HAND SEGMENTATION FOR HAND-OBJECT INTERACTION FROM DEPTH MAP

Byeongkeun Kang* *Kar-Han Tan*† *Nan Jiang** *Hung-Shuo Tai*† *Daniel Tretter*‡ *Truong Nguyen**

* Department of Electrical and Computer Engineering, UC San Diego, La Jolla, CA 92093 USA
† NovuMind Inc., Santa Clara, CA 95054 USA
‡ Hewlett-Packard, Inc., Palo Alto, CA 94304 USA

ABSTRACT

Hand segmentation for hand-object interaction is a necessary preprocessing step in many applications such as augmented reality, medical application, and human-robot interaction. However, typical methods are based on color information which is not robust to objects with skin color, skin pigment difference, and light condition variations. Thus, we propose hand segmentation method for hand-object interaction using only a depth map. It is challenging because of the small depth difference between a hand and objects during an interaction. To overcome this challenge, we propose the two-stage random decision forest (RDF) method consisting of detecting hands and segmenting hands. To validate the proposed method, we demonstrate results on the novel dataset of hand segmentation for hand-object interaction. The proposed method achieves high accuracy in short processing time comparing to the other state-of-the-art methods.

Index Terms— Hand segmentation, human-machine interaction, random decision forest, depth map

1. INTRODUCTION

Recently, with the expansion of virtual reality (VR), augmented reality (AR), robotics, and intelligent vehicles, the development of new interaction technologies has become unavoidable since these applications require more natural interaction methods rather than input devices. For these applications, many researches have been conducted such as gesture recognition and hand pose estimation. However, most technologies focus on understanding interactions which do not involve touching or handling any real world objects although understanding interactions with objects is important in many applications. We believe that this is because hand segmentation is much more difficult in hand-object interaction. Thus, we present a framework of hand segmentation for hand-object interaction.

1.1. Related work

Hand segmentation has been studied for many applications such as hand pose estimation [1–6], hand tracking [7–9], and gesture/sign/grasp recognition [10, 11]. In color image-based methods, skin color-based method has been popular [10, 12–16]. For hand-object interaction, Oikonomidis *et al.* and Romero *et al.* segmented hands by thresholding skin color in HSV space [4, 5, 7, 8]. Wang *et al.* processed hand segmentation using a learned probabilistic model where the model is constructed from the color histogram of the first frame [6]. Tzionas *et al.* applied skin color-based segmentation using the Gaussian mixture model [17]. However, skin color-based segmentation has limitations in interacting with objects in skin color, segmenting from other body parts, skin pigment difference, and light condition variations. An alternative method is wearing a specific color glove [18].

For depth map-based methods, popular methods are using a wrist band [3, 9, 11] or using random decision forest (RDF) [1, 2, 19]. Although the method using a black wristband is simple and effective, it is inconvenient. Moreover, the method cannot segment hands from objects during hand-object interaction since it processes segmentation by finding connected components. Tompson *et al.* [1] and Sharp *et al.* [2] proposed the RDF-based methods based on [19]. Although the purposes of the methods are slightly different comparing to the proposed method, the methods are the most relevant methods.

In this paper, we propose the hand segmentation method for hand-object interaction using only a depth map to avoid the limitations of skin color-based methods. We present the two-stage RDF method to achieve high accuracy efficiently.

2. METHOD

We propose two-stage RDF for hand segmentation for hand-object interaction. In our two-stage RDF, the first RDF detects hands by processing the RDF on an entire depth map. Then, the second RDF segments hands in pixel-level by applying the RDF in the detected region. This cascaded architecture is designed for the second RDF to focus on the segmentation of hands from objects and close body parts such as an arm.

This work is supported in part by NSF grant IIS-1522125.

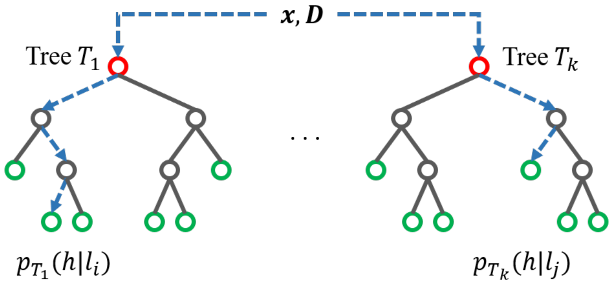

Fig. 1. Random decision forest. Red, black, and green circles represent root nodes, split nodes, and leaf nodes, respectively.

RDF consists of a collection of decision trees as shown in Fig. 1. Each decision tree is composed of a root node, splitting nodes, and leaf nodes. Given an input data at the root node, it is classified to child nodes based on the split function at each splitting node until it reaches a leaf node. In this paper, the input data is the location of each pixel on a depth map. The split function uses the feature of the depth difference between two relative points on the depth map in [19]. At a leaf node, a conditional probability distribution is learned in a training stage, and the learned probability is used in a testing stage. For more details about RDF, we refer the readers to [20–22].

2.1. Training

Given a training dataset $\mathcal{D}$, the algorithm randomly selects a set $\mathcal{D}_i$ of depth maps $\boldsymbol{D}$ and then randomly samples a set of data points $\boldsymbol{x}$ in the region of interest (ROI) on the selected depth maps $\boldsymbol{D}$. The ROI is the entire region on the depth maps in the first stage. It is the detected regions using the first RDF in the second stage (see Fig. 2). The sampled set of data points $\boldsymbol{x}$ and the corresponding depth maps $\boldsymbol{D}$ are inputs to the training of a decision tree.

Using the inputs ($\boldsymbol{x}$, $\boldsymbol{D}$), the algorithm learns a split function at each splitting node and a conditional probability distribution at each leaf node. First, learning the split function includes learning a feature $f(\cdot)$ and a criteria θ. We use the feature $f(\cdot)$ of the depth difference between two relative points $\{\boldsymbol{x}+\boldsymbol{u}/\boldsymbol{D}_{\boldsymbol{x}}, \boldsymbol{x}+\boldsymbol{v}/\boldsymbol{D}_{\boldsymbol{x}}\}$ in [19] as follows:

$$f(\boldsymbol{x}, \boldsymbol{D}, \boldsymbol{u}, \boldsymbol{v}) = D_{\boldsymbol{x}+\boldsymbol{u}/\boldsymbol{D}_{\boldsymbol{x}}} - D_{\boldsymbol{x}+\boldsymbol{v}/\boldsymbol{D}_{\boldsymbol{x}}} \tag{1}$$

where $\boldsymbol{D}_{\boldsymbol{x}}$ denotes the depth at a pixel $\boldsymbol{x}$ on a depth map $\boldsymbol{D}$; $\boldsymbol{u} \in \mathbb{R}^2$ and $\boldsymbol{v} \in \mathbb{R}^2$ represent offset vectors for each relative point. Then, the criteria θ decides to split the data $\boldsymbol{x}$ to the left child or the right child.

$$f(\boldsymbol{x}, \boldsymbol{D}, \boldsymbol{u}, \boldsymbol{v}) \lesseqgtr \theta \tag{2}$$

Thus, the algorithm learns two offset vectors $(\boldsymbol{u}, \boldsymbol{v})$ and a criteria θ at each splitting node.

Since the goal is separating the data points $\boldsymbol{x}$ of different classes to different child nodes, the objective function is designed to evalutate the separation using the learned offset vectors and criteria as follows:

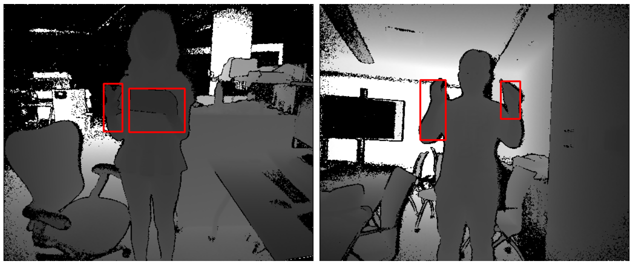

Fig. 2. Detection of hands using the RDF in the first stage.

$$L(\boldsymbol{x}, \boldsymbol{D}, \boldsymbol{u}, \boldsymbol{v}, \theta) = -\sum_{c\in\{l,r\}} \sum_{h\in\{0,1\}} \frac{|\boldsymbol{x}_c|}{|\boldsymbol{x}|} p(h|c) \log p(h|c) \tag{3}$$

where c and h are indexes for child nodes $\{l, r\}$ and for classes, respectively; $|\boldsymbol{x}_c|$ denotes the number of data points in the c child node; $p(h|c)$ is the estimated probability of being the class h at the child node c.

To learn offsets and a criteria, the algorithm randomly generates possible candidates and selects the candidate with a minimum loss $L(\cdot)$ as follows:

$$(\boldsymbol{u}, \boldsymbol{v}, \theta) = \operatorname*{argmin}_{(\boldsymbol{u},\boldsymbol{v},\theta)} L(\boldsymbol{x}, \boldsymbol{D}, \boldsymbol{u}, \boldsymbol{v}, \theta) \tag{4}$$

Learning a split function at each splitting node is repeated until the node satisfies the condition for a leaf node. The condition is based on (1) the maximum depth of the tree, (2) the probability distribution $p(h|c)$, and (3) the amount of training data $|\boldsymbol{x}|$ at the node. Specifically, it avoids too many splitting nodes by limiting the maximum depth of the tree and by terminating if the child node has a high probability for a class or if the amount of remaining training data is too small.

At each leaf node, the algorithm stores the conditional probability $p(h|l)$ (probability of being each class h given reaching the node l) for the prediction in a testing stage.

2.2. Testing

Using the learned RDF, the algorithm predicts the probability of being a class for a new data $\boldsymbol{x}$. The new data is classified to child nodes using the learned split function at each splitting node until it reaches a leaf node. At the leaf node l, the learned conditional probability $p_T(h|l)$ is loaded. These steps are repeated for entire trees T in the forest $\mathcal{T}$. Then, the probabilities are averaged to predict the probability $p(h|\boldsymbol{x})$ of being a class h for the new data $\boldsymbol{x}$.

$$p(h|\boldsymbol{x}) = \frac{1}{|\mathcal{T}|} \sum_{T\in\mathcal{T}} p_T(h|l) \tag{5}$$

where $|\mathcal{T}|$ is the number of trees in the learned forest $\mathcal{T}$.

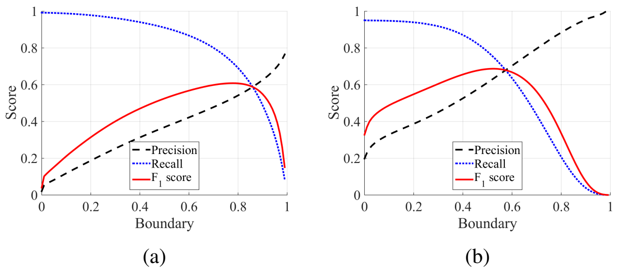

Fig. 3. Scores depending on the decision boundary on the validation dataset. (a) Score of the RDF in the first stage. (b) Score of the two-stage RDF with filtering in Section 2.3.

In the first stage, the first RDF is applied on an entire depth map to compute a probability map. Then, the probability map is used to detect hands as shown in Fig. 2. In the second stage, the second RDF processes the data points in the detected regions to predict the probability of being each class. The proposed two-stage RDF improves both accuracy and efficiency by focusing on each task in each stage.

Decision boundaries are exhaustively searched with the step size of 0.01 using the predicted probability maps of the validation dataset as shown in Fig. 3. Although the most typical boundary is 0.5 for a probability map, we found that it is not the best parameter. The selected boundaries are shown in Table 1.

2.3. Modified bilateral filter

Before classifying a data $\boldsymbol{x}$ to a class h, modified bilateral filter is applied to the predicted probability $p(h|\boldsymbol{x})$ to make the probability more robust. Since the probability $p(h|\boldsymbol{x})$ is predicted for each pixel independently, the probability is stabilized by averaging using the probabilities of the data points in close distance and similar intensity on the depth map.

Unlike typical bilateral filter whose weights are based on the input image (in this case, the probability map) [23], the weights in the modified bilateral filter are based on a separate image, the depth map. The filtering is defined as follows:

$$\widetilde{p}(h|\boldsymbol{x}) = \frac{1}{w} \sum_{\boldsymbol{x}_i \in \Omega} g_r(|\boldsymbol{D}_{\boldsymbol{x}_i} - \boldsymbol{D}_{\boldsymbol{x}}|) g_s(\|\boldsymbol{x}_i - \boldsymbol{x}\|) p(h|\boldsymbol{x}_i) \quad (6)$$

where Ω is the set of pixels within the filter's radius and the pre-defined depth difference; w is the normalization term, $w = \sum_{\boldsymbol{x}_i \in \Omega} g_r(|d_{\boldsymbol{I}}(\boldsymbol{x}_i) - d_{\boldsymbol{I}}(\boldsymbol{x})|) g_s(\|\boldsymbol{x}_i - \boldsymbol{x}\|)$; $g_r(\cdot)$ and $g_s(\cdot)$ are the Gaussian functions for the depth difference r and for the distance s from the data point $\boldsymbol{x}$, respectively. $g_r(r) = \exp(-\frac{r^2}{2\sigma_r^2}); g_s(s) = \exp(-\frac{s^2}{2\sigma_s^2})$. The parameters in the filter were selected based on the experiments using validation dataset. The selected parameters are as follows: the maximum depth difference to be considered is $400mm$. Both standard deviations (σ_r and σ_s) are 100.

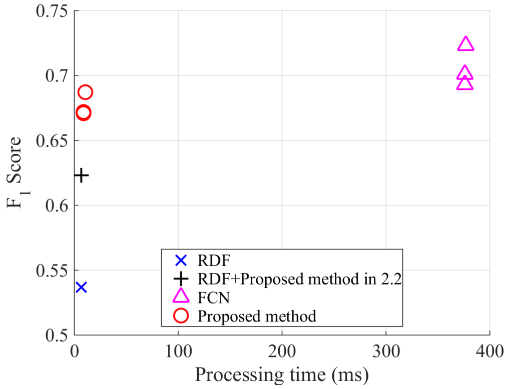

Fig. 5. Analysis of accuracy and efficiency.

3. EXPERIMENTAL EVALUATIONS

3.1. Dataset

We collected a new dataset using Microsoft Kinect v2 since we were not able to find a publicly available dataset for hand-object interaction with pixel-wise annotation. The newly collected dataset consists of 27,525 pairs of depth maps and ground truth labels from 6 people (3 males and 3 females) interacting with 21 different objects. Also, the dataset includes the cases of one hand and both hands in a scene. The dataset is separated into 19,470 pairs for training, 2,706 pairs for validation, and 5,349 pairs for testing, respectively. The dataset will be publicly available.

3.2. Results

The proposed method is analyzed by demonstrating the results on the dataset in Section 3.1. For the quantitative comparison of accuracy, we measure F_1 score, precision, and recall as follows:

$$\text{precision} = \frac{\text{tp}}{\text{tp} + \text{fp}}, \quad \text{recall} = \frac{\text{tp}}{\text{tp} + \text{fn}}$$
$$F_1 = 2 \times \frac{\text{precision} \times \text{recall}}{\text{precision} + \text{recall}} \quad (7)$$

where tp, fp, and fn represent true positive, false positive, and false negative, respectively. For the comparison of efficiency, we measure the processing time using a machine with Intel i7-4790K CPU and Nvidia GeForce GTX 770.

The proposed method is compared with the RDF-based method in [1, 19] and the fully convolutional networks (FCN) in [24, 25] using only a depth map. The proposed method is not compared with color-based methods since the characteristics of depth sensors and color imaging sensors are quite different. For example, a captured depth map using a depth sensor does not vary depending on light condition. However, a captured color image varies a lot depending on light condition. Thus, choosing the capturing environment affects the

Table 1. Quantitative comparison. The two boundaries for the proposed method are for each stage.

Method			Score			Processing time
Method	Boundary	Filter	Precision	Recall	F_1 score	(ms)
RDF [1, 19]	0.50	-	0.381	0.912	0.537	6.7
RDF [1, 19] + Proposed in Sec. 2.2	0.78	-	0.545	0.727	0.623	6.7
FCN-32s [24, 25]	-	-	0.700	0.686	0.693	376
FCN-16s [24, 25]	-	-	0.680	0.722	0.701	376
FCN-8s [24, 25]	-	-	0.704	0.744	0.723	377
Proposed method	0.50, 0.50	-	0.592	0.774	0.671	8.9
	0.50, 0.52	-	0.608	0.751	0.672	8.9
	0.50, 0.52	11 × 11	0.629	0.756	0.687	10.7

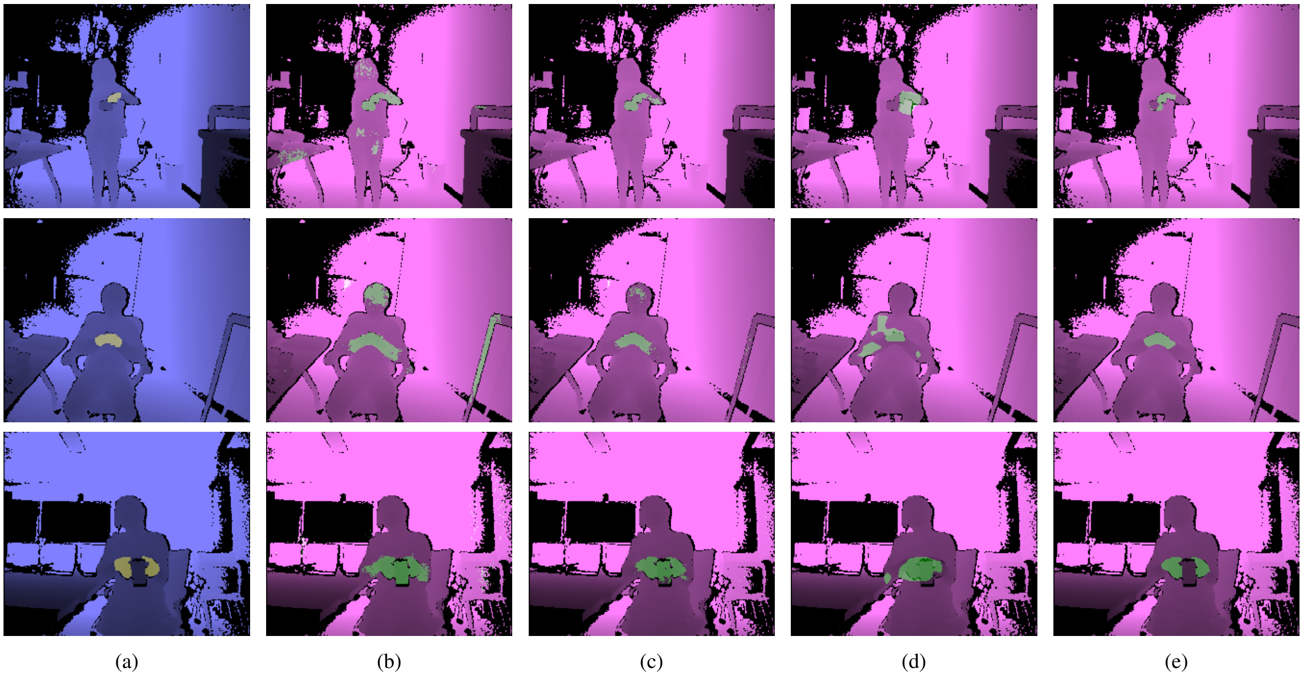
(a) (b) (c) (d) (e)

Fig. 4. Visual comparison. (a) Ground truth label. (b) Result using RDF [1, 19]. (c) Result using RDF [1, 19] with the proposed method in Section 2.2. (d) Result using FCN-8s [24, 25]. (e) Result using the proposed method. The results and ground truth are visualized using different color channels for better visualization.

comparison of results using depth maps and color images. Hence, we only compare the proposed method with the state-of-the-art methods which can process using only depth maps.

Table 1 and Fig. 4 show quantitative results and visual results. The quantitative results show that the proposed method achieves about 25% and 8% relative improvements in F_1 score comparing to the RDF-based methods [1, 19] and its combination with the proposed method in Section 2.2, respectively. Comparing to the deep learning-based methods [24, 25], the proposed method achieves about 7% lower accuracy, but processes in about 42 times shorter processing time. Thus, deep learning-based methods can not be used in real-time applications. Fig. 5 shows the comparison of methods in accuracy and efficiency. The proposed method achieves high accuracy in short processing time.

4. CONCLUSION

In this paper, we present two-stage RDF method for hand segmentation for hand-object interaction using only a depth map. The two stages consist of detecting hands and segmenting hands. The proposed method achieves high accuracy in short processing time efficiently.

5. REFERENCES

[1] J. Tompson, M. Stein, Y. Lecun, and K. Perlin, "Real-time continuous pose recovery of human hands using convolutional networks," *ACM Trans. Graph.*, vol. 33, no. 5, pp. 169:1–169:10, Sept. 2014.

[2] T. Sharp, C. Keskin, D. Robertson, J. Taylor, J. Shotton, D. Kim, C. Rhemann, I. Leichter, A. Vinnikov, Y. Wei, D. Freedman, P. Kohli, E. Krupka, A. Fitzgibbon, and S. Izadi, "Accurate, robust, and flexible real-time hand tracking," in *Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems*. 2015, CHI '15, pp. 3633–3642, ACM.

[3] C. Qian, X. Sun, Y. Wei, X. Tang, and J. Sun, "Realtime and robust hand tracking from depth," in *Computer Vision and Pattern Recognition (CVPR), 2014 IEEE Conference on*, June 2014, pp. 1106–1113.

[4] J. Romero, H. Kjellstrom, and D. Kragic, "Hands in action: real-time 3d reconstruction of hands in interaction with objects," in *Robotics and Automation (ICRA), 2010 IEEE International Conference on*, May 2010.

[5] J. Romero, H. KjellströM, C. H. Ek, and D. Kragic, "Non-parametric hand pose estimation with object context," *Image Vision Comput.*, vol. 31, no. 8, Aug. 2013.

[6] Y. Wang, J. Min, J. Zhang, Y. Liu, F. Xu, Q. Dai, and J. Chai, "Video-based hand manipulation capture through composite motion control," *ACM Trans. Graph.*, vol. 32, no. 4, pp. 43:1–43:14, July 2013.

[7] I. Oikonomidis, N. Kyriazis, and A.A. Argyros, "Full dof tracking of a hand interacting with an object by modeling occlusions and physical constraints," in *Computer Vision (ICCV), 2011 IEEE International Conference on*, Nov. 2011, pp. 2088–2095.

[8] A. A. Argyros and M. I. A. Lourakis, "Real-time tracking of multiple skin-colored objects with a possibly moving camera," in *Computer Vision - ECCV 2004*, May 2004, pp. 368–379.

[9] B. Kang, Y. Lee, and T. Nguyen, "Efficient hand articulations tracking using adaptive hand model and depth map," in *Advances in Visual Computing*, Dec. 2015.

[10] M. Cai, K.M. Kitani, and Y. Sato, "A scalable approach for understanding the visual structures of hand grasps," in *Robotics and Automation (ICRA), 2015 IEEE International Conference on*, May 2015, pp. 1360–1366.

[11] B. Kang, S. Tripathi, and T. Nguyen, "Real-time sign language fingerspelling recognition using convolutional neural networks from depth map," in *Pattern Recognition, 2015 3rd IAPR Asian Conference on*, Nov. 2015.

[12] M. J. Jones and J. M. Rehg, "Statistical color models with application to skin detection," *International Journal of Computer Vision*, vol. 46, pp. 81–96, Jan. 2002.

[13] R. Khan, A. Hanbury, J. Stttinger, and A. Bais, "Color based skin classification," *Pattern Recognition Letters*, vol. 33, pp. 157–163, Jan. 2012.

[14] C. Li and K.M. Kitani, "Pixel-level hand detection in ego-centric videos," in *Computer Vision and Pattern Recognition, 2013 IEEE Conference on*, June 2013.

[15] S. L. Phung, A. Bouzerdoum, and D. Chai, "Skin segmentation using color pixel classification: analysis and comparison," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 27, 2005.

[16] P. Kakumanu, S. Makrogiannis, and N. Bourbakis, "A survey of skin-color modeling and detection methods," *Pattern Recognition*, vol. 40, pp. 1106–1122, Mar. 2007.

[17] D. Tzionas and J. Gall, "3d object reconstruction from hand-object interactions," in *International Conference on Computer Vision (ICCV)*, Dec. 2015.

[18] R. Y. Wang and J. Popović, "Real-time hand-tracking with a color glove," *ACM Trans. Graph.*, vol. 28, 2009.

[19] J. Shotton, A. Fitzgibbon, M. Cook, T. Sharp, M. Finocchio, R. Moore, A. Kipman, and A. Blake, "Real-time human pose recognition in parts from single depth images," in *Computer Vision and Pattern Recognition (CVPR), 2011 IEEE Conference on*, June 2011.

[20] A. Criminisi and J. Shotton, "Decision forests for computer vision and medical image analysis," *Advances in Computer Vision and Pattern Recognition*, 2013.

[21] L. Breiman, "Random forests," *Machine Learning*, vol. 45, no. 1, pp. 5–32, 2001.

[22] T. Ho, "Random decision forests," in *Proceedings of the Third International Conference on Document Analysis and Recognition*, Washington, DC, USA, 1995.

[23] C. Tomasi and R. Manduchi, "Bilateral filtering for gray and color images," in *Computer Vision, Sixth International Conference on*, Jan. 1998, pp. 839–846.

[24] Jonathan Long, Evan Shelhamer, and Trevor Darrell, "Fully convolutional networks for semantic segmentation," in *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2015.

[25] E. Shelhamer, J. Long, and T. Darrell, "Fully convolutional networks for semantic segmentation," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. PP, no. 99, pp. 1–1, 2016.