertex represents a random variable and the edges represent a dependency between two variables. Through the Hammersley-Clifford Theorem [7], the probability distribution is defined over the sets of cliques C in the graph as

$$P_{\mathbf{w}}(\mathbf{y}|\mathbf{x}) = \frac{1}{Z} \prod_{c \in C} \phi_c(\mathbf{y_c}), \tag{1}$$

where $\mathbf{y_c} = \{y_i\}_{i \in c}$, $\phi_c(\cdot)$ is a clique potential that is implicitly a function of the data $\mathbf{x}$ and model parameters $\mathbf{w}$. The potential measures the affinity of the assignment $\mathbf{y_c}$ to the variables of the clique, and Z is the normalizing partition function defined to be $Z = \sum_{\mathbf{y}'} \prod_{c \in C} \phi_c(\mathbf{y}'_{\mathbf{c}})$; note that the computation of Z is exponential.

Because CRFs model the joint distribution, performing inference on a graph with arbitrary clique interactions is NP-hard in general due to the exponential output space of possible solutions. To accomodate this combinatorial issue, most random fields use a pairwise model where only the potentials around each node i and each edge (ij) are considered. Still, in general, inference on arbitrary pairwise potentials on graphs with cylic interactions is also NP-hard [8]. Common approximation inference techniques include graph-cuts [9], [8] and belief propagation (BP) and its variants [10], [11]. We refer to [12], [13] for thorough recent surveys.

Learning the model parameters in CRFs is traditionally done by maximizing the conditional likelihood of Equation 1, with regularization on the model parameters. It can be shown that computing this gradient involves computing the marginal probabilities at each iteration to define the partition function; these marginals are often approximated with the beliefs from BP [14]. It is important to note that this approach does not extend in general to high-order interactions as BP computation is exponential in the size of the largest clique [15]. In our model, we will have hundreds of cliques containing 30 to 60 nodes.

C. Pairwise Associative Markov Networks

In our final model, we use an Associative Markov Network (AMN) as originally described by Taskar et al. in [16]. In the following we describe the pairwise model using the notation from [16]. First, we define $\mathbf{x} = \{\mathbf{x_i}, \mathbf{x_{ij}}\}$ to be the extracted features from the scene where $\mathbf{x_i} \in \mathbb{R}^{d_n}$ and $\mathbf{x_{ij}} \in \mathbb{R}^{d_e}$ are the features that describe node i and the relationship between nodes i and j, respectively and $\mathbf{x} \geq 0$. In our application, $\mathbf{x_i}$ are statistics of the local distribution of points around point i and $\mathbf{x_{ij}}$ measures how well the features from i and j agree. The AMN model uses log-linear potentials to represent the dependence of the potentials on the extracted features. That is, $\log \phi_i(l_k) = \mathbf{w_n^k} \cdot \mathbf{x_i}$ where $y_i = l_k$ (the label value assigned to node i) and $\mathbf{w_n^k} \in \mathbb{R}^{d_n}$ are the weights used when a node is assigned l_k. The potential over an edge models an associative/Pott's behavior that favors the two linked nodes taking on the same labels and penalizes otherwise: $\log \phi_{ij}(l_k, l_k) = \mathbf{w_e^k} \cdot \mathbf{x_{ij}}$, where $\mathbf{w_e^k} \in \mathbb{R}^{d_e}$ are the weights used when linked nodes are both assigned l_k, and $\forall l_k \neq l_o, \log \phi_{ij}(l_k, l_o) = 0$. As will be discussed in the next section, enforcing non-negative edge potentials enables efficient inference. To achieve this, the edge weight vectors are constrained by $\mathbf{w_e^k} \geq \mathbf{0}$. Finally, changing the representation of an assignment $\mathbf{y}$ with a vector of $K \cdot N$ indicator variables where $\mathbf{y} = \{y_i^k, k, i | y_i^k = I(y_i = l_k)\}$, the log of the joint-conditional probability is defined as $\log P_{\mathbf{w}}(\mathbf{y}|\mathbf{x}) =$

$$\sum_{i=1}^{N} \sum_{k=1}^{K} (\mathbf{w_n^k} \cdot \mathbf{x}_i) y_i^k + \sum_{(ij) \in E} \sum_{k=1}^{K} (\mathbf{w_e^k} \cdot \mathbf{x}_{ij}) y_i^k y_j^k - \log Z_{\mathbf{w}}(\mathbf{x}), \tag{2}$$

where $Z_{\mathbf{w}}(\mathbf{x}) = \sum_{\mathbf{y}'} \prod_{i=1}^{N} \phi_i(y'_i) \prod_{ij \in E} \phi_{ij}(y'_i, y'_j)$.

To simplify notation, we define a $K(d_n + d_e)$ length row vector $\mathbf{w} = \{\mathbf{w_n}, \mathbf{w_e}\}$ with $\mathbf{w_n} = \{\mathbf{w_n^1}, \ldots, \mathbf{w_n^K}\}$ and $\mathbf{w_e} = \{\mathbf{w_e^1}, \ldots, \mathbf{w_e^K}\}$. Also we redefine $\mathbf{y}$ to be a $K(N + |E|)$ column vector $\mathbf{y} = \{\mathbf{y_n}, \mathbf{y_e}\}^T$ with $\mathbf{y_n} = \{\ldots, y_i^1, \ldots, y_i^K, \ldots\}$ and $\mathbf{y_e} = \{\ldots, y_{ij}^1, \ldots, y_{ij}^K, \ldots\}$ where $y_{ij}^k = y_i^k \wedge y_j^k$. Finally, we construct $\mathbf{X}$ to be a $K(d_n + d_e) \times K(N + |E|)$ matrix such that $\log P_{\mathbf{w}}(\mathbf{y}|\mathbf{x}) = \mathbf{wXy} - \log Z_{\mathbf{w}}(\mathbf{x})$. This matrix contains the features repeated multiple times in the columns and padded with zeros appropriately.

In order to solve the inference task $\mathbf{y}^* = \arg\max_{\mathbf{y}} P_{\mathbf{w}}(\mathbf{y}|\mathbf{x})$, the authors in [16] formulate a linear program (LP) over a non-integral (and constrained) version of $\mathbf{y}$ and setting $\mathbf{wXy}$ as the objective function. Finding the optimal parameters $\mathbf{w}$ is formulated as a max-margin learning problem. Given labeled data $(\mathbf{x}, \hat{\mathbf{y}})$, the goal is to find the weights that maximize the margin of confidence in $P_{\mathbf{w}}(\hat{\mathbf{y}}|\mathbf{x})$ versus $P_{\mathbf{w}}(\mathbf{y}|\mathbf{x}) \quad \forall \mathbf{y} \neq \hat{\mathbf{y}}$. This learning problem is formulated as the following convex program:

$$\begin{aligned} \min_{\mathbf{w},\xi} \quad & \tfrac{\lambda}{2}\|\mathbf{w}\|^2 + \xi \\ \text{s.t} \quad & \mathbf{wX}\hat{\mathbf{y}} + \xi \geq \max_{\mathbf{y}} \ \mathbf{wXy} + \mathscr{L}(\mathbf{y}, \hat{\mathbf{y}}), \end{aligned} \tag{3}$$

where ξ is a slack variable that represents the gap in the total energy between the optimal and achieved solutions and $\mathscr{L}(\mathbf{y}, \hat{\mathbf{y}})$ is a loss function that acts as a margin term for structured output spaces. As in [16] and [2], we use the Hamming distance between the true and achieved assignments for our loss function. In [16], the authors show how to substitute the dual of the inference LP to bound the non-linear constraint which then results in a valid quadratic program (QP) and can then be solved by optimization software. Note that one important property of this formulation is that estimating the intractable partition function Z has been successfully omitted from the derivation.

In the next section, the model will be extended to efficiently learn higher order interactions beyond the pairwise case.

III. LEARNING HIGH-ORDER ASSOCIATIVE MARKOV NETWORKS

A. Alternative Optimization Approaches

In this section we will show how to extend the AMN model to efficiently learn high-order interactions. In contrast to learning with a QP solver, the technique uses a gradient-based method from [5] that requires performing inference at each step. Here we review alternative algorithms for inference and learning that we use to learn the high-order AMN model.

Inference Instead of using an LP for inference, the authors of [16] mention that inference on the pairwise model can also be performed through graph-cut inference, i.e. finding the st-mincut of a specially constructed graph. However, note that the original QP learning must still rely on an LP formulation of inference. We briefly review the graph-cut approach as it is essential for incorporating the high-order model. Performing inference on our random field can then be thought of as minimizing its energy, that is, the negative of the clique potentials:

$$E(\mathbf{y}) = \sum_{i=1}^{N} E_i(y_i) + \sum_{(ij)\in E} E_{ij}(y_i, y_j), \tag{4}$$

with $E_i(y_i) = -\log \phi_i(y_i)$ and $E_{ij}(y_i, y_j) = -\log \phi_{ij}(y_i, y_j)$. In [8], the authors state that pairwise submodular energy functions of binary variables ($K = 2$) are *graph-representable*, that is, a directed graph can be constructed such that the st-mincut is the function's minimum value and the vertices left connected to the source and terminal define the respective binary variable values. Note that a function of one binary variable is always submodular, a second-order function of binary variables is submodular iff $E_{ij}(0,0) + E_{ij}(1,1) \leq E_{ij}(0,1) + E_{ij}(1,0)$, and the sum of two submodular functions is submodular. If $K = 2$, it is clear that the pairwise AMN energy is submodular and thus can be minimized by finding the st-mincut; we refer to [8] for how to construct the graph. If $K > 2$, the function is not submodular but can be approximated up to a factor of 2 with the α-expansion algorithm; we refer to [9] for more details. This formulation is practically appealing as it does not require optimization software and is memory-efficient for large random fields.

Learning In [5], the authors use the subgradient method to optimize objective

$$\min_{\mathbf{w}} \frac{\lambda\|\mathbf{w}\|^2}{2} + \max_{\mathbf{y}}(\mathbf{wXy} + \mathscr{L}(\mathbf{y}, \hat{\mathbf{y}})) - \mathbf{wX}\hat{\mathbf{y}}, \tag{5}$$

which is the unconstrained version of Equation 3. Since Equation 5 is convex in $\mathbf{w}$, a solution can be achieved through subgradient descent. The key to compute the subgradient of Equation 5 is to use the property: if $f(a,b)$ is differentiable in a, then $\nabla_a f(a, b^*)$ is a subgradient of the convex function $\max_b f(a,b)$ for $b^* \in \arg\max_b f(a,b)$. Therefore, a subgradient $\mathbf{g_w}$ of Equation 5 is

$$\mathbf{g_w} = \lambda\mathbf{w} + \mathbf{Xy}^* - \mathbf{X}\hat{\mathbf{y}}. \tag{6}$$

Solving the loss-augmented inference problem $\mathbf{y}^* = \max_{\mathbf{y}}(\mathbf{wXy} + \mathscr{L}(\mathbf{y}, \hat{\mathbf{y}}))$ can still be solved with graph-cuts as long as the loss function $\mathscr{L}(\mathbf{y}, \hat{\mathbf{y}})$ does not affect submodularity. To ensure this, the Hamming loss function is typically chosen as it decomposes over the node potentials, which does not affect submodularity. Starting at time $t = 0$ with $\mathbf{w}_0 = \mathbf{0}$, the solution is then achieved through descent until convergence, or T iterations, using the update rule:

$$\mathbf{w}_{t+1} \leftarrow \mathscr{P}_{\mathscr{W}}[\mathbf{w}_t - \alpha \mathbf{g}_{\mathbf{w_t}}], \tag{7}$$

where $\mathscr{P}_{\mathscr{W}}$ projects $\mathbf{w}$ onto a convex set $\mathscr{W}$ formed by any specific convex constraints on $\mathbf{w}$; for AMNs, this projection enforces any negative $\mathbf{w_e}$ to become 0. A common step-size is $\alpha = \frac{\gamma}{\sqrt{t}}$, for some positive γ and current iteration t.

B. P^n Potts Model

In order to model high-order interactions, we need to be able to efficiently solve the model. Kohli et al. recently proposed a class of energy functions that can be efficiently minimized, called the P^n Pott's model [4]. This model extends the general idea of the Pott's model with cliques of arbitrary order n. That is, the model is associative and favors all variables in the clique taking on the same label and penalizes otherwise:

$$E_c(\mathbf{y_c}) = \begin{cases} \lambda_c^k & \text{if } \forall i \in c, y_i = l_k \\ \lambda_{\max} & \text{otherwise,} \end{cases} \tag{8}$$

where $\lambda_{\max} \geq \lambda_c^k, \forall l_k \in \{l_1, \ldots, l_K\}$. The authors show how to construct a graph such that its st-mincut represents the minimized energy for $K = 2$ (the energy is submodular), and for $K > 2$ minimizes the optimal move for the α-expansion algorithm. Hence this potential is graph-representable.

C. Learning The New Model

In the AMN model, we would then like to incorporate the high-order information with the pairwise model, that is our energy function is the sum of Equations 4 and 8:

$$E(\mathbf{y}) = \sum_{i=1}^{N} E_i(y_i) + \sum_{(ij)\in E} E_{ij}(y_i, y_j) + \sum_{c\in S} E_c(\mathbf{y_c}), \tag{9}$$

where S is the set of high-order cliques we would like to consider; we will discuss their construction in the next section. We can minimize (perform inference on) Equation 9 through the Additivity Theorem [8]: minimizing the sum of two graph-representable functions is the same as finding the st-mincut of the two merged graph representations.

Modeling the high-order energy terms in the AMN log-linear model, we define the clique potentials $\forall c \in S, E_c(\mathbf{y_c}) = -\log \phi_c(\mathbf{y_c})$ where

$$\log \phi_c(\mathbf{y_c}) = \begin{cases} \mathbf{w_c^k} \cdot \mathbf{x_c} & \text{if } \forall i \in c, y_i = l_k \\ 0 & \text{otherwise,} \end{cases} \tag{10}$$

and $\mathbf{w_c^k}$ are the weights used when all the variables in clique c obtain value l_k and $\mathbf{x_c} \in \mathbb{R}^{d_c}$ are the features that describe clique c. Now, we define $\mathbf{y} = \{\mathbf{y_n}, \mathbf{y_e}, \mathbf{y_c}\}^T$ where $\mathbf{y_c} = \{\dots, y_c^1, \dots, y_c^K, \dots\}$ is a $|S| \cdot K$ indicator vector with $y_c^k = \prod_{i\in c} y_i^k$. In addition, we define a $K \cdot d_c$ weight vector $\mathbf{w_c} = \{\mathbf{w_c^1}, \dots, \mathbf{w_c^K}\}$ and $\mathbf{w} = \{\mathbf{w_n}, \mathbf{w_e}, \mathbf{w_c}\}$. As before, adjust $\mathbf{X}$ such that $\mathbf{wXy} =$

$$\sum_{i=1}^{N}\sum_{k=1}^{K}(\mathbf{w_n^k} \cdot \mathbf{x}_i) y_i^k + \sum_{(ij)\in E}\sum_{k=1}^{K}(\mathbf{w_e^k} \cdot \mathbf{x}_{ij}) y_{ij}^k + \sum_{c\in S}\sum_{k=1}^{K}(\mathbf{w_c^k} \cdot \mathbf{x_c}) y_c^k. \tag{11}$$

Thus, in contrast to cross-validation for parameter selection [4], we can now learn models with high-order potentials that are log-linear in $\mathbf{w}$ using the subgradient method and graph-cut inference as done before. Note that to satisfy the P^n Pott's model (submodularity constraints), we must have $\lambda_c^k = -\mathbf{w_c} \cdot \mathbf{x_c} \leq \lambda_{\max} = 0$. As done with the edge weights, we project negative high-order clique weights to 0 at each iteration.

D. Discussion

We note that the high-order version of the model was originally formulated in [16]. However, the difference in this work is a feasibility issue. As stated above, because the learning step was formulated as QP, an optimization solver such as CPLEX must be used. The authors in [5] state that a training set of approximately $30,000$ points neared the upper bound of what could be handled by the solver due to memory constraints. Memory constraints with the subgradient method only require efficient inference such as graph-cuts. In our experiments in the next two sections, we will be using training sets with an increase of size by 30% and 64%, respectively. An overview of the learning algorithm is presented in Algorithm 1

Algorithm 1 Learning algorithm overview

```
Inputs: Cliques $C$, ground truth labeling $\widehat{\mathbf{y}}$, regularization term $\lambda$, step-size $\alpha_t$, number of iterations $T$
Compute features for cliques in $C$: $\mathbf{X} = \{\{\mathbf{x_i}\}, \{\mathbf{x_{ij}}\}, \{\mathbf{x_c}\}\}$
$\mathbf{w} \leftarrow 0$
for $T$ iterations do
    $\mathbf{y}^* \leftarrow$ graphcut_inference($\mathbf{X}, \mathbf{w}$) (Eq. 9)
    $\mathbf{w} \leftarrow$ subgradient_update($\mathbf{X}, \widehat{\mathbf{y}}, \mathbf{y}^*, \lambda, \alpha_t$) (Eq. 7)
end for
return w
```

IV. EXPERIMENTAL RESULTS

In this section we first describe how our random fields are constructed and the features used in our experiments. We then propose and analyze the context approximation method which we use in our onboard experiments described in the next section.

A. Clique Construction

For our edge potentials, we model local, spatial interactions by iterating over the nodes (3-D points) and linking each node to its closest b-nearest neighbors (b-NN). In Section IV-D, we evaluate graphs using 3-NN and 5-NN.

In images, the lattice structure of the domain allows for a natural definition of high order cliques; however, an analogy with 3-D point clouds is unclear due to varying point density and lack of an intrinsic data structure. Drawing inspiration from [17], we define a clique in a 3-D point cloud as a set of locally similar points resulting from clustering over the nodes' features and locations. We use simple k-means clustering where a ratio $k_{ratio} = 0.0263$ of clusters versus nodes with the training data was found experimentally to capture the same local geometric structure. During testing $k_{\text{test}} = k_{ratio} \cdot N_{\text{test}}$. An example of the clustering on test data is illustrated in Figure 2.

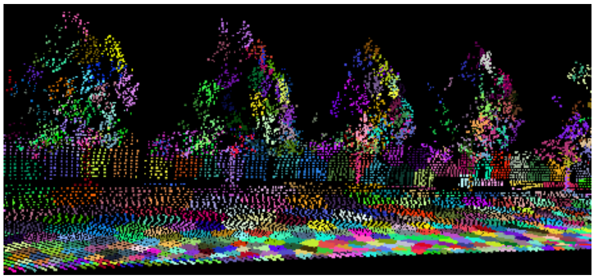

Fig. 2. Clusters on a scene to be used as the high-order cliques.

B. Features

We implemented three geometric features commonly used in spectral analysis of point clouds [18]. We define $\lambda_2 \geq \lambda_1 \geq \lambda_0$ to be the eigenvalues of the scatter matrix M defined over a local neighborhood $\mathcal{N}_p$ around point p. These features capture the {point, surface, linear}-"ness" of the local geometry: $\{\sigma_p = \lambda_0, \sigma_s = \lambda_1 - \lambda_0, \sigma_l = \lambda_2 - \lambda_1\}$, respectively. We refer to these as the spectral features. Next, we estimate the local tangent $\vec{v}_t$ and normal $\vec{v}_n$ vectors for each point by using the principal and smallest eigenvectors of M, respectively. We then compute the cosine and sine of the angles formed between the directions of $\vec{v}_t$ and $\vec{v}_n$ against the vertical and horizontal plane, resulting in four values. We estimate a confidence in the features by scaling the values based on the strength of the extracted directions: $\{\vec{v}_t, \vec{v}_n\}$ by $\{\sigma_l, \sigma_s\}/\max(\sigma_l, \sigma_p, \sigma_s)$, respectively. We will refer to these scaled values as the directional features.

In our experiments, the spectral features are defined using a neighborhood of points within a radius of 0.6 m; the concatenation of the spectral and directional features define the node features $\mathbf{x}_i$. We define an affinity feature between two scalars values f_1 and f_2 as $1/(1+|f_1-f_2|)$. Then, the edge features $\mathbf{x}_{ij}$ are defined to be the concatenation of the two nodes' spectral features and the affinities between the directional features. For the high-order clique features $\mathbf{x}_c$, we computed the same types of features as for the nodes except that the support volume to calculate the scatter matrix is the locations of the nodes that constitute the clique/segment. All of our feature vectors also contain a bias feature of 1.0.

C. *Implementation details*

We used publicly available implementations of k-means clustering [19] and maxflow-mincut [20] performing inference. With the k-means library, we used all the default parameters for Lloyd's algorithm except reduced the number of stages in half in order to reduce computation time. During testing, we do not create nodes or cliques if their features cannot be computed; this will happen if the support volume neighborhood contains less than four points. For α-expansion inference during the field test, at each time step we initialize the labelings from the previous iteration as the majority of the nodes are not deleted from the random field.

D. *Context Approximation Analysis*

In order to perform onboard classification with the AMN model, we need to be able to efficiently 1) construct the necessary cliques in the graph and 2) perform inference. We will now empirically analyze the computational costs of these two tasks for our problem. From our previous work [3], we found that while the pairwise model significantly performs better than SVMs, the large amount of time taken for inference makes the approach unattractive for onboard processing. The expensive inference time can mainly be attributed to the cyclic and long range dependencies resulting from a highly linked random field. Such random fields are needed in order to propagate information in the pairwise model.

Because AMN potentials enforce the nodes in a clique to agree with the same label, it may not be necessary to have a random field with complicated pairwise dependencies if we can model a node's surrounding context in a different manner. Instead, we propose an approximation to achieve contextual information that tries to enforce local *regions* to keep the same label while utilizing node-level information. Instead of creating a random field with many dependencies, we construct a random field with disjoint high-order cliques with the goal that the cliques themselves provide sufficient context. This formulation models context beyond the point-wise level, but does not introduce long range dependencies. This is a counter-intuitive approximation as we have effectively separated the cliques from each other and are now dealing with a seemingly harder problem of modeling high-order interactions instead of simpler pairwise interactions. However, since there are no long-range dependencies, the constructed directed graph used for graph-cut inference is simpler and, as the following will show, optimizing high-order potentials is efficient and still maintains contextual classification. Furthermore, note that this is not the same as performing clique-wise classification as the node potentials are also used.

We trained three models: using nodes and high-order cliques resulting from clustering (HOC), using nodes and edges constructed with 3-NN (3-NN), and using nodes and edges constructed with 5-NN (5-NN). Our training set consisted of 39,188 data points with 5 labels: {*vegetation, wire, pole/trunk, load bearing, and facade*}. The training parameters for HOC were found to be step size $\gamma = 0.1$ and regularization term $\lambda = 0.1$. The training parameters for both 3-NN and 5-NN were $T = 700$, $\gamma = 0.007$, and $\lambda = 0.1$. We performed quantitative evaluation on a testing set of 1.2 million labeled points; note that this is separate data from the field test experiments in Section V. The classification performances are given in Table I, where rows are the ground truth labels and columns are the predicted labels. The 5-NN model performs the best overall with better or about equal precision and recall rates for four of the five classes. The behavior of more true-positives with HOC on the physically smaller *pole/trunk* and *wire* classes can be attributed to the smoothing effect that pairwise models commonly exhibit [3]. Although most accurate, the timing information in Table II illustrates that 5-NN is not practically feasible in an onboard setting. With our context approximation (HOC), we observe comparable classification with most efficient computation.

	HOC	3-NN	5-NN
Classification	3.643	16.123	35.277
Clique construction	11.055	5.843	9.036
Total	14.698	21.966	44.313

TABLE II

AVERAGE COMPUTATION TIMINGS (S) PER 100,000 POINTS FOR A DATASET OF 1.2 MILLION POINTS.

V. FIELD TESTS

This section presents results from the context approximate HOC model tested onboard the Demo-III XUV. The task considered is environment modeling and scene interpretation, that is, we are interested in autonomously creating accurate, detailed representations of unknown environments. Such a representation is potentially useful for other applications such as autonomous navigation.

Confusion Matrix with High-order Cliques Only (HOC). Overall classification rate: 0.871.

	vegetation	wire	pole/trunk	load bearing	facade	Recall
vegetation	198690	7396	7469	562	11810	0.879
wire	113	1293	12	1	71	0.868
pole/trunk	1840	111	4350	8	435	0.645
load bearing	8665	89265	2388	806039	17430	0.872
facade	4284	3057	3410	3	56084	0.839
Precision	0.930	0.013	0.247	0.999	0.653	

Confusion Matrix with Pairwise model (3-NN). Overall classification rate: 0.884.

	vegetation	wire	pole/trunk	load bearing	facade	Recall
vegetation	211234	4011	6168	29	4485	0.935
wire	176	1242	44	0	28	0.834
pole/trunk	3120	96	3196	21	311	0.474
load bearing	10274	56751	23085	807086	26591	0.874
facade	2662	1814	2650	0	59712	0.893
Precision	0.929	0.019	0.091	1.000	0.655	

Confusion Matrix with Pairwise model (5-NN). Overall classification rate: 0.889.

	vegetation	wire	pole/trunk	load bearing	facade	Recall
vegetation	209007	4760	6385	127	5648	0.925
wire	120	1286	84	0	0	0.863
pole/trunk	2945	589	2861	67	282	0.424
load bearing	2931	78873	8126	812278	21579	0.879
facade	1018	1231	1510	0	63079	0.944
Precision	0.968	0.015	0.151	1.000	0.696	

TABLE I

CONFUSION MATRIX WITH HIGH-ORDER CLIQUES ONLY (HOC), PAIRWISE MODEL (3-NN), AND PAIRWISE MODEL (5-NN)

The three main challenges we faced in using a random field onboard of a mobile robot for on-line data processing are: 1) maintaining a coherent graph structure, 2) avoiding data accumulation as more laser data are collected and the vehicle is moving, and 3) efficient feature computations. We address those problems by leveraging on previous work for efficient data accumulation and features computation, and we develop an insertion-deletion scheme for the random field.

First the laser data is accumulated into a dense 3-D scrolling map discretized into ten centimeters edge voxels. At data insertion, the spectral features sufficient statistics are stored and the features are computed efficiently for the new or updated voxels following [1]. Figure 3 illustrates the following process. The newly inserted points are clustered. The resulting clusters are treated as the high-order cliques and are added to the accumulated random field. As the vehicle moves, nodes and cliques outside an area of interest are deleted. The cliques' associated nodes are deleted when its centroid location is outside the area of interest. By construction we ensure node assignment to a unique clique[1]. An illustration of the evolution of the number of nodes and cliques within the accumulated random field is presented in Figure 5. The results were produced on the trail illustrated in Figure 6-center.

Code integration was done on an off-the-shelf computer with a 3 GHz CPU and 2 GB of memory. The Demo-III eXperimental Unmanned Vehicle (XUV) [22] used for field testing is illustrated in Figure 4. The vehicle is equipped with state of the art pose estimation and 3-D laser sensing. Data from a subsection of the environment was previously collected and hand labeled; the training set consisted of 49,110 points with the same types of labels. A HOC model

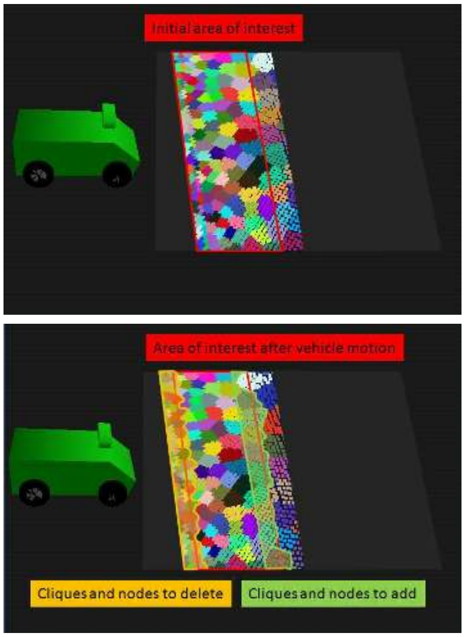
Initial area of interest
Area of interest after vehicle motion
Cliques and nodes to delete
Cliques and nodes to add

Fig. 3. Illustration of the insertion and deletion procedure for nodes and cliques. Top: vehicle starting position with some cliques within the area of interest, represented in red. Bottom: the vehicle moved some distance, the area of interest (in red) overlaps completely and partially with some cliques; the cliques to delete are shaded in orange; the cliques to add are shaded in green; deletion and addition of cliques are performed only if the centroid is within the area of interest.

[1] Since the time of submission, we have performed additional field-test experiments where we investigate the efficiency of non-disjoint cliques (by using overlapping segmentations and linking nodes between adjacent cliques as they are inserted) and compare with the pairwise model [21].

was trained with parameters step size $\gamma = 0.001$ and regularization term $\lambda = 0.1$.

We tested the code onboard the vehicle over a variety of terrains including trails surrounded by dense vegetation (Figure 6-center), a forest with large tree trunks, understory vegetation and canopy (Figure 6-right), and also a mock-up urban terrain with paved roads, utility poles, grass, and buildings (Figure 6-left). The code was tested over almost 10 km. Illustration of the representative terrains, classification results and computation times are presented in Figure 6. The computation times include timings for node feature computation (nodes), cliques construction and features computation (cliques), insertion (merge) and deletion (delete) of nodes and cliques as the vehicle is moving, and the classification time (classification). We were able to obtain classification results at least every three seconds in the most stringent conditions tested: using a large area of interest (25×50 m), in a highly cluttered environment, and with large pan-tilt laser turret motion ($\pm 10\,\text{deg} \times \pm 20\,\text{deg}$). The vehicle speed was between one and two meters per second.

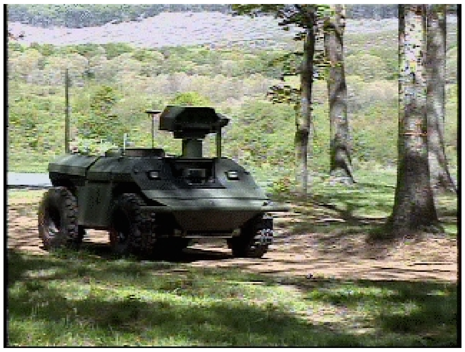

Fig. 4. Demo-III eXperimental Unmanned Vehicle (XUV)

VI. CONCLUSION

In this paper we presented a contribution to the problem of 3-D point cloud classification onboard a mobile vehicle using a conditional random field for scene interpretation and environment modeling. Our first contribution showed how higher-order interactions can be efficiently learned using the subgradient method. The second contribution analyzed, implemented, and tested a context approximation method for classifying streaming data from a 3-D laser range finder. Experiments onboard a mobile robot were conducted in several types of environments (urban, forest, trail) to support the claims.

VII. ACKNOWLEDGMENTS

Prepared through collaborative participation in the Robotics consortium sponsored by the U.S Army Research Laboratory under the Collaborative Technology Alliance Program, Cooperative Agreement DAAD19-01-209912. The authors would like to thank General Dynamics Robotic Systems for its support.

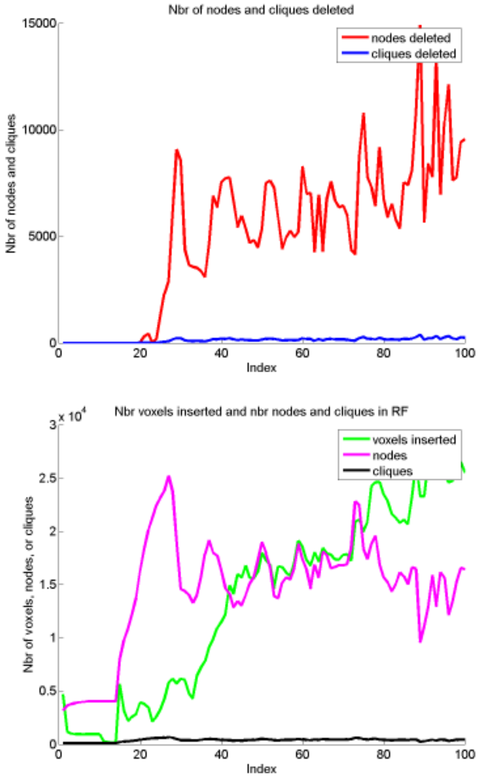

Fig. 5. Evolution of the number of nodes and cliques deleted (top) and present in the accumulated random field (bottom) while driving on a trail. As expected, the number of nodes and cliques increases first as laser data is accumulated, then levels off while the vehicle is moving as nodes and cliques are inserted but also deleted. More and more nodes are deleted because the surrounding of the trail is becoming increasingly cluttered with vegetation.

REFERENCES

[1] J.-F. Lalonde, N. Vandapel, and M. Hebert, "Data structures for efficient dynamic processing in 3-d," *The International Journal of Robotics Research*, vol. 26, no. 8, pp. 777–796, August 2007.

[2] D. Anguelov, B. Taskar, V. Chatalbashev, D. Koller, D. Gupta, G. Heitz, and A. Ng, "Discriminative learning of markov random fields for segmentation of 3-d scan data," in *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2005.

[3] D. Munoz, N. Vandapel, and M. Hebert, "Directional associative markov network for 3-d point cloud classification," in *Fourth International Symposium on 3D Data Processing, Visualization and Transmission*, 2008.

[4] P. Kohli, M. Kumar, and P. Torr, "P3 and beyond: Solving energies with higher order cliques," in *IEEE International Conference on Computer Vision and Pattern Recognition*, 2007.

[5] N. Ratliff, J. Bagnell, and M. Zinkevich, "Online subgradient methods for structured prediction," in *Eleventh International Conference on Artificial Intelligence and Statistics*, 2007.

[6] J. Lafferty, A. McCallum, and F. Pereira, "Conditional random fields: Probabilistic models for segmenting and labeling sequence data," in *International Conference on Machine Learning*, 2001.

[7] S. Li, *Markov Random Field Modeling in Image Analysis*. Springer-Verlag Telos, 2001.

[8] V. Kolmogorov and R. Zabih, "What energy functions can be minimized via graph cuts?" *PAMI*, 2004.

[9] Y. Boykov, O. Veksler, and R. Zabih, "Fast approximate energy minimization via graph cuts," *PAMI*, 2001.

[10] J. Yedidia, W. Freeman, and Y. Weiss, "Generalized belief propagation," in *NIPS*, 2000.

[11] V. Kolmogorov, "Convergent tree-reweighted message passing for energy minimization," *PAMI*, 2006.

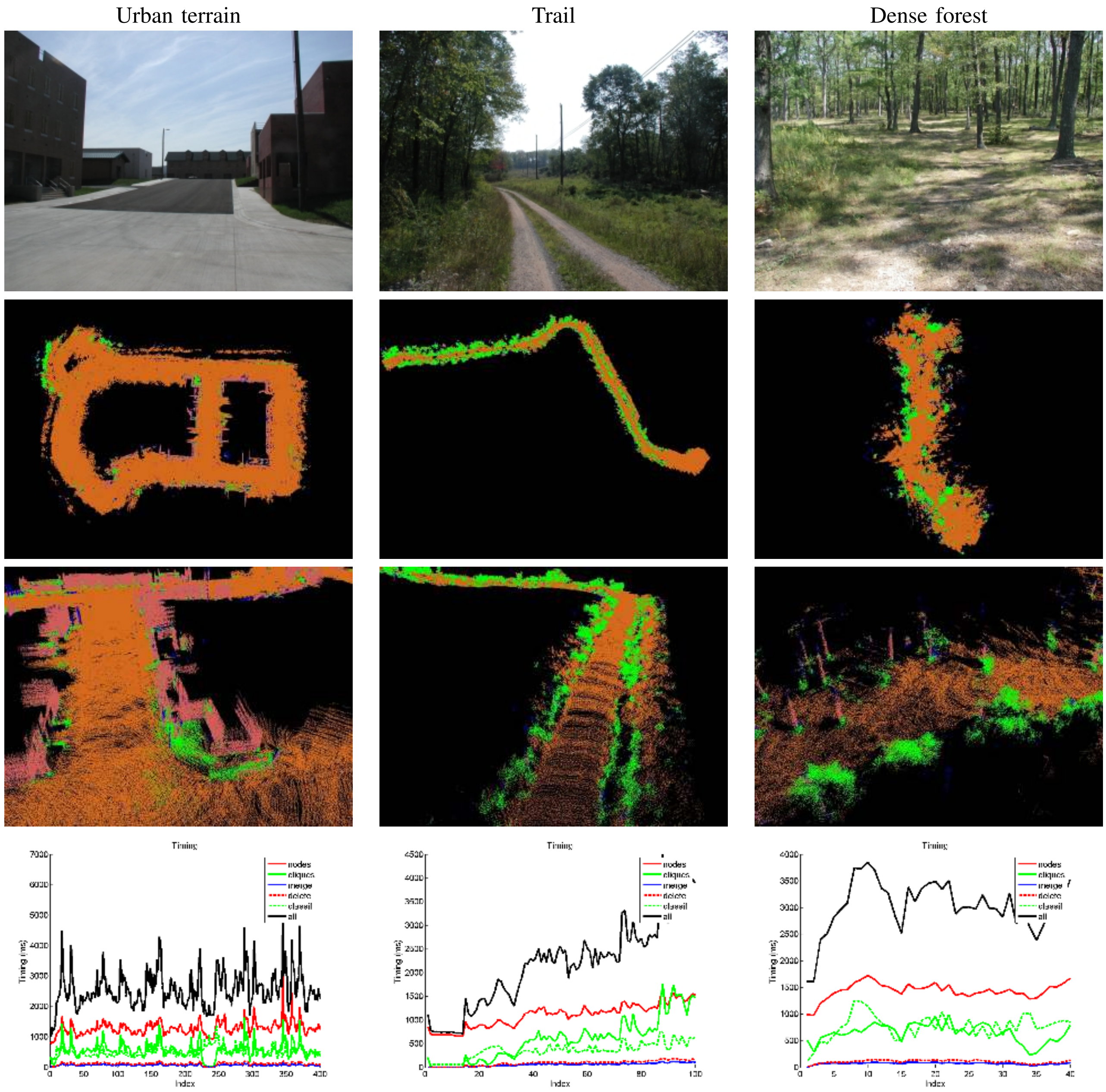

Fig. 6. Onboard data processing results for an **urban environment** (left column), a **trail** (center column), and a **dense forest** (right column). From top to bottom: a representative image of part of the terrain, classification results for the complete scene, a close up view, and the timings. The **urban** terrain covers an area of 215 × 165 m and the vehicle drove over 1.7 km, the **trail** is 360 m in length, and the vehicle path is the **forest** is 93 m long. The processing area covers an area 50 meters wide and 25 meters deep in front of the vehicle. The vehicle operates at speeds between one and two meters per second. Color code: foliage/grass (green), large tree trunks/facade (dark red), small tree trunks (blue), ground (orange).

[12] N. Komodakis, P. Torr, V. Kolmogorov, and Y. Boykov, "Discrete optimization methods in computer vision," Tutorial at International Conference on Computer Vision (ICCV), October 14-15, 2007.

[13] R. Szeliski, R. Zabih, D. Scharstein, O. Veksler, V. Kolmogorov, A. Agarwala, M. Tappena, and C. Rother, "A comparative study of energy minimization methods for markov random fields," *PAMI*, 2007.

[14] C. Sutton and A. McCallum, *Introduction to Statistical Relational Learning*, L. Getoor and B. Taskar, Eds. MIT press, 2006.

[15] C. Bishop, *Pattern Recognition and Machine Learning*. Springer, 2006.

[16] B. Taskar, V. Chatalbashev, and D. Koller, "Learning associative markov networks," in *ICML*, 2004.

[17] A. Willis, J. Speicher, and D.Cooper, "Surface sculpting with stochastic deformable 3d surfaces," in *ICPR*, 2004.

[18] G. Medioni, M.-S. Lee, and C.-K. Tang, *A Computational Framework for Segmentation and Grouping*. Elsevier, 2000.

[19] T. Kanungo, D. Mount, N. Netanyahu, C. Piatko, R. Silverman, and A. Wu, "An efficient k-means clustering algorithm: Analysis and implementation," *PAMI*, 2002.

[20] Y. Boykov and V. Kolmogorov, "An experimental comparison of min-cut/max- flow algorithms for energy minimization in vision," *PAMI*, 2004.

[21] D. Munoz, N. Vandapel, and M. Hebert, *Onboard 3-D Point Cloud Classification with Associative Markov Networks*. manuscript in preparation, 2009.

[22] J. A. Bornstein and C. Shoemaker, "Army ground robotics research program," *Proceedings of the SPIE - The International Society for Optical Engineering*, vol. 5083, no. 1, pp. 303 – 310, 2003.