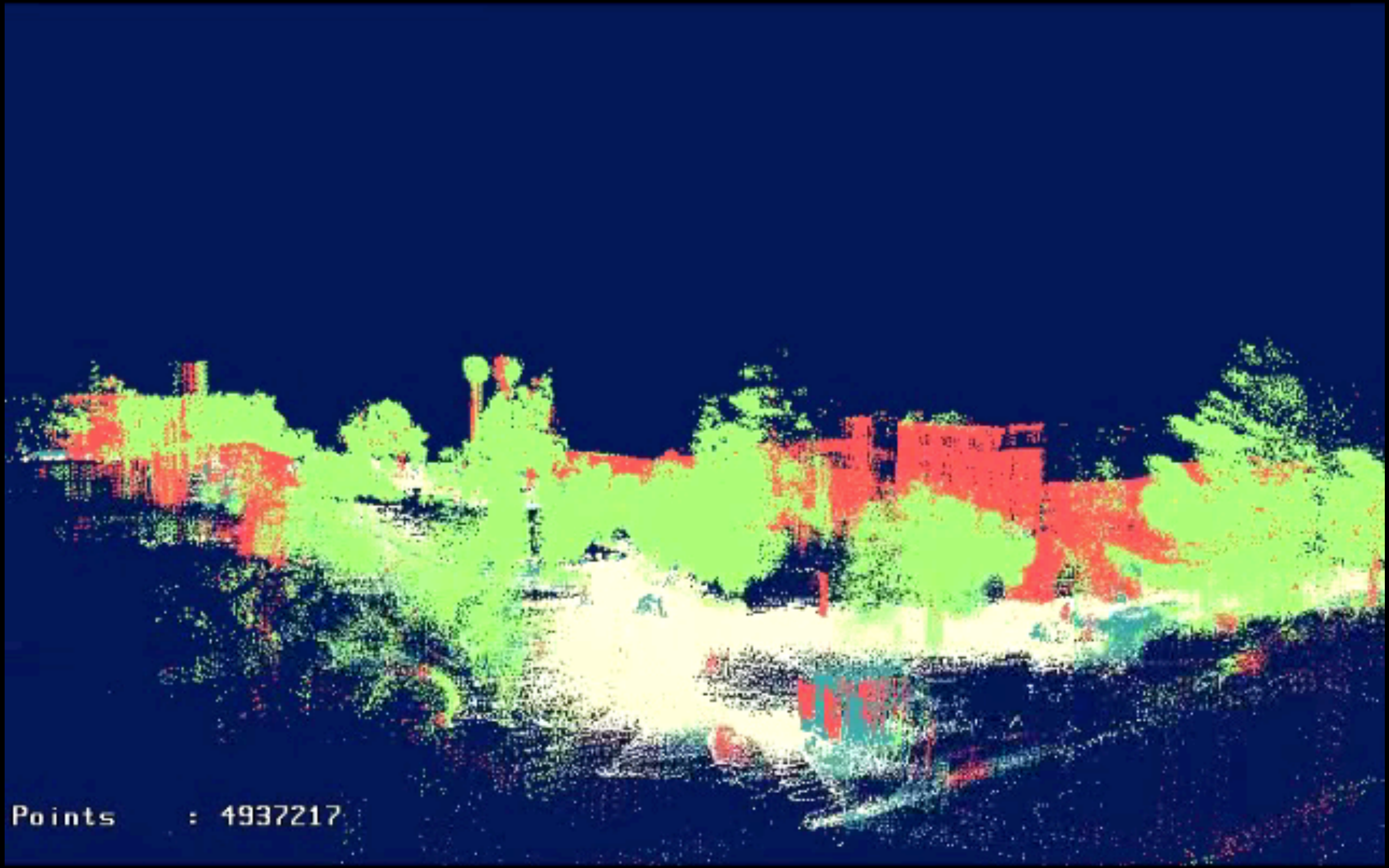
Discriminative Learning of Markov Random Fields for Segmentation of 3D Scan Data

Yixuan Li
Adam Holmes

Cornell University

3D Segmentation

e.g. 3D scan segmentation of Stanford University



http://ai.stanford.edu/~drago/Projects/Detector/index.html

3D Segmentation

- Why is this challenging?
  - 3D scanning points lack color clues
  - data is often noisy and sparse
  - extracting features for unseen object can be hard

1. How to compactly represent model?
2. How to efficiently inference with the model?

Paper Highlight

1. Higher prediction accuracy than SVM

   - Markov Random Fields (MRFs) incorporate both node and edge features
   - Enforce the preference that adjacent points have the same label.

2. Higher computational efficiency than CRFs

   - Formulate a compact quadratic programming based on maximum-margin framework , making the estimation tractable.
   - Scale up to tens of millions of points and multiple object classes.

Learning Algorithm Overview

- Learning Phase
  - Training data: scene points labeled with classes
  - Goal: find a good set of feature weights $\vec{w} = (\vec{w}_n, \vec{w}_e)$
  - Approach: maximum margin Markov network (M^3N)

- Segmentation Phase
  - Goal: classify the points of a new scene
  - Approach: compute both point and edge features and run the graph-cut algorithm using the weights $\vec{w}$.

Markov Random Fields

- Motivation: neighboring scan points can be correlated
- Definitions:
  - Markov network $G = (V, E)$, $V = \{1, 2, ..., N\}$
  - Edge (i, j): probabilistic interaction between nodes
  - Labels are the discrete variables $\mathbf{Y} = \{Y_1, Y_2, ..., Y_N\}$, where $Y_i \in \{1, 2, ..., K\}$

Markov Random Fields

- Nodes and edges associated with potentials $\phi_i(Y_i)$, $\phi_{ij}(Y_i, Y_j)$
- Associative Markov Network: $\phi_{ij}(k,k) \geq 1$, $\phi_{ij}(k,l) = 1 \ \forall k \neq l$
- The joint distribution specified by the network is:

$$P_\phi(\mathbf{y}) = \frac{1}{Z} \prod_{i=1}^{N} \phi_i(y_i) \prod_{ij \in E} \phi_i(y_i, y_j)$$

where Z is the partition function given by

$$Z = \sum_{y'} \prod_{i=1}^{N} \phi_i(y_i^{'}) \prod_{ij \in E} \phi_i(y_i^{'}, y_j^{'})$$

- Dependence of potentials on features: $\log \phi_i(k) = \mathbf{w}_n^k \cdot \mathbf{x}_i$ and $\log \phi_{ij}(k,k) = \mathbf{w}_e^k \cdot \mathbf{x}_{ij}$

Markov Random Fields

- *Maximum a posteriori* (MAP) inference:
  - find the maximum of the conditional distribution:

$$\log P_{\mathbf{w}}(\mathbf{y} \mid \mathbf{x}) = \sum_{i=1}^{N} \sum_{k=1}^{K} (\mathbf{w}_n^k \cdot \mathbf{x}_i) y_i^k + \sum_{ij \in E} \sum_{k=1}^{K} (\mathbf{w}_e^k \cdot \mathbf{x}_{ij}) y_i^k y_j^k$$

where $y_i^k = I(y_i = k)$

Markov Random Fields

- *Maximum a posteriori* (MAP) inference:
  - find the maximum of the conditional distribution:

$$\log P_{\mathbf{w}}(\mathbf{y} \mid \mathbf{x}) = \sum_{i=1}^{N} \sum_{k=1}^{K} (\mathbf{w}_n^k \cdot \mathbf{x}_i) y_i^k + \sum_{ij \in E} \sum_{k=1}^{K} (\mathbf{w}_e^k \cdot \mathbf{x}_{ij}) y_i^k y_j^k$$

where $y_i^k = I(y_i = k)$

  - formulate the problem as integer programming:

$$\max \sum_{i=1}^{N} \sum_{k=1}^{K} (\mathbf{w}_n^k \cdot \mathbf{x}_i) y_i^k + \sum_{ij \in E} (\mathbf{w}_e^k \cdot \mathbf{x}_{ij}) y_{ij}^k$$

$$s.t. \quad y_i^k, y_j^k \in \{0,1\}$$

$$y_{ij}^k \leq y_i^k, \;\; y_{ij}^k \leq y_j^k, \;\; \forall ij \in E, k.$$

Markov Random Fields

- *Maximum a posteriori* (MAP) inference:

  - find the maximum of the conditional distribution:

$$\log P_{\mathbf{w}}(\mathbf{y} \mid \mathbf{x}) = \sum_{i=1}^{N} \sum_{k=1}^{K} (\mathbf{w}_n^k \cdot \mathbf{x}_i) y_i^k + \sum_{ij \in E} \sum_{k=1}^{K} (\mathbf{w}_e^k \cdot \mathbf{x}_{ij}) \boxed{y_i^k y_j^k}$$

where $y_i^k = I(y_i = k)$

  - formulate the problem as integer programming:

$$\max \sum_{i=1}^{N} \sum_{k=1}^{K} (\mathbf{w}_n^k \cdot \mathbf{x}_i) y_i^k + \sum_{ij \in E} (\mathbf{w}_e^k \cdot \mathbf{x}_{ij}) \boxed{y_{ij}^k}$$

$$s.t. \quad y_i^k, y_j^k \in \{0,1\}$$

$$\boxed{y_{ij}^k \leq y_i^k, \quad y_{ij}^k \leq y_j^k, \quad \forall ij \in E, k.}$$

Markov Random Fields

- *Maximum a posteriori* (MAP) inference:

  - find the maximum of the conditional distribution:

$$\log P_{\mathbf{w}}(\mathbf{y} \mid \mathbf{x}) = \sum_{i=1}^{N} \sum_{k=1}^{K} (\mathbf{w}_n^k \cdot \mathbf{x}_i) y_i^k + \sum_{ij \in E} \sum_{k=1}^{K} (\mathbf{w}_e^k \cdot \mathbf{x}_{ij}) \boxed{y_i^k y_j^k}$$

  where $y_i^k = I(y_i = k)$

  - formulate the problem as integer programming:

$$\max \sum_{i=1}^{N} \sum_{k=1}^{K} (\mathbf{w}_n^k \cdot \mathbf{x}_i) y_i^k + \sum_{ij \in E} (\mathbf{w}_e^k \cdot \mathbf{x}_{ij}) \boxed{y_{ij}^k}$$

$$s.t. \quad y_i^k, y_j^k \in \{0,1\}$$

$$\boxed{y_{ij}^k \le y_i^k, \quad y_{ij}^k \le y_j^k, \quad \forall ij \in E, k.}$$

Why these two problems are equaivalent?

Markov Random Fields

- *Maximum a posteriori* (MAP) inference:

  - find the maximum of the conditional distribution:

$$\log P_{\mathbf{w}}(\mathbf{y} \mid \mathbf{x}) = \sum_{i=1}^{N} \sum_{k=1}^{K} (\mathbf{w}_n^k \cdot \mathbf{x}_i) y_i^k + \sum_{ij \in E} \sum_{k=1}^{K} (\mathbf{w}_e^k \cdot \mathbf{x}_{ij}) y_i^k y_j^k$$

  where $y_i^k = I(y_i = k)$

  - Using linear programming relaxation:

$$\max \sum_{i=1}^{N} \sum_{k=1}^{K} (\mathbf{w}_n^k \cdot \mathbf{x}_i) y_i^k + \sum_{ij \in E} (\mathbf{w}_e^k \cdot \mathbf{x}_{ij}) y_{ij}^k$$

$$s.t. \quad y_i^k \geq 0, \ \forall i, k; \ \ \sum_k y_i^k = 1, \ \forall i;$$

$$y_{ij}^k \leq y_i^k, \ \ y_{ij}^k \leq y_j^k, \ \ \forall ij \in E, k.$$

Learning the weights

- Maximum margin estimation:
  - Maximize our confidence in the true labels relative to any other possible joint labeling

Learning the weights

- Maximum margin estimation:
  - Maximize our confidence in the true labels relative to any other possible joint labeling

  - Define the gain of the true labels $\hat{\mathbf{y}}$ over another joint labeling $\mathbf{y}$ as:

$$\log P_{\mathbf{w}}\left(\hat{\mathbf{y}}|\mathbf{x}\right) - \log P_{\mathbf{w}}\left(\mathbf{y}|\mathbf{x}\right) = \mathbf{w}\mathbf{X}\left(\hat{\mathbf{y}} - \mathbf{y}\right)$$

Learning the weights

$$\log P_{\mathbf{w}}\left(\mathbf{y}|\mathbf{x}\right) \quad = \quad \sum_{i=1}^{N=2} \sum_{k=1}^{K=2} \left(\mathbf{w}_n^k \cdot \mathbf{x}_i\right) y_i^k + \sum_{i,j\in\mathcal{E}} \sum_{k=1}^{K=2} \left(\mathbf{w}_e^k \cdot \mathbf{x}_{ij}\right) y_{ij}^k$$

Learning the weights

$$\begin{aligned}\log P_{\mathrm{w}}(\mathbf{y}|\mathbf{x}) &= \sum_{i=1}^{N=2}\sum_{k=1}^{K=2}\left(\mathbf{w}_n^k\cdot\mathbf{x}_i\right)y_i^k+\sum_{i,j\in\mathcal{E}}\sum_{k=1}^{K=2}\left(\mathbf{w}_e^k\cdot\mathbf{x}_{ij}\right)y_{ij}^k\\ &= \left(\mathbf{w}_n^1\cdot\mathbf{x}_1\right)y_1^1+\left(\mathbf{w}_n^2\cdot\mathbf{x}_1\right)y_1^2+\left(\mathbf{w}_n^1\cdot\mathbf{x}_2\right)y_2^1+\left(\mathbf{w}_n^2\cdot\mathbf{x}_2\right)y_2^2\\ &\quad+\left(\mathbf{w}_e^1\cdot\mathbf{x}_{12}\right)y_{12}^1+\left(\mathbf{w}_e^2\cdot\mathbf{x}_{12}\right)y_{12}^2\end{aligned}$$

Learning the weights

$$
\begin{aligned}
\log P_{\mathbf{w}}(\mathbf{y}|\mathbf{x}) &= \sum_{i=1}^{N=2} \sum_{k=1}^{K=2} \left(\mathbf{w}_n^k \cdot \mathbf{x}_i\right) y_i^k + \sum_{i,j \in \mathcal{E}} \sum_{k=1}^{K=2} \left(\mathbf{w}_e^k \cdot \mathbf{x}_{ij}\right) y_{ij}^k \\
&= \left(\mathbf{w}_n^1 \cdot \mathbf{x}_1\right) y_1^1 + \left(\mathbf{w}_n^2 \cdot \mathbf{x}_1\right) y_1^2 + \left(\mathbf{w}_n^1 \cdot \mathbf{x}_2\right) y_2^1 + \left(\mathbf{w}_n^2 \cdot \mathbf{x}_2\right) y_2^2 \\
&\quad + \left(\mathbf{w}_e^1 \cdot \mathbf{x}_{12}\right) y_{12}^1 + \left(\mathbf{w}_e^2 \cdot \mathbf{x}_{12}\right) y_{12}^2 \\
&= \begin{pmatrix} \mathbf{w}_n^1 & \mathbf{w}_n^2 & \mathbf{w}_e^1 & \mathbf{w}_e^2 \end{pmatrix} \begin{pmatrix} \quad & \quad & \quad \\ \\ \\ \\ \\ \quad & \quad & \quad \end{pmatrix} \begin{pmatrix} y_1^1 \\ y_1^2 \\ y_2^1 \\ y_2^2 \\ y_{12}^1 \\ y_{12}^2 \end{pmatrix}
\end{aligned}
$$

Learning the weights

$$
\begin{aligned}
\log P_{\mathbf{w}}(\mathbf{y}|\mathbf{x}) &= \sum_{i=1}^{N=2}\sum_{k=1}^{K=2}\left(\mathbf{w}_n^k \cdot \mathbf{x}_i\right) y_i^k + \sum_{i,j\in\mathcal{E}}\sum_{k=1}^{K=2}\left(\mathbf{w}_e^k \cdot \mathbf{x}_{ij}\right) y_{ij}^k \\
&= \left(\mathbf{w}_n^1 \cdot \mathbf{x}_1\right) y_1^1 + \left(\mathbf{w}_n^2 \cdot \mathbf{x}_1\right) y_1^2 + \left(\mathbf{w}_n^1 \cdot \mathbf{x}_2\right) y_2^1 + \left(\mathbf{w}_n^2 \cdot \mathbf{x}_2\right) y_2^2 \\
&\quad + \left(\mathbf{w}_e^1 \cdot \mathbf{x}_{12}\right) y_{12}^1 + \left(\mathbf{w}_e^2 \cdot \mathbf{x}_{12}\right) y_{12}^2 \\
&= \begin{pmatrix} \mathbf{w}_n^1 & \mathbf{w}_n^2 & \mathbf{w}_e^1 & \mathbf{w}_e^2 \end{pmatrix}
\begin{pmatrix}
\mathbf{x}_1 & 0 & \mathbf{x}_2 & 0 & 0 & 0 \\
0 & \mathbf{x}_1 & 0 & \mathbf{x}_2 & 0 & 0 \\
0 & 0 & 0 & 0 & \mathbf{x}_{12} & 0 \\
0 & 0 & 0 & 0 & 0 & \mathbf{x}_{12}
\end{pmatrix}
\begin{pmatrix} y_1^1 \\ y_1^2 \\ y_2^1 \\ y_2^2 \\ y_{12}^1 \\ y_{12}^2 \end{pmatrix}
\end{aligned}
$$

Learning the weights

$$
\begin{aligned}
\log P_{\mathbf{w}}(\mathbf{y}|\mathbf{x}) &= \sum_{i=1}^{N=2}\sum_{k=1}^{K=2}\left(\mathbf{w}_n^k\cdot\mathbf{x}_i\right)y_i^k + \sum_{i,j\in\mathcal{E}}\sum_{k=1}^{K=2}\left(\mathbf{w}_e^k\cdot\mathbf{x}_{ij}\right)y_{ij}^k \\
&= \left(\mathbf{w}_n^1\cdot\mathbf{x}_1\right)y_1^1 + \left(\mathbf{w}_n^2\cdot\mathbf{x}_1\right)y_1^2 + \left(\mathbf{w}_n^1\cdot\mathbf{x}_2\right)y_2^1 + \left(\mathbf{w}_n^2\cdot\mathbf{x}_2\right)y_2^2 \\
&\quad + \left(\mathbf{w}_e^1\cdot\mathbf{x}_{12}\right)y_{12}^1 + \left(\mathbf{w}_e^2\cdot\mathbf{x}_{12}\right)y_{12}^2 \\
&= \begin{pmatrix}\mathbf{w}_n^1 & \mathbf{w}_n^2 & \mathbf{w}_e^1 & \mathbf{w}_e^2\end{pmatrix}
\begin{pmatrix}
\mathbf{x}_1 & 0 & \mathbf{x}_2 & 0 & 0 & 0 \\
0 & \mathbf{x}_1 & 0 & \mathbf{x}_2 & 0 & 0 \\
0 & 0 & 0 & 0 & \mathbf{x}_{12} & 0 \\
0 & 0 & 0 & 0 & 0 & \mathbf{x}_{12}
\end{pmatrix}
\begin{pmatrix} y_1^1 \\ y_1^2 \\ y_2^1 \\ y_2^2 \\ y_{12}^1 \\ y_{12}^2 \end{pmatrix} \\
&= \mathbf{w}\cdot\mathbf{X}\cdot\mathbf{y}
\end{aligned}
$$

Learning the weights

- Maximum margin estimation:
  - Maximize our confidence in the true labels relative to any other possible joint labeling

Learning the weights

- Maximum margin estimation:
  - Maximize our confidence in the true labels relative to any other possible joint labeling
  - Define the gain of the true labels $\hat{\mathbf{y}}$ over another joint labeling $\mathbf{y}$ as:

$$\log P_{\mathbf{w}}\left(\hat{\mathbf{y}}|\mathbf{x}\right) - \log P_{\mathbf{w}}\left(\mathbf{y}|\mathbf{x}\right) = \mathbf{w}\mathbf{X}\left(\hat{\mathbf{y}} - \mathbf{y}\right)$$

Learning the weights

- Maximum margin estimation:
  - Maximize our confidence in the true labels relative to any other possible joint labeling
  - Define the gain of the true labels $\hat{\mathbf{y}}$ over another joint labeling $\mathbf{y}$ as:

$$\log P_{\mathbf{w}}\left(\hat{\mathbf{y}}|\mathbf{x}\right) - \log P_{\mathbf{w}}\left(\mathbf{y}|\mathbf{x}\right) = \mathbf{w}\mathbf{X}\left(\hat{\mathbf{y}} - \mathbf{y}\right)$$

  - Want the gain to scale linearly with the number of mislabeled points $l\left(\hat{\mathbf{y}}, \mathbf{y}\right)$:

$$\max \gamma \text{ s.t. } \mathbf{w}\mathbf{X}\left(\hat{\mathbf{y}} - \mathbf{y}\right) \geq \gamma l\left(\hat{\mathbf{y}}, \mathbf{y}\right)$$

$$\left\|\mathbf{w}\right\|^2 \leq 1$$

Learning the weights

- Note that $l\left(\hat{\mathbf{y}}, \mathbf{y}\right) = N - \hat{\mathbf{y}}_n^\top \mathbf{y}_n$

Learning the weights

- Note that $l\left(\hat{\mathbf{y}}, \mathbf{y}\right) = N - \hat{\mathbf{y}}_n^\top \mathbf{y}_n$
- Add in a slack variable ξ

$$\min \quad \frac{1}{2}\mathbf{w}^2 + C\xi$$

$$\text{s.t.} \quad \mathbf{w}\mathbf{X}\left(\hat{\mathbf{y}} - \mathbf{y}\right) \geq N - \hat{\mathbf{y}}_n^\top \mathbf{y}_n - \xi, \forall \mathbf{y} \in \mathcal{Y}$$

Learning the weights

- Note that $l\left(\hat{\mathbf{y}}, \mathbf{y}\right) = N - \hat{\mathbf{y}}_n^\top \mathbf{y}_n$
- Add in a slack variable ξ

$$
\begin{aligned}
\min \quad & \frac{1}{2}\mathbf{w}^2 + C\xi \\
\text{s.t.} \quad & \mathbf{w}\mathbf{X}\left(\hat{\mathbf{y}} - \mathbf{y}\right) \geq N - \hat{\mathbf{y}}_n^\top \mathbf{y}_n - \xi, \forall \mathrm{y} \in \mathcal{Y}
\end{aligned}
$$

- Exponentially many constraints → replace them with a single nonlinear constraint,

$$
\mathbf{w}\mathbf{X}\hat{\mathbf{y}} - N + \xi \geq \max_{\mathbf{y} \in \mathcal{Y}} \mathbf{w}\mathbf{X}\mathbf{y} - \hat{\mathbf{y}}_n^\top \mathbf{y}_n
$$

Learning the weights

- Note that $l\left(\hat{\mathbf{y}}, \mathbf{y}\right) = N - \hat{\mathbf{y}}_n^\top \mathbf{y}_n$
- Add in a slack variable ξ

$$\begin{aligned} \min \quad & \frac{1}{2}\mathbf{w}^2 + C\xi \\ \text{s.t.} \quad & \mathbf{w}\mathbf{X}\left(\hat{\mathbf{y}} - \mathbf{y}\right) \geq N - \hat{\mathbf{y}}_n^\top \mathbf{y}_n - \xi, \forall \mathrm{y} \in \mathcal{Y} \end{aligned}$$

- Exponentially many constraints → replace them with a single nonlinear constraint,

$$\mathbf{w}\mathbf{X}\hat{\mathbf{y}} - N + \xi \geq \max_{\mathbf{y} \in \mathcal{Y}} \mathbf{w}\mathbf{X}\mathbf{y} - \hat{\mathbf{y}}_n^\top \mathbf{y}_n$$

- Where the right hand side can be found using the MAP

Learning the weights

- Plugging the dual of the MAP linear program, we get the following quadratic program for learning the weights:

$$\begin{aligned}
\min \quad & \frac{1}{2}|\mathbf{w}|^2 + C\xi \\
\text{s.t.} \quad & \mathbf{w}\mathbf{X}\hat{\mathbf{y}} - N + \xi \geq \sum_{i=1}^{N} \alpha_i;\ \mathbf{w}_e \geq 0; \\
& \alpha_i - \sum_{i,j \in \mathcal{E}} \alpha_{ij}^k \geq \mathbf{w}_n^k \cdot \mathbf{x}_i - y_i^k,\ \forall i, k; \\
& \alpha_{ij}^k + \alpha_{ji}^k \geq \mathbf{w}_e^k \cdot \mathbf{x}_{ij},\ \alpha_{ij}^k, \alpha_{ji}^k \geq 0,\ \forall i, j \in \mathcal{E}, k
\end{aligned}$$

Experiment: Terrain classification

•Data set: 3-D map of parts of Stanford from a robot equipped with a laser scanner

- 35 million noisy 3-D points

•Classify points into:

- Ground, Building, Tree, Shrubbery

•Classifying the ground is trivial

- Threshold the z-coordinate at ~0

Features

1. Distribution of surrounding points relative to principal plane

2. Distribution of points in vertical cylinder (r=0.25m)

3. Binary feature: whether within 2m of the ground

Edges

- Associative Markov Network (AMN) requires pair-wise connections ('edges')

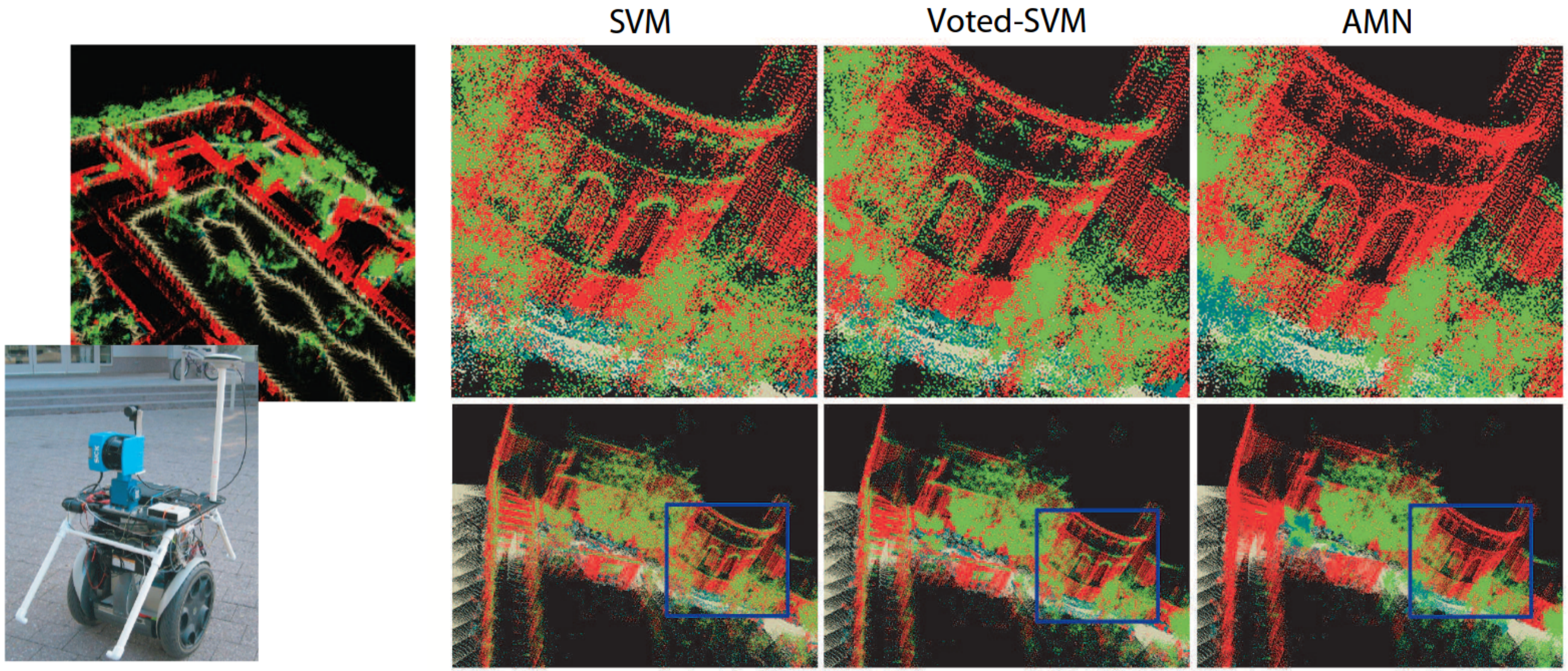
- Each point is randomly connected to 6 other points
  - 3 from sphere of radius 0.5m
  - 3 from vertical cylinder of radius 0.25m

Experimental Setup

•Training:
  - Roughly 30,000 images that represent the classes well

•Compare multi-class SVM, Voted SVM, and AMN
  - All used same training data and features

Results



a) Robot and campus map

b) Segmentation Results

Accuracy: SVM: 68%, Voted-SVM: 73%, AMN: 93%

Conclusions

•A simple Associative Markov Network (AMN) model was introduced for segmenting 3D image data

  - Model rewards cases where nearby points have the same label

•Classification is done by maximum a-posteriori (MAP) inference, which maximizes the log-likelihood

•Training of the weights is done by maximizing the margin between the log-likelihoods of the true labeling $\hat{\mathbf{y}}$ and any other labeling $\mathbf{y}$

•Experiments on classifying 3D images demonstrated a large gain in accuracy over an SVM