An Overview of SemanticPaint

Adam Kosiorek [*]

Abstract. We describe SemanticPaint [14] — a system for capturing and interactive segmentation of 3D environments. It allows adding new object categories at runtime, updates the model with and infers labels of newly seen data in an online fashion.

1 Introduction

Capturing your own environment has never been easier. SemanticPaint [14] 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.

2 Related Work

Scene Understanding Object recognition, detection and segmentation has been done with colour and RGBD images, point clouds, meshes and volumetric representations. [4] uses a TSDF model 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 on images and projected into the 3D model, while geometric features are computed on the mesh directly. Segmentation is done via a CRF. This approach achieves state-of-the-art results on KITTI and NYU datasets for indoor and outdoor scene segmentation. [1] uses a Voxel-based CRF for offline simultaneous reconstruction and segmentation. Each voxel contains information about visibility and occlusion as well as group membership. The first two are used 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. [3] is a first step towards online registration and segmentation. Using RGBD images, the framework constructs a model of the environment and updates it with each new frame. When a significant change in the model is detected, it is split into a static and a dynamic part, where the latter is assumed to have moved in space.

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

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.

Model-based SLAM KinectFusion [5] has brought online 3D reconstruction from a single depth sensor. It uses model-based tracking to match an incoming frame against a global volume in the TSDF [6] format. SLAM++ extends KinectFusion by object classification capabilities. It classifies objects by matching object in the scene against a database of objects in real time. It then builds an object-pose graph, which is a very sparse and compact representation of the world. It works online, but requires a previously prepared object database. [2] enables 3D reconstruction of small scenes using a single 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 the poor precision at texture-less surfaces.

3 Pipeline Overview

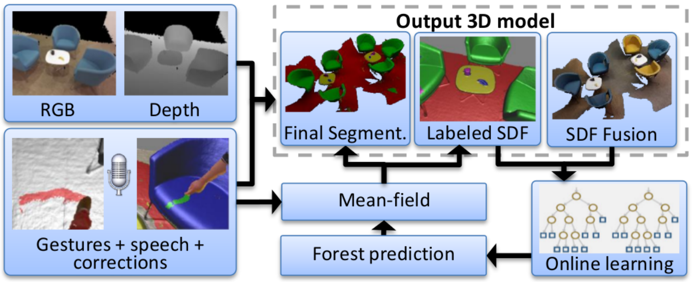

Fig. 1: The SemanticPaint Pipeline.

The pipeline (cf. Fig. 1) starts with capturing the environment as a stream of noisy RGBD images and combining them into a 3D model updated in an online fashion, similarly to KinectFusion [5]. The user can choose which objects to label and does 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 used to update a Conditional Random Field (CRF) segmentation model. They are also passed to a Streaming Random Forest (SRF), an online version of Random Forest classifier, which learns to predict labels of previously unseen elements of the scene. Finally, when the user activates the 'test' mode, the SRFs predictions are used to update the CRF. 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 [6] format directly.

4 Voxel Oriented Patches

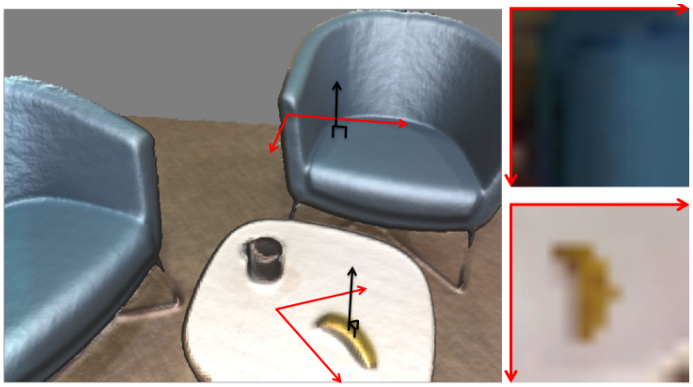

Fig. 2: Voxel Oriented Patch. Colours shown in RGB for illustration purposes.

Voxel Oriented Patch (VOP, cf. Fig. 2) features are used to guide classification efforts. They are efficiently computed from the TSDF volume representation. 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 and taken from the plane $(\mathbf{p} - \mathbf{p}_i) \cdot \mathbf{n}_i = 0$. The patch contains colour values stored in the TSDF on the plane in CIELab to mitigate illumination effects. Additionally, 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 [9].

5 Streaming Random Forests

A random forest is an ensemble of classification or regression trees, whose outputs are combined to produce a result with lower variance [7]. 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.

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

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.

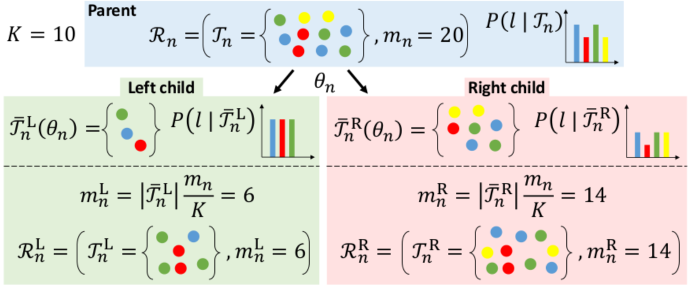
K = 10
Parent
$\mathcal{R}_n = \left(\mathcal{T}_n = \{\ \}, m_n = 20\right)$
$P(l \mid \mathcal{T}_n)$
Left child
θ_n
Right child
$\bar{\mathcal{T}}_n^{\mathrm{L}}(\theta_n) = \{\ \}$
$P(l \mid \bar{\mathcal{T}}_n^{\mathrm{L}})$
$\bar{\mathcal{T}}_n^{\mathrm{R}}(\theta_n) = \{\ \}$
$P(l \mid \bar{\mathcal{T}}_n^{\mathrm{R}})$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

Fig. 3: Splitting reservoirs in Streaming Random Forest.

Streaming Random Forest is an inherently online variant of a Random Forest classifier. It begins by creating a reservoir R_n of at most K samples stored in a list T_n at the root node, which 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 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 only limited memory. Constructing child reservoirs from parent reservoirs (cf. Fig 3) lessens computational load.

6 Dynamic Conditional Random Field

Conditional Random Fields are often used for segmentation [1], but they usually assume availability of all data. Here, a pairwise CRF [10] with a time dependent underlying model is used. It requires a novel inference algorithm that can handle updates of the geometry and user specified labels. 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

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

By taking negative log likelihood of eq. 2 we arrive at 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 potential $\phi_i(x_i)$ encodes the cost of assigning a label to v_i, while $\phi_{ij}(x_i, x_j)$ is a prior cost of assuming different labels.

Initially, all voxels are encouraged to take the background label by penalizing all other ones. 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, thus imposing the labeling. If user labels a region more than once, the old labels are simply overwritten. To process an encircling action, a GMM is fit with the foreground class taken as a convex hull of the encircled region and the background class 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. For all voxels that haven't been explicitly labeled 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 discontinuity 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 [11] to handle the constantly changing energy landscape and implemented on GPU. First, the original probability 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:

$$Q_i^t(l) = \frac{1}{Z_i} e^{M_i(l)},\ t = 1, \ldots, T \tag{5}$$

$$M_i(l) = \phi_i(l) + \sum_{l' \in \mathcal{L}} \sum_{j \in \mathcal{N}(i)} Q_j^{t-1}(l')\phi_{ij}(l, l') \tag{6}$$

with Z_i a normalizing factor and final class chosen as the minimizer of Q_i^T. Energy distribution is assumed to change only gradually. 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.

$$\widetilde{Q}_i^t(x_i) = \gamma Q_i^{t-1}(x_i) + (1-\gamma) P_F^{t-1}(x_i = l|\mathbf{D}),\ \gamma \in [0,1] \tag{7}$$

8 Results

Each part of the pipeline is evaluated separately. Several video sequences were recorded to evaluate segmentation and VOP features. A set of key frames covering the whole scene from each sequence was hand-labeled to create groundtruth 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 which around a third was used for testing. Table 1 summarizes segmentation 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.

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%

SRF was compared against Online Random Forests (ORF) [7] and Hoeffding trees (HT) [12]. In order to simulate non-IID data, a dataset of 300 objects of 51 categories [13] was used. One revolution of each object was recorded with an RGBD camera from three different viewpoints. Online setting was created by sequentially adding new categories. Two thirds of each viewpoint of each object is used for training. Results are summarized by Fig. 5. SRF clearly outperforms its competitors.

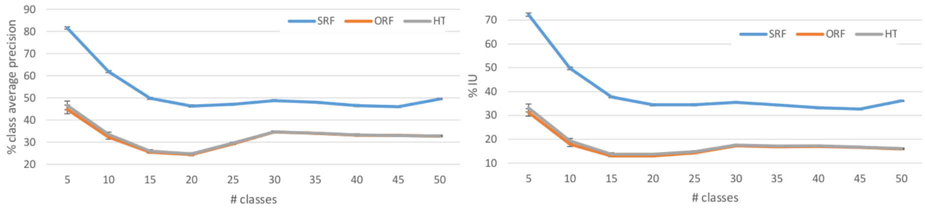

Fig. 5: Comparison of Streaming Random Forest against Online Random Forest and Hoeffding Trees

9 Discussion

This paper contributes an interactive system that enables users to create customized models of 3D environments. It can be used to gather groundtruth for large-scale visual recognition systems, robot navigation and to aid partially sighted people.

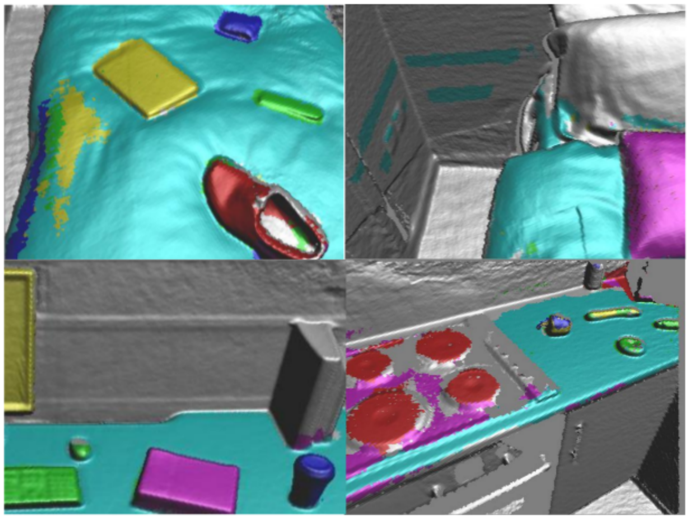

Fig. 6: Failure cases.

The system has some limitations, however. 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. Additionally, strong changes of lighting conditions impair classification ability at certain viewpoints. These failures are shown in Fig. 6. 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 some room for improvement. Introducing priors associated with the type of environment or with an object class (*e.g.* vertical walls) could greatly improve classification accuracy. Failure cases from Fig. 6 could be avoided if pure geometric features were used. Some research is required on scaling the system to bigger scenes and moving to outdoor environments.

References

1. Kim, B.-S., Kohli, P., and Savarese, S. 2013. 3d Scene Understanding By Voxel-Crf. In Proc. ICCV.
2. 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.
3. 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).
4. 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.
5. Newcombe, R. A., Izadi, S., Hilliges, O., Molyneaux, D., Kim, D., Davison, A. J., Kohli, P., Shotton, J., Hodges, S., and Fitzgib-Bon, A. 2011. Kinectfusion: Real-Time Dense Surface Mapping and Tracking. In Proc. ISMAR.
6. 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.
7. Saffari, A., Leistner, C., Santner, J., Godec, M., and Bischof, H. 2009. On-Line Random Forests. In IEEE ICCV Workshop.
8. Vitter, J. S. 1985. Random Sampling With A Reservoir. ACM Toms 11, 1.
9. Lowe, D. G. 1999. Object Recognition From Local Scale-Invariant Features. In Proc. ICCV.
10. Lafferty, J., Mccallum, A., and Pereira, F. C. 2001. Conditional Random Fields: Probabilistic Models For Segmenting and Labeling Sequence Data.
11. Krahenbhl, P. and Koltun, V. 2011. Efficient Inference In Fully Connected Crfs With Gaussian Edge Potentials. In NIPS.
12. Domingos, P. and Hulten, G. 2000. Mining High-Speed Data Streams. In Proc. SIGKDD.
13. Lai, K., Bo, L., Ren, X.,, and Fox, D. 2011. A Large-Scale Hierarchical Multi-View Rgb-D Object Dataset. In Proc. ICRA.
14. Valentin, J., Vineet, V., Cheng, M.-M., Kim, D., Shotton, J., Kohli, P., Niessner, M., Criminisi, A., Izadi, S., Torr, P. 2015. SemanticPaint: Interactive 3d Labeling and Learning At Your Fingertips. SIGGRAPH.