Feature extraction for efficient image and video segmentation

Jakub Vojvoda, Vítězslav Beran

Brno University of Technology, Faculty of Information Technology

Abstract

The segmentation of sensory data of various domains is often crucial pre-processing step in many computer vision methods and applications. In this work, we propose a method that leverages the quantization of local feature's distributions for the depth and the temporal information. Three variants of the segmentation method is designed and evaluated reflecting various data domains: space (color and texture), temporal (motion) and depth domain. Each variant was tested on appropriate dataset showing the usability of designed method for applications like areal-image analysis, hand detection and moving-people detection. The pilot experiments shows the characteristics of the approach and computational costs of designed variants.

Keywords: color/texture segmentation, motion segmentation, RGB-D/T segmentation

1. Introduction

Image segmentation is a computer vision problem of dividing an input image into multiple regions which correspond to different objects in the image. The assumption is that individual objects and background differ in characteristic such as color, texture, shape, location or motion. Some kinds of this process can be found in applications involving object detection, classification and recognition. Importance of the problem lies in the fact that this approximation can speed up and improve next higher-level processing steps.

Existing approaches differ in many aspects, e.g. accuracy, stability, computational costs etc. and can be generally divided into threshold based, edge based, region based and methods based on clustering. Recent scientific results indicate the effectivity of reduction of data dimensionality and approximated representation in sensory data processing approaches. The local features extracted from data of various domains are statistically analyzed and quantized decreasing the computational demands in further processing of these data. The similar approach is widely known as bag-of-features based on visual vocabulary in image-retrieval or image-classification domain.

Depth data segmentation become one of the hot topics in robotics field, mostly driven by a new era of cheap depth sensors (Kinect 1, resp. 2, ASUS xTion Pro etc.) available on the market in last years. The sensors provide the perception systems with the depth information of pixels in the close environment. The segmentation of depth data is again important pre-processing stage for many applications like: object classification, hand detection, scene interpretation etc.

In this paper, existing methods for segmentation are discussed in more details in section 2. Section 3 presents the proposed method for data segmentation and section 4 describes features and metrics proposed for different data domains. Results of the pilot experiments showing methods' characteristics and computational cost are presented in section 5.

2. Related works

Numerous methods for image and video segmentation exist. However, the idea and application of depth information in computer vision applications have become popular recently.

Blas et al. [1] convert input image into CIELAB color space and represent a texture using the local surroundings, which is similar to the Local Binary Patterns (LBP) method. They cluster the data to textons using k-means algorithm. Thanks to the calculation of histograms over textons, they are able to represent the "mixed terrain". Finally, the histogram profiles are merged using the Earth Mover Distance (EMD). The method achieves best results using a 3x3 window for extracting texture information and 32×32 window for extracting histograms.

Greggio et al. [2] described unsupervised algorithm that is capable of learning finite mixture model from multivariate data online. The method is based on the expectation-maximization algorithm to Gaussian Mixtures. The segmentation starts with only one mixture component and progressively adapts the mixture by adding new components when necessary. There are defined split and stopping criteria.

Taylor and Cowley [3] represent pixels of the input image by simple color descriptors. Given this set of feature vectors, the segmentation employs a set of randomly chosen splitting planes. The planes are used for hashing each region to n-bit code. After finding the local maximum (cluster centers) in the set of the hash codes, the vectors are assigned to the closest region based on Hamming distance. The described method achieves good results and low computation time.

The segmentation presented by Chamorro-Martínez et al. [4] is based on a growing region algorithm. The region is used as

Email addresses: jakub.vojvoda@gmail.com (Jakub Vojvoda), beranv@fit.vutbr.cz (Vítězslav Beran)

a fuzzy subset of pixels, where every pixel has a membership degree to that region. The authors use the HSI color space and membership for the fuzzy region is based on distance.

Rao et al. [5] described a method of segmentation, where the authors work with CIELAB color model. They use texture descriptors based on a pixel neighborhood. The vectors are coded based on minimal length of descriptors and the boundaries are coded using the Freeman chain code. The segmentation is then based on a hierarchical and iterative region-merging process.

Many works that deal with the depth data segmentation task mainly addressing the requirements for stability and low computational demands. Pulli and Pietikäinen [6] apply normal decomposition in their approach. They explore various techniques of range data normal estimation (comparing their performance and accuracy on clean as well as noisy datasets). The techniques include quadratic surface least squares or LSQ planar fitting. A least-trimmed-squares method is utilized for comparison. Hulik et al. [7] used similar approach and combined it with the improved tile-RANSAC approach of Ying Yang and Förstner [8] and introduced plane-prediction that improves the efficiency of the previous methods keeping the methods' accuracy. An alternative approach to plane detection in point, presented by Borrmann et al. [9], is based on 3D Hough trans-form. Dube and Zell [10] also employ randomized Hough transform for real-time plane extraction.

The goal of our work is to leverage the quantization step to computed distribution of local data features for the data segmentation tasks. We present the general method and propose its application for three data domains: space (color and texture), temporal (motion) and depth domain. We tested the methods' variants and report the characteristics of the approach and computational costs on applications like areal-image analysis, hand detection and moving-people detection.

The other goal of this work is to use presented segmentation algorithm for real time processing. Therefore, it was necessary to improve the execution time of the most consuming part of the method, while preserving its accuracy.

3. Segmentation algorithm

The general idea of segmentation algorithm is to extract and quantize the local feature vectors. We call them also textons [1] and merge them using their local distribution analysis into larger segments. Steps of segmentation algorithm are shown in Figure 1.

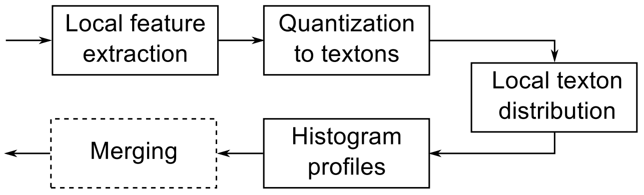

Figure 1: Segmentation pipeline

The **local feature extraction** method varies for different data domains and is discussed in details in the next section. In general, the feature vectors for each pixel are extracted using window and quantized into textons. The extraction is done by square window and the quantization is based on **clustering** of extracted local features. The obtained cluster centers are used as textons where its number depends on the number of clusters obtained from a clustering algorithm. The labels, which corresponds to indexes of this clusters/textons, are then assigned to each pixel of image based on Euclidean distance in the feature space. For this purpose we use k-means++ [11] algorithm where each pixel is assigned to the closest texton.

The matrix of indexes from previous part serves for describing the local image parts by histograms of textons. The calculation of the **histogram profiles** is optimized by constructing integral images for each of the texton cluster, so the computation takes only four addition operations for every texton. The histograms are extracted by window of the adjustable size where the window center is a pixel, for which this histogram is computed. The size is significantly larger than size of the window used for the local feature extraction and implicitly we work with the window of size 31 × 31 pixels. By this extraction it is possible to capture a density of texture elements.

The histogram features are statistically analyzed again, but in contrast to previous quantization process, we aim to use as low reasonable cluster amount as possible. This process results in local histogram profiles. The number of extracted histogram profiles depends again on number of clusters initially chosen in the k-means++ [11] clustering algorithm. Generally, this process results in over-segmented image where the over-segmentation depends on number of extracted histogram profiles.

Merging is the last and the optional step of segmentation process where is an effort to improve results by merging these over-segmented areas. We experimented with various ways (Euclidean distance, Earth Movers Distance, Histogram correlation, Bhattacharyya distance and others) that could achieve better results. The main problem was to find a threshold, which would lead to the ideal merging.

Based on experiments, we chose a method based on distance matrix as the most appropriate one. This method differs from method used in [1] which is based on the Earth Movers Distance and is computationally more expensive. The matrix is used to merge similar clusters and the threshold is computed by approximation with the Cauchy-Lorentz distribution. It was determined by analyzing distances and using polynomial approximation:

$$threshold = 0.0311 + 0.1723 \cdot \exp^{-4 \cdot \log 2 \cdot \left(\frac{max - 0.6015}{0.1108}\right)^2} \tag{1}$$

where max is a maximum value of the distance matrix. The distance matrix is $n \times n$ hollow matrix where n is number of histogram profiles.

We use the Euclidean metric to compute distance of two histogram profiles. While there are more than two profiles, two profiles with the lowest distances below this threshold are merged to one and its number and the size of the distance matrix is reduced. The distances between merged and other profiles are updated as follow

$$d(A \cap B, C) = min(d(A, C), d(B, C)) \tag{2}$$

where d is distance, A, B, C are histogram profiles and $A \cap B$ is result of merging profiles A and B.

4. Features extraction variants

The type of segmentation differs in feature extraction procedure reflecting the characteristics of particular data domains. Three feature types have been proposed for each of targeted data domain. Firstly, we describe a texture based segmentation for color images that was introduced by [1] and that we used to design our method variants for other data domains: motion based segmentation for image sequences and RGB-D/T segmentation combining all color, depth and motion information.

4.1. Texture based segmentation

The Texture based segmentation in our work is used as fundamental approach [1] for evaluation and comparison with our newly proposed variants for depth and temporal data. The input image is converted from the RGB color space to the CIELAB model first. This model defines the brightness of the color $L \in \langle 0, 100 \rangle$, the color position between red and green $a \in \langle -127, 127 \rangle$ and the color position between yellow and blue $b \in \langle -127, 127 \rangle$. This values are used as parts of the feature vector that is extracted via window of size 3×3 pixels.

Each central pixel is represented by given L, a, b values of the CIELAB color model and the pixel neighborhood is represented as weighted difference between L values of surrounding pixels and center pixel. It means that each pixel is described by 11-dimensional feature vector.

$$\begin{array}{|c|c|c|} \hline L_1 & L_2 & L_3 \\ \hline L_8 & \boldsymbol{L_c} & L_4 \\ \hline L_7 & L_6 & L_5 \\ \hline \end{array} \quad \rightarrow \quad p(i, j) = \begin{bmatrix} W_1 \cdot L_c \\ W_2 \cdot a_c \\ W_2 \cdot b_c \\ W_3 \cdot (L_1 - L_c) \\ \vdots \\ W_3 \cdot (L_8 - L_c) \end{bmatrix} \tag{3}$$

The equation 3 presents the process of local feature extraction (same as in [1]) where weight W_1 is equal to $\frac{1}{2}$, W_2 to 1 and W_3 is equal to $\frac{1}{2}$. The color of center pixel is defined by values L_c, a_c and b_c. The Euclidean norm in 11-dimensional space is then used as the distance between the feature vectors. The norm is defined as follow

$$\begin{aligned} d_L &= (W_1 \cdot (L_{ci} - L_{cj}))^2 \\ d_{ab} &= (W_2 \cdot (a_{ci} - a_{cj}))^2 + (W_2 \cdot (b_{ci} - b_{cj}))^2 \\ d_s &= \sum_{k=1}^{N} (W_3 \cdot (L_{ki} - L_{ci} - L_{kj} + L_{cj}))^2 \\ D_{texture} &= \sqrt{d_L + d_a + d_b + d_s} \end{aligned} \tag{4}$$

where i and j are indexes of two vectors, k is index of surrounding pixels and N value depends on the size of the neighborhood (equals 8 for 3×3 window).

4.2. Motion segmentation

In this type of segmentation, the features are extracted by computation of a dense optical flow using the Gunnar Farneback algorithm [12]. This two-frame motion estimation algorithm computes flow by approximating frames by quadratic polynomials and constructing global shift. It uses the Gaussian averaging window of size 2×2 pixels with $\sigma = 1.1$. We define 5 iterations and 5×5 neighborhood to find polynomial expansion in each pixel.

The output of this process is a flow between two frames, from which magnitude and angle is obtained, after converting into polar coordinates. Each pixel of the image is described by 2-dimensional feature vector defined as

$$p(i, j) = \begin{bmatrix} R_c \\ \\ \alpha_c \end{bmatrix} \tag{5}$$

where R_c is magnitude and α_c is angle of the flow in center pixel. As the norm between two feature vectors, 2-dimension Euclidean distance is defined as

$$\begin{aligned} d_m &= (R_{ci} - R_{cj})^2 + (\alpha_{ci} - \alpha_{cj})^2 \\ D_{motion} &= \sqrt{d_m} \end{aligned} \tag{6}$$

where i and j are indexes of the corresponding feature vectors.

4.3. RGB-D/T segmentation

Finally, we designed the operator for fusion of all, color, depth and motion information. Firstly, a texture information is extracted from the input color image. The texture is described using L_c, a_c and b_C color value of the center pixel and difference between this L_c value and L value of surrounding pixels (see section 4.1) via 3×3 kernel. Another used features are magnitude and angle, obtained by the Gunnar Farneback optical flow (see section 4.2) and depth information normalized to $\langle 0, 1 \rangle$.

$$p(i, j) = \begin{bmatrix} W_1 \cdot L_c \\ \vdots \\ W_3 \cdot (L_8 - L_c) \\ z_c \\ R_c \\ \alpha_c \end{bmatrix} \tag{7}$$

It means that each pixel is described by 14-dimensional vector, as can be seen at equation 7, where first 11 values are same as for texture segmentation, $z_c \in \langle 0, 1 \rangle$ is depth value for center pixel, R_c is magnitude and α_c is angle obtained from optical flow in this position. The norm between two feature vectors is defined as follow

$$\begin{aligned} d_z &= (z_{ci} - z_{cj})^2 \\ D_{RGBDT} &= \sqrt{d_L + d_{ab} + d_s + d_z + d_m} \end{aligned} \tag{8}$$

where i and j are indexes of two feature vectors, d_L, d_{ab} and d_s are defined in equation 4 and d_m in equation 6.

5. Pilot experiments

Each segmentation method variant has been evaluated on specific dataset to reflect the targeted application of the proposed variant. The pilot experiments were focused to show the differences of segmentation results and to measure the additional computational cost of extended segmentation procedure.

5.1. Texture-based segmentation

Results of the fundamental texture segmentation was collected and evaluated using the Berkeley Segmentation Dataset [13]. The dataset contains 500 images divided into training, testing images and image for evaluation. The evaluation process was done on all of them using the EvaluationSegmentation[1] evaluation protocol (software support and ground truth data).

Table 1 presents the results of evaluation. The metrics are defined as follow, where TP is true positive, TN true negative, FP false positive and FN false negative rate.

- Precision
$$p = \frac{TP}{TP + FP} \tag{9}$$
- Recall (Sensitivity)
$$r = \frac{TP}{TP + FN} \tag{10}$$
- F-measure
$$fm = \frac{2pr}{p + r} \tag{11}$$
- Specificity
$$s = \frac{TN}{TPN + FP} \tag{12}$$
- Accuracy
$$a = \frac{TP + TN}{TP + FP + TN + FN} \tag{13}$$
- Fallout
$$f_0 = \frac{FP}{FP + TN} \tag{14}$$

Method	Precision	Recall	F-measure	Specificity	Accuracy	Fallout
Texture-based segmentation	0.526	0.689	0.571	0.633	0.641	0.366

Table 1: Results of texture-based segmentation accuracy on the Berkeley Segmentation Dataset.

The **areal image segmentation** task was selected as the representative application of texture based segmentation method. We used the Aerial Image Segmentation Dataset [14] for the pilot experiments. In contrast with other types of segmentation, a significantly smaller window for histogram profiles extraction had to be used. We used the window of size 11×11 pixels, which preserves more details in the image. Then we used 11-dimensional local feature vectors, 16 textons and 8 histogram profiles without final merging. The results in Figure 2 shows the ability of the approach to adapt to local image contrasts and stability for various texture areas.

[1]`www.visceral.eu/resources/evaluatesegmentation-software`

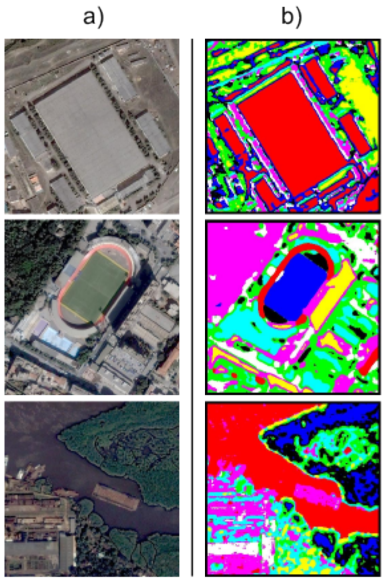

Figure 2: A result of areal image segmentation a) color input images b) texture segmentation

5.2. Motion-based and Depth-based segmentation

We used the **moving people detection** task to study and analyze the results of motion and RGB-D/T segmentation. For this purpose was used the RGB-D People Dataset [15], which contains sequences of images with corresponding depth data. The data contains mostly walking or standing people seen from different orientations and with different levels of occlusions.

In case of motion segmentation, we used 2-dimensional local feature vectors for each pixel (see section 4.2), 16 textons, 9 histogram profiles extracted by 31×31 window with final merging. The result of the motion segmentation at Figure 3c shows than even with using only 2-dimensional feature vectors it is possible to segment motion accurately.

Used parameter values of RGB-D/T segmentation differ only in size of local feature vectors and merging. In this task 14-dimensional vectors (see section 4.3) and no merging was used. An example of the result can be seen in figure 3d where can be seen that by using color, motion and depth information simultaneously, the accuracy of the method is increased. In case of moving objects, the shape and the borders of the object are preserved and accurate.

The Hand-Hand Interaction data set [16], which contains color and depth image sequences of interactions, was used to experiment and analyze the characteristics of texture- and RGB-D/T segmentation in the **hand detection** task. The results can be seen in figure 4.

In this part of experiments, we set parameter values to 16 textons, 9 histogram profiles and histogram extraction window of size 31x31 pixel using final merging. For texture segmentation 11-dimensional and for RGB-D/T segmentation 14-dimensional local feature vectors was used (see section 4).

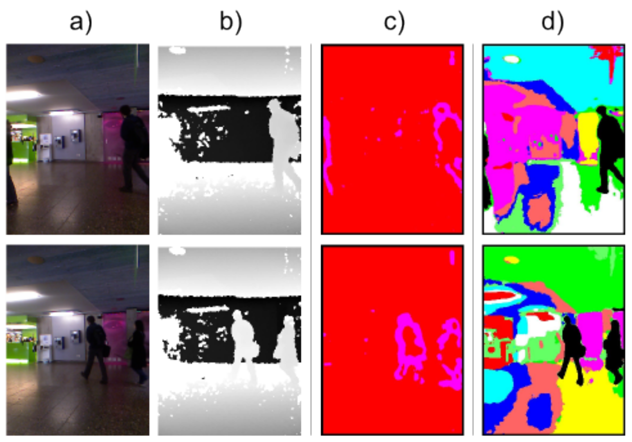

Figure 3: a) color image b) depth image c) motion segmentation d) RGB-D/T segmentation

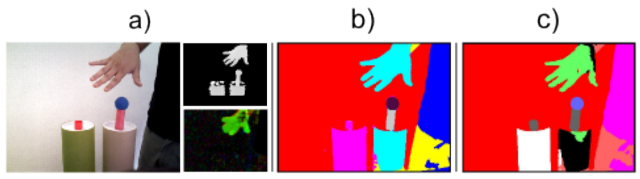

Figure 4: a) color image, depth image and motion visualization b) texture segmentation c) RGB-D/T segmentation

5.3. *Computational cost*

In our pilot experiments, we were interested in computational demands of particular segmentation pipeline parts. The main goal of these experiments is to decrease the computational cost of the particular parts and make the pipeline suitable for the real time processing. For this purpose we used basic laptop Intel Core i5-2450M (2.50 GHz) 4 GB.

Table 2 presents an average percentage of total time consumed by segmentation. The time was measured on the RGB-D People Dataset [15], which contains 480x640 pixel images. Obviously, the results shows that the most time consuming part is clustering.

	Texture Segmentation	Motion segmentation	RGB-D/T segmentation
Local feature extraction	9.85%	23.25%	17.55%
Quantization to textons	48.47%	40.15%	43.31%
Local texton distribution	22.26%	24.74%	16.05%
Histogram profiles	19.21%	11.76%	22.82%
Merging	0.21%	0.10%	0.27%
	0.89 fps	0.96 fps	0.65 fps

Table 2: The percentage of time required for different parts of segmentation and average number of processed frames per second (fps)

We further modified our feature and histogram profiles extraction steps to experiment with selecting only subsets of data during this process. While preserving the accuracy of the segmentation, we decrease the amount of selected pixels that results in lower computational demands.

The execution time was measured on the images of size 321×481 pixels. The images were obtained from the Berkeley Segmentation Dataset [13].

Extraction step	[1,1]	[1,2]	[2,1]	[2,2]
Computation time in ms	582.58	431.87	295.24	142.08
Frame rate in fps	1.72	2.32	3.39	7.04

Table 3: Computational cost in milliseconds (ms) and number of processed frames per second (fps)

The table 3 shows our improved results where the first row defines a value of the feature extraction offset M and value of the histogram profile extraction offset N. The offset is defined in both axis and means that every Nth feature and every Mth histogram will be extracted. The [1,1] column defines extraction without improvement. In the second row, the average computation time of segmentation is showed and in the third row frame rate in fps is presented. The advantage of this approach is that the algorithm with this reduction is suitable for real time segmentation while the accuracy of the method is preserved.

6. Conclusion

In this work, we presented the method that was designed to decrease the segmentation pre-processing by leveraging the local feature dimensionality reduction by the quantization approach. Following the efficient selected segmentation methods, we proposed and tested the modification of the method for depth and temporal data. The important step of our method is the quantization of the local distribution of extracted features to decrease the computational costs of the whole segmentation process.

The proposed approach was implemented and evaluated on datasets of three application domains: areal-image segmentation, hand detection and pedestrian detection.

Pilot experiments showed that described segmentation pipeline is suitable for more than texture based segmentation. Described algorithm is capable of segmenting mixture patterns what is usable also for motion and RGB-D/T segmentation. Also our modification method using the sampling of the local features during extraction step shows the possibility to improve the computational demands of the method while preserving the method accuracy.

References

[1] Blas MR, Agrawal M, et al. Fast color/texture segmentation for outdoor robots. In: IROS. 2008, p. 4078–85.

[2] Greggio N, Bernardino A, Santos-Victor J. Image segmentation for robots: Fast self-adapting gaussian mixture model. In: Proceedings of the 7th International Conference on Image Analysis and Recognition -

Volume Part I. ICIAR'10; Berlin, Heidelberg: Springer-Verlag; 2010, p. 105–16.

[3] Taylor CJ, Cowley A. Fast segmentation via randomized hashing. 2009.

[4] Chamorro-Martínez J, Sánchez D, Prados-Suarez B. A fuzzy colour image segmentation applied to robot vision. In: VII Online World Conference on Soft Computing in Industrial Applications. 2002,.

[5] Rao SR, Mobahi H, Yang A, Sastry S, Ma Y. Natural image segmentation with adaptive texture and boundary encoding. In: Computer Vision - ACCV 2009; vol. 5994 of *Lecture Notes in Computer Science*. Springer Berlin Heidelberg. ISBN 978-3-642-12306-1; 2010, p. 135–46.

[6] Pulli K, Pietikäinen M. Range image segmentation based on decomposition of surface normals. In: Proceedings of the Scandinavian conference on image analysis; vol. 2. Citeseer; 1993, p. 893–.

[7] Hulík R, Beran V, Španěl M, Kršek P, Smrž P. Fast and accurate plane segmentation in depth maps for indoor scenes. In: IEEE/RSJ International Conference on Intelligent Robots and Systems. Department of Computer Graphics and Multimedia FIT BUT. ISBN 978-1-4673-1737-5; 2012, p. 1–6.

[8] Yang MY, Förstner W. Plane detection in point cloud data. In: Proceedings of the 2nd int conf on machine control guidance, Bonn; vol. 1. 2010, p. 95–104.

[9] Borrmann D, Elseberg J, Lingemann K, Nüchter A. The 3d hough transform for plane detection in point clouds: A review and a new accumulator design. 3D Research 2011;2(2):1–13.

[10] Dube D, Zell A. Real-time plane extraction from depth images with the randomized hough transform. In: Computer Vision Workshops (ICCV Workshops), 2011 IEEE International Conference on. IEEE; 2011, p. 1084–91.

[11] Arthur D, Vassilvitskii S. K-means++: The advantages of careful seeding. In: Proceedings of the Eighteenth Annual ACM-SIAM Symposium on Discrete Algorithms. SODA '07; Philadelphia, PA, USA: Society for Industrial and Applied Mathematics; 2007, p. 1027–35.

[12] Farnebäck G. Two-frame motion estimation based on polynomial expansion. In: Proceedings of the 13th Scandinavian Conference on Image Analysis. SCIA'03; Berlin, Heidelberg: Springer-Verlag. ISBN 3-540-40601-8; 2003, p. 363–70.

[13] Martin D, Fowlkes C, Tal D, Malik J. A database of human segmented natural images and its application to evaluating segmentation algorithms and measuring ecological statistics. In: Proc. 8th Int'l Conf. Computer Vision; vol. 2. 2001, p. 416–23.

[14] Yuan J, Gleason S, Cheriyadat A. Systematic benchmarking of aerial image segmentation. Geoscience and Remote Sensing Letters, IEEE 2013;10(6):1527–31.

[15] Spinello L, Arras KO. People detection in RGB-D data. In: Proc. of The International Conference on Intelligent Robots and Systems (IROS). 2011,.

[16] Tzionas D, Ballan L, Srikantha A, Aponte P, Pollefeys M, Gall J. Capturing hands in action using discriminative salient points and physics simulation. CoRR 2015;.