SemanticPaint

Adam Kosiorek [⋆]

Abstract. The short abstract (50-80 words) is intended to give the reader an overview of the work.

1 Introduction

Capturing your own environment has never been easier. SemanticPaint can register your surroundings which, after 3D reconstruction, can be semantically segmented in an interactive way. Not only it works in real time but also requires no pretraining. Adding new object categories on the fly is facilitated by online model updates. The user is provided with instantaneous feedback and can relabel any object to correct errors. SemanticPaint makes capturing customized environment models with object classes particular to the user's interest easy and efficient.

The pipeline starts with capturing the environment as a stream of noisy RGBD images and combining them into a 3D model updated in an online fashion. The user can choose which objects to label and can do so by "touching" a small part of an object or encircling it with his hand and uttering the label. The label and the affected data points are further passed to a Streaming Random Forest (SRF) classifier which constantly learns and labels all visible voxels. To further improve classification results a spatially dense labeling is produced by an efficient mean-field inference algorithm. One of the biggest strengths of SemanticPaint is the efficiency of each part of the pipeline, which translates to real time performance. Algorithms used in the pipeline were adapted to work on volumetric data in the TSDF format directly in order to avoiding costly conversions to mesh or point-cloud formats. To allow this, the Voxel Oriented Patch features — a new type of a discriminative feature describing the voxel space — has been designed.

2 Related Work

Acquisition and Reconstruction Capturing the geometry of the surrounding world has been a long standing problem [1]. An offline processing of multiple images allowed to reconstruct digital heritage and construct world-scale 3D models with remarkable quality [6]. The inception of low-cost RGBD sensors and powerful GPUs enabled online 3D scanning, augmented reality and using 3D environment models for navigation purposes [7]. [3] enables 3D reconstruction of small

[⋆] Advisor: M.Eng. Keisuke Tateno, Chair for Computer Aided Medical Procedures & Augmented Reality, TUM, WS 2015/16.

scenes using a single off-the-shelf RGB camera. It uses a sparse tracking method to first estimate the camera's pose and then select key frames and relative to them secondary frames, from which 3D stereo reconstruction is performed. The achieved results are similar to KinectFusion with the only limitation being that the precision of texture-less surfaces' reconstruction is somewhat lacking.

Scene Understanding Object recognition, detection and segmentation has been done with 2D RGB images, RGBD data, point clouds and volumetric representations. [2] uses a Voxel-based CRF for simultaneous reconstruction and segmentation. Each voxel contains information about visibility and occlusion as well as group membership. The first two are used to to improve reconstruction by mitigating depth-map noise. The visibility values are constrained by that each ray from the camera can hit only one visible vertex. The group membership information encodes priors given by bounding boxes of detected objects. A graph-cut algorithm is used for global inference. The whole scene has to be registered beforehand, since any change in the CRF's structure would require restarting the inference algorithm. No computational performance was reported. [4] is a first step towards online simultaneous registration and segmentation. Using RGBD images, the framework constructs a model of the environment and updates it each time a new frame comes in. When a significant change in the model is detected, the resulting model is split into a static and a dynamic part, where the latter is assumed to have moved in space. The movement is detected by comparing the expected and observed intensity values at each voxel. The system is online, but is far from real time with 0.7 to 2s processing time per frame. [5] shows that converting to another data representation can help to improve segmentation accuracy and inference speed. TSDF model is used to reconstruct a 3D scene from a sequence of RGBD images. The volumetric representation is triangulated and a 3D mesh is recovered. Next, visual features are computed directly on images and projected into the 3D model, while geometric features are computed on the mesh directly. Segmentation is done via a CRF. The approach is tested on the augmented KITTI and NYU datasets for indoor and outdoor scene segmentation where it delivers state-of-the-art results.

3 Space Representation

The space is represented by a 3D volume constructed from the sequence of RGBD images using Truncated Signed Distance Function (TSDF) [8]. Large-scale scene handling is facilitated by the hashed volumetric representation of [9] with a 6mm^3 voxel. Each voxel contains: TSDF distance, colour information in the LAB space, a weight for averaging distance values, a user annotation class, a predictied class, an outcome of the mean-field inference from before and after classification and the probability of each class as inferred by the mean-field algorithm, for each class. The total number of bits per voxel is $96 + [\text{number of classes}] \cdot 16$.

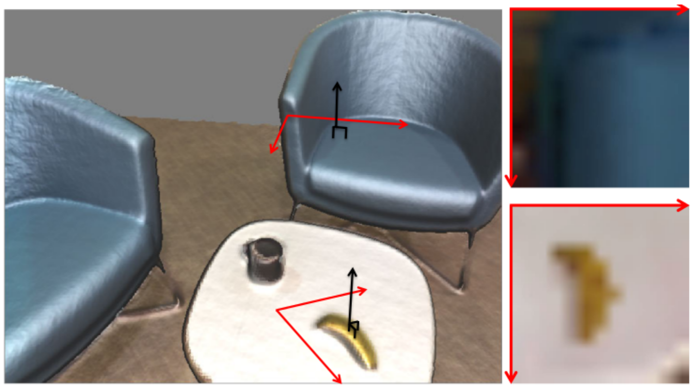


Fig. 1. Voxel Oriented Patch. Colours shown in RGB for illustration purposes.

4 Voxel Oriented Patches

This work relies heavily on classification and since even the most powerful classifier is useless if inappropriate features are supplied this paper contributes Voxel Oriented Patch (VOP) features. They are efficiently computed from the TSDF volume representation (cf. figure 1). Let V_i be a VOP and $\mathbf{n}^i$ the normal of voxel i. An image patch of dimensions $r \times r$ is centered on voxel i from the plane $(\mathbf{p} - \mathbf{p}_i) \cdot (n)_i = 0$. The patch containes colour values stored in the TSDF on the plane in CIELab to mitigate illumination effects. Additionaly, V_i stores distance to the nearest dominant horizontal surface. $r = 13$ with a resolution of $10\frac{mm}{pixel}$ is used. Rotation invariance is achieved exactly as in [12]. The patch is rotated according to the dominant gradient direction. It is computed by creating a histogram of gradient directions with 36 bins, where the directions are weighted by gradient magnitudes and a gaussian window centered at the voxel. Furthermore, a quadratic polynomial is fit to the three biggest values of the histogram, whose maximum is the final directiion.

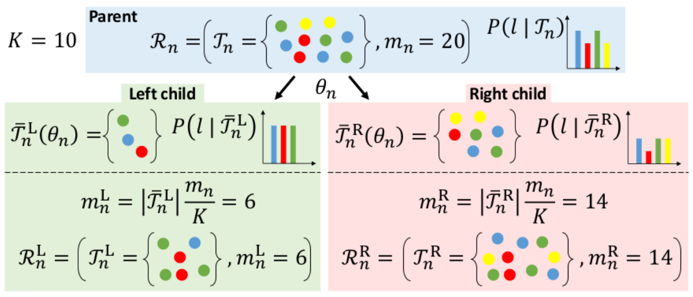
5 Streaming Random Forests

A random forest is an ensamble of classification or regression trees, whose outputs are combined to produce a result with lower variance [10]. Each tree is typically trained only on a subset of the training data S comprised of pairs (i, l) where i is a sample and l its label. Let $f(i; \theta) \in \{L, R\}$ denote the binary split function with learned parameters θ, which specify a feature this node uses. The tree's output amounts to the probability distribution $P_F(x_i = l|\mathbf{D})$ stored at each node. Trees learn in a greedy way: for each node a split function is chosen from a distribution Θ of candidate split functions to maximize the information gain.

$$G(S, S^L, S^R) = H(S) - \sum_{d \in \{L,R\}} \frac{|S^d|}{|S|} H(S^d) \tag{1}$$

Where $H(S) = -\sum_{(l,i)\in S} p(c_i = l) \log p(c_i = l)$ is Shannon Entropy. Training set is then split into subsets $S^d(\theta)$, $d \in \{L, R\}$ used to train child nodes. This procedure requires that all training data is available.

Online Random Forest [10] allow runtime model updates and they differ in that for a new node n they generate a distribution of split function Θ_n and maintain a set of label statistics for each of them, updated when a new sample comes to the node. Label statistics are sufficient to compute information gain and thus the node n is further split if it has seen a predefined number of samples or if the minimal value of the information gain is low enough. Storing and updating statistics in leaf nodes is computationally expensive which limits runtime performanc



$K = 10$
Parent
$\mathcal{R}_n = (\mathcal{T}_n = \{\ldots\}, m_n = 20)$
$P(l \mid \mathcal{T}_n)$
θ_n
Left child
Right child
$\bar{\mathcal{T}}_n^{\mathrm{L}}(\theta_n) = \{\ldots\}$
$P(l \mid \bar{\mathcal{T}}_n^{\mathrm{L}})$
$\bar{\mathcal{T}}_n^{\mathrm{R}}(\theta_n) = \{\ldots\}$
$P(l \mid \bar{\mathcal{T}}_n^{\mathrm{R}})$
$m_n^{\mathrm{L}} = |\bar{\mathcal{T}}_n^{\mathrm{L}}| \frac{m_n}{K} = 6$
$m_n^{\mathrm{R}} = |\bar{\mathcal{T}}_n^{\mathrm{R}}| \frac{m_n}{K} = 14$
$\mathcal{R}_n^{\mathrm{L}} = (\mathcal{T}_n^{\mathrm{L}} = \{\ldots\}, m_n^{\mathrm{L}} = 6)$
$\mathcal{R}_n^{\mathrm{R}} = (\mathcal{T}_n^{\mathrm{R}} = \{\ldots\}, m_n^{\mathrm{R}} = 14)$

Fig. 2. Splitting reservoirs in Streaming Random Forest.

Streaming Random Forest begins by creating a reservoir R_n of at most K samples stored in a list T_n at the root node. It represents an unbiased sample of all training data seen so far. Initially, the first K samples are stored. Then, a sample in the reservoir is exchanged with an incoming one with probability $p = \frac{K}{m_n}$, where m_n is the number of samples seen so far at node n. Splitting the node requires computing the objective function for each $\theta \in \Theta_n$ over R_n. $H(R_n)$ is computed from the normalized class label histogram $P(l|R_n)$. Let $|R_n|$ be the amount of samples stored in the reservoir n and let R_n^d be the splitting induced by θ. Finally, the split function is chosen as $f_n = \arg\max_{\theta\in\Theta_n} G(S, S^L, S^R|\theta)$ and set the number of samples seen by the child node to $|m_n^d| = |R_n^d| \max\left(1, \frac{m_n}{K}\right)$. Since the total number of samples stored is bounded by K this approach uses less memory. Constructing child reservoirs from parent reservoirs (cf. fig 2) lessens the computational load.

The GPU implementation can handle up to 17 million voxel classifications per second, while between 3 and 10 million voxels are visible at any one time [9]. To improve performance, voxels are batched and only a single batch is evaluated per frame.

6 Dynamic Conditional Random Field

Conditional Random Fields are often used for segmentation [2], but they usually assume availability of all data. Here, a pairwise CRF [13] with a time dependent underlying model is used. It requires a novel inference alogorithm that can handle updates of the geometry and user specified labes. The class of each voxel is modeled by a random variable x_i. The label distribution over the volume $\mathcal{V}$ factorizes into likelihood terms $\psi_i(x_i)$ and prior terms $\psi_{ij}(x_i, x_j)$. Let $\mathcal{E}_i$ be a neighbourhood of v_i, $\mathbf{D}_t$ volumetric data at time t and $\mathbf{x}$ a vector of all x_i. The posterior distribution is given by

$$P(\mathbf{x}|\mathbf{D}) = \prod_{i \in \mathcal{V}} \left(\psi_i(x_i) \prod_{j \in \mathcal{E}_i} \psi_{ij}(x_i, x_j) \right) \tag{2}$$

Neighbourhood of 6 cm is used. Negative log likelihood of eq. 2 defines the energy

$$E_t(\mathbf{x}) = \sum_{i \in \mathcal{V}} \left(\phi_i(x_i) + \sum_{j \in \mathcal{E}_i} \phi_{ij}(x_i, x_j) \right) + K \tag{3}$$

Unary and pairwise potentials $\phi_i(x_i)$ and $\phi_{ij}(x_i, x_j)$ encode the cost of assigning a label to v_i and a cost of assuming different labels by voxels in a neighbourhood, respectively. These costs are constatly changing due to the chainging predictions of the random forest and user interaction.

Initially, all voxels are encouraged to take a background label by setting $\phi_i(l) = 0$ for background class and grater than zero otherwise. When the user touches an object, a set of touched voxels $\mathcal{H}_S$ is registered and their unary potentials set to $\phi_i(l) = \infty$ if their labels are different then the specified one, and zero if they are the same, thus imposing the labeling. If user labels a region more than once, the old labels are simply overwritten.

To process the encircling action, the labeling is projected into the current frame. Next, a GMM is fit with the foreground class taken as a convex hull of the encirvled region and background as the rest of the image. For every pixel in a bounding box around the user annotation the unary potentials are updated as

$$\phi_i(l) = \begin{cases} \log P_E(fg|\mathbf{a}_i) & \text{if } l = \text{fg} \\ \log(1 - P_E(fg|\mathbf{a}_i)) & \text{if } l = \text{bg} \end{cases} \tag{4}$$

with $P_E(\text{fg}|\mathbf{a}_i)$ the probability of assuming the foreground label, which stems from the normalized likelihood $P(\mathbf{a}_i|\text{fg})$ from the GMM.

$$P_E(\text{fg}|\mathbf{a}_i) = \frac{P(\mathbf{a}_i|\text{fg})}{P(\mathbf{a}_i|\text{fg}) + P(\mathbf{a}_i|\text{bg})} \tag{5}$$

For all voxels that haven't been explicitly labaled by the user unitary potentials are updated with the predicted values as $\phi_i(l) = -\log P_F(x_i = l|\mathbf{D})$.

Finally, smoothness is achieved by the standard Potts model by assigning a disconuity cost $\phi_{ij}(x_i, x_j) = \lambda_{ij}$ if voxels have different labels and zero otherwise, with λ_{ij} a function of difference in position, intensity and normal directions.

7 Efficient Mean-Field Inference

The mean-field inference algorithm is adapted from [14] to handle the constantly changing energy landscape and implemented on GPU. First, the original prbability distribution $P(\mathbf{x})$ is approximated by $Q(\mathbf{x})$ under KL-divergence $KL(Q||P)$. $Q(\mathbf{x})$ is chosen such that the marginal of each random variable is independent, that is $Q(\mathbf{x}) = \prod_i Q_i(\mathbf{x})_\mathbf{i})$. Iterative updates yield:

$$Q_i^t(l) = \frac{1}{Z_i} e^{M_i(l)},\ t = 1, \dots, T \tag{6}$$

$$M_i(l) = \phi_i(l) + \sum_{l' \in \mathcal{L}} \sum_{j \in \mathcal{N}(i)} Q_j^{t-1}(l') \phi_{ij}(l, l') \tag{7}$$

with Z_i a normalizing factor and final class chosen as the minizer of Q_i^T. Energy distribution is assumed to change only gradualy. Moreover, SRF classification results would impact the final segmentation after several frames due to their effect on unitary potentials. To speed up the process the initial energy value at a new frame is set to a weighted sum of the previous frame's state and the SRF prediction. It works well in practice and allows visually pleasing label propagation effect.

$$\widetilde{Q}_i^t(x_i) = \gamma Q_i^{t-1}(x_i) + (1-\gamma) P_F^{t-1}(x_i = l|\mathbf{D}),\ \gamma \in [0, 1] \tag{8}$$

8 Results

Each part of the pipeline is evaluated separately. Several video sequences were recorded to evaluate segmentation and VOP features. A set of keyframe covering the whole scene from each sequence was hand-labeled to create groudtruth data. They were further projected into the 3D volume, resulting in 4176, 12346, 7583, 8916 frames for *LivingRoom*, *Kitchen*, *Bedroom* and *Desk* sequences respectively, of wich around a third was used for testing. Table 1 summarises segmantation results. User interaction gives very accurate labeling, but it affects a single object instance only. SRF prediction works well and is improved by further inference. Table 2 compares VOPs against other features. The numbers reflect the percentage of correctly classified pixels with SRF using different features. VOP clearly outperforms all other features. Δ RGB mean stands for the mean difference of two randomly selected patches.

SRF was compared against Online Random Forests (ORF) [10] and Hoeffding trees (HT) [16]. In order to simulate non-IID data, a dataset of 300 objects of 51 categories [17] was used. One revolution of each object was recorded with an RGBD camera from three different viewpoints. Online setting was created by

Table 1. Segmentation Results.

Component	**LivingRoom**	**Bedroom**	**Kitchen**	**Desk**	**Average**
User Interaction	99.35%	97.61%	96.09%	97.73%	97.7%
Forest Prediction	94.57%	88.31%	82.58%	90.29%	88.94%
Final Inference	96.26%	95.19%	90.69%	95.55%	94.42%

Table 2. Comparison of VOP against other features.

Feature	**LivingRoom**	**Bedroom**	**Kitchen**	**Desk**	**Average**
VOP	**94.57%**	**88.31%**	82.58%	**90.29%**	**88.94%**
Δ RGB mean	80%	71.84%	76.29%	73.42%	75.39%
Depth probe	77.54%	61.79%	**84.9%**	68.9%	73.06%
Color Probe	56.39%	65.68%	60.77%	60.74%	60.9%
SURF	43.74%	67.12%	57%	58.13%	56.5%
SPIN	58.77%	43.22%	48.41%	36.1%	46.63%

sequentially adding new categories. Two thirds of each viewpoint of each object is used for training. Results are summarized by figure 3. SRF clearly outperforms its competitors.

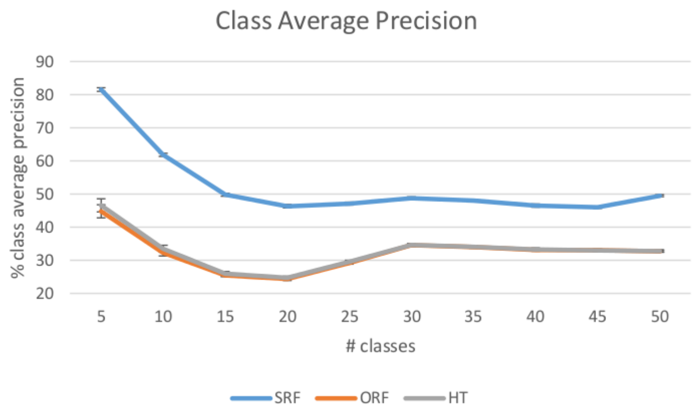
9 Discussion

This paper contributes an interactive system that enables users to create customized models of 3D enviornments. It can be used to gather groundtruth for large-scale visual recognition systems, robot navigation and to aid partially sighted people. There are some limitations, however. Use of the system is now restricted to simple indoor scenes. Computational load increases with the size of the scene while prediction accuracy drops with the number of object categories.

The system has also some limitations. Combing colour and depth data requires careful sensor calibration. Combing colour and depth data requires careful sensor calibration. Lack thereof results in misclassification at object boundaries. Additionaly, strong changes of lighting conditions impair classification ability at certain viewpoints. These failures are shwon in figure 4. Moreover, computational performance drops with the increasing size of the scene and classification accuracy decreases with the number of classes. What is more, no global context is used for classification.

10 Future Work

There is big room for improrvement. Introducting priors associated with the type of enviornment or with an object class (*e.g.* vertical walls) could greatly improve classification accuracy. Failure cases from Fig. 4 could be avoided if pure geometric features were used. Some research is required on scaling the system to bigger scenes and moving to outdoor enviornments.



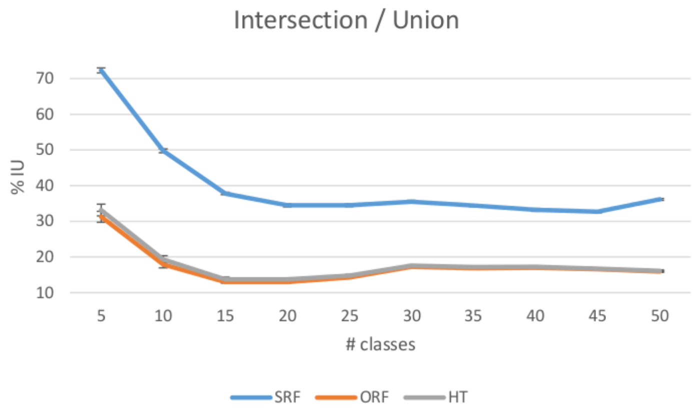


Fig. 3. Comparison of Streaming Random Forest against Online Random Forest and Hoeffding Trees.

References

1. ROBERTS , L. G. 1963. Machine perception of three-dimensional solids. Ph.D. thesis, Massachusetts Institute of Technology.
2. KIM , B.-S., KOHLI , P., AND SAVARESE , S. 2013. 3D scene understanding by voxel-CRF. In Proc. ICCV.
3. PRADEEP , V., RHEMANN , C., IZADI , S., ZACH , C., BLEYER , M., AND BATHICHE , S. 2013. Monofusion: Real-time 3D reconstruction of small scenes with a single web camera. In Proc. ISMAR.
4. HERBST , E., HENRY , P., AND FOX , D. 2014. Toward online 3-d object segmentation and mapping. In IEEE International Conference on Robotics and Automation (ICRA).
5. VALENTIN , J. P., SENGUPTA , S., WARRELL , J., SHAHROKNI , A., AND TORR , P. H. 2013. Mesh based semantic modelling for indoor and outdoor scenes. In Proc. CVPR.

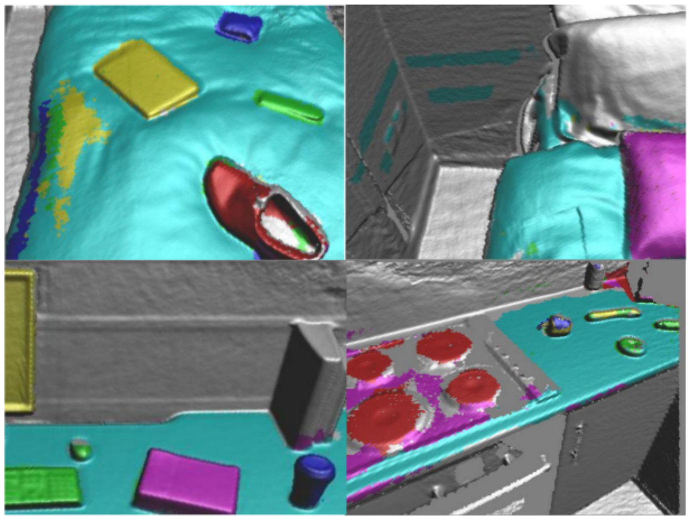


Fig. 4. Failure cases.

6. LEVOY , M., PULLI , K., CURLESS , B., RUSINKIEWICZ , S., KOLLER , D., EREIRA , L., GINZTON , M., ANDERSON , S., DAVIS , J., G NSBERG , J., ET AL . 2000. The digital Michelangelo project: 3D scanning of large statues. In Proc. SIGGRAPH. ACM.
7. NEWCOMBE , R. A., IZADI , S., HILLIGES , O., MOLYNEAUX , D., KIM , D., DAVISON , A. J., KOHLI , P., SHOTTON , J., HODGES , S., AND FITZGIBBON , A. 2011. KinectFusion: Real-time dense surface mapping and tracking. In Proc. ISMAR.
8. CURLESS , B. AND LEVOY , M. 1996. A volumetric method for building complex models from range images. In Proceedings of the 23rd annual conference on Computer graphics and interactive techniques. ACM, 303312.
9. NIESSNER , M., ZOLLHOFER , M., IZADI , S., AND STAMMINGER , M. 2013. Real-time 3D reconstruction at scale using voxel hashing. ACM TOG 32, 6
10. SAFFARI , A., LEISTNER , C., SANTNER , J., GODEC , M., AND BISCHOF , H. 2009. On-line random forests. In IEEE ICCV Workshop.
11. VITTER , J. S. 1985. Random sampling with a reservoir. ACM TOMS 11, 1.
12. LOWE , D. G. 1999. Object recognition from local scale-invariant features. In Proc. ICCV.
13. LAFFERTY , J., McCALLUM , A., AND PEREIRA , F. C. 2001. Conditional random fields: Probabilistic models for segmenting and labeling sequence data.
14. KRAHENBHL, P. AND KOLTUN , V. 2011. Efficient inference in fully connected CRFs with Gaussian edge potentials. In NIPS.
15. KOLLER , D. AND FRIEDMAN , N. 2009. Probabilistic Graphical Models: Principles and Techniques. MIT Press
16. DOMINGOS , P. AND HULTEN , G. 2000. Mining high-speed data streams. In Proc. SIGKDD.
17. LAI , K., BO , L., REN , X., , AND FOX , D. 2011. A large-scale hierarchical multi-view rgb-d object dataset. In Proc. ICRA.