Accepted in IEEE Int. Conference on Image Processing (ICIP 2018), 2018

MOTION OCCLUSIONS FOR AUTOMATIC GENERATION OF RELATIVE DEPTH MAPS

Louiza Oudni, Carlos Vázquez, Stéphane Coulombe

Department of Software and IT Engineering
École de technologie supérieure
Montreal, Quebec, Canada

ABSTRACT

Recovering of the depth structure of a scene from monocular video content provides an important advantage in applications such as AR (placing and removing of objects) or 3D-TV and 3D cinema (2D-to-3D video conversion). In this paper, we present an automatic method to generate relative depth maps from monocular video sequences. It relies on the dynamic occlusion depth cue to recover the depth order of objects in the scene. The forward and backward motion analysis between each two consecutive frames allows the calculation of their dynamic occlusions. We estimate the motion using a modified version of the EpicFlow. Our modifications to this optical flow method made it coherent in forward-backward directions without compromising its performance. Thanks to this new feature, occlusions are simpler to calculate than the approaches used in the relevant literature. The obtained occlusions allow order deduction of the objects contained in the image. These objects are obtained using a segmentation approach which considers both color and motion. Ours results show a small improvement to the quality of the optical flow while adding the forward/backward coherence. With respect to the depth ordering our approach obtains slightly better results than the reference method while removing a computationally costly step from the processing.

Index Terms— relative depth map, occlusions, depth ordering, segmentation

1. INTRODUCTION

Retrieving depth information from 2D content is possible by exploiting depth cues present in 2D images, such as linear perspectives [1], motion parallax [2], static occlusions [3] and motion occlusions. The motion occlusion cues are reliable and present in all scenes types and at all distances [4]. For that reason, many approaches use them to retrieve depth ordering information. The approach proposed in [5] treats the case of static scenes containing a single moving object, where the object is partially occluded by scene parts. The case of static scenes containing multiple moving objects is treated in [6]. Other approaches do not consider restrictions on camera motion [7, 8, 9]. The method followed in [9] has the advantage of segmenting the scene by jointly considering color and motion information, but a parametric region-based optical flow has to be computed in order to calculate motion occlusions.

In our proposed approach, the idea is to compute forward/backward coherent optical flow in order to simplify the computation of occlusions' relations. Once calculated, the occlusions are used to generate segmentation and define the depth ordering of objects, in an approach similar to [9]. In a nutshell, the color and motion information are used to represent the image with a hierarchical structure called a binary partition tree [10]. After that, this binary tree is pruned according to the occlusion relations. The pruning step results in segmentation of the scene. The last step of the method consists in ordering regions of this segmentation according to the estimated depth relationships. This paper is organized as follows. In section 2, we present the proposed approach. We expose the obtained results and discuss it in section 3, followed by a conclusion in section 4.

2. PROPOSED APPROACH

We propose to estimate relative depth order using motion occlusion cues. The proposed approach consists of four main steps: the estimation of coherent forward/backward optical flow, the computation of occlusions based on the results of the optical flow, a partitioning of the image using color and motion information and lastly an assignment of depth order to each region in the partition by exploiting the occlusion information. Figure 1 illustrates these steps applied to the sequence *chair1* of the CMU dataset [11].

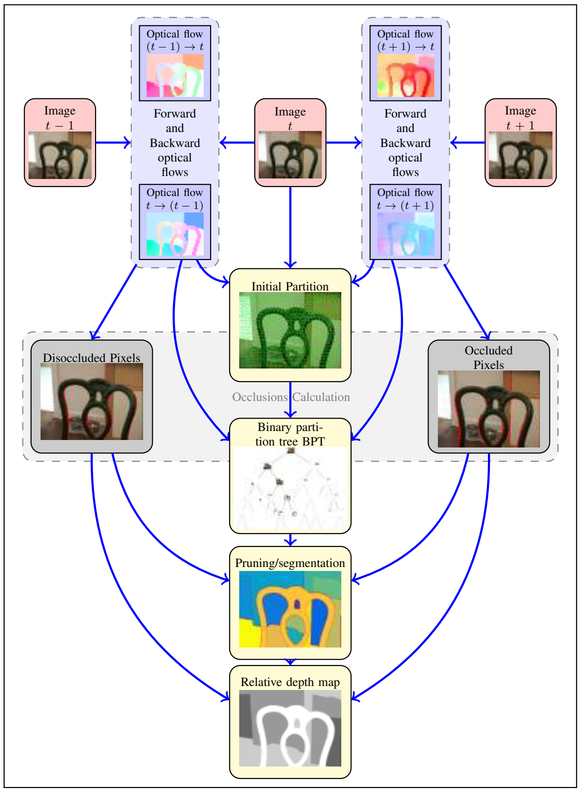
Optical flow $(t-1) \to t$
Image $t-1$
Forward and Backward optical flows
Optical flow $t \to (t-1)$
Image t
Optical flow $(t+1) \to t$
Forward and Backward optical flows
Optical flow $t \to (t+1)$
Image $t+1$
Initial Partition
Disoccluded Pixels
Occluded Pixels
Occlusions Calculation
Binary partition tree BPT
Pruning/segmentation
Relative depth map

Fig. 1: Proposed depth ordering estimation steps

Paper presented at 2018 25th IEEE International Conference on Image Processing (ICIP), Athens, Greece, 7-10 Oct. 2018
https://doi.org/10.1109/ICIP.2018.8451417

2.1. Forward/backward coherent optical flow

Motion fields are estimated using a modified version of the EpicFlow method [12] because, contrary to other methods, EpicFlow explicitly preserves the edges, which are essential for occlusion detection. The main steps of the EpicFlow method consist in interpolating matched points between the two images according to a geodesic distance in order to keep the edges, followed by a step of energy minimization to obtain the final flow estimation. The DeepMatching method [13] allows to generate the matching, the SED method [14] is used to compute the edges and Voronoï regions are computed to approximate the geodesic distance. In this paper, we modified the original EpicFlow in order to ensure the forward/backward coherence. The idea is to jointly calculate the backward and forward optical flows by ensuring the coherence of the used matched points and of the calculated regions. Figure 2 illustrates these modifications.

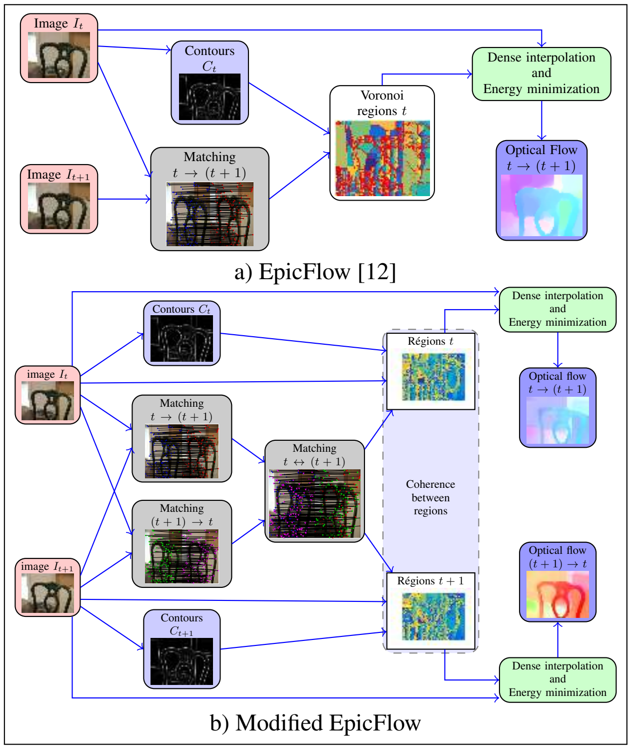
Image I_t
Contours C_t
Dense interpolation and Energy minimization
Voronoi regions t
Image I_{t+1}
Matching $t \to (t+1)$
Optical Flow $t \to (t+1)$
Contours C_t
Dense interpolation and Energy minimization
Régions t
image I_t
Optical flow $t \to (t+1)$
Matching $t \to (t+1)$
Matching $t \leftrightarrow (t+1)$
Coherence between regions
Matching $(t+1) \to t$
image I_{t+1}
Optical flow $(t+1) \to t$
Régions $t+1$
Contours C_{t+1}
Dense interpolation and Energy minimization

a) EpicFlow [12]

b) Modified EpicFlow

Fig. 2: EpicFlow and modified EpicFlow steps

We can observe that the DeepMatching method doesn't provide the same corresponding points when matching an image I_t with I_{t+1} or matching I_{t+1} with I_t. The first proposed modification consists in applying the matching procedure in both directions and merging the results into a unique set of matching points, so we have coherent entry points to compute both forward and backward optical flows. The second modification changes the computation of Voronoï regions. In addition to contour information, the computation of the modified regions uses the color information expressed in the CIELAB color space as shown in Algorithm 1.

The distance used to calculate the regions is:

$$D_r(p_k, p) = D_{lab}(p_k, p) + D_{contour}(p_k, p)$$

```
Data: $N$ matching points $p_k = (x_k, y_k)$,
      $I$ image in CIELAB color space, $C$ contour image
Result: $R$ Image of regions' labels
begin
    $D$ array with same size as the image,
    initialized to maximum value
    $k \leftarrow 1$
    while $k \leq N$ do
        $x_{start} \leftarrow x_k - offset$
        $x_{end} \leftarrow x_k + offset$
        $y_{start} \leftarrow y_k - offset$
        $y_{end} \leftarrow y_k + offset$
        $x \leftarrow x_{start}$
        while $x \leq x_{end}$ do
            $y \leftarrow y_{start}$
            while $y \leq y_{end}$ do
                $d \leftarrow D_r((x, y), (x_k, y_k))$
                if $d < D(x, y)$ then
                    $D(x, y) \leftarrow d$
                    $R(x, y) \leftarrow k$
                end
                $y \leftarrow y + 1$
            end
            $x \leftarrow x + 1$
        end
        $k \leftarrow k + 1$
    end
    return $R$
end
```

Algorithm 1: Region calculation for optical flow estimation

where:

$$\begin{aligned} D_{lab}(p_k, p) = (I_l(p_k) - I_l(p))^2 \\ + (I_a(p_k) - I_a(p))^2 \\ + (I_b(p_k) - I_b(p))^2 \end{aligned}$$

I_l, I_a, I_b are respectively the three color components expressed in the CIELAB color space. And

$$D_{contour}(p_k, p) = \left[\sum_{p_i \in L} C(p_i) \right]^2$$

L is the segment that joins pixel p to pixel p_k, and C is the contour image of I computed by the SED algorithm [14].

2.2. Occlusion Computation

We propose to estimate the occlusions between two images I_t and I_{t+1}, using the forward $\mathrm{w}_{t \to (t+1)}$ and the backward $\mathrm{w}_{(t+1) \to t}$ optical flows calculated using the modified EpicFlow. Figure 3 summarizes the occluding pixels detection procedure.

Let us define $\mathcal{L} = (p_u, p_o)$ a set of pixel pairs, where $p_u \in I_t$ is occluded in I_{t+1} by the pixel p_o. We know that an occluded pixel does not have a corresponding pixel in the next image, so the optical flow can't be coherent at its position, despite the fact that the calculated motion tries to ensure backward/forward coherence. Then, a pixel p is potentially occluded if: $p \neq p_{ret}$, where:

$$\begin{cases} p_{t+1} = p + \mathrm{w}_{t \to (t+1)}(p) \\ p_{ret} = p_{t+1} + \mathrm{w}_{(t+1) \to t}(p_{t+1}) \end{cases} \tag{1}$$

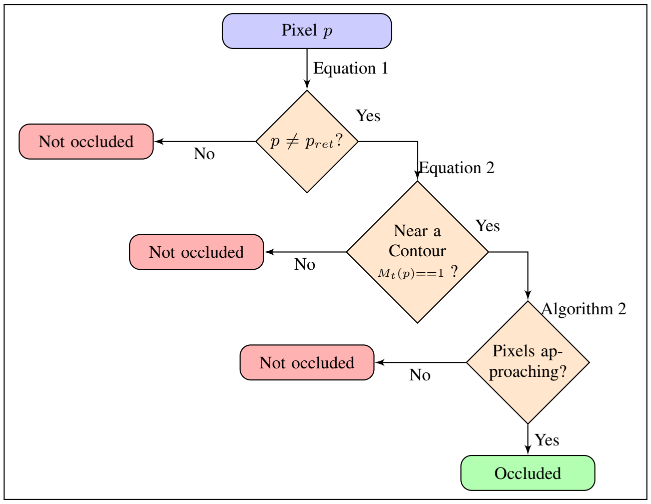

Fig. 3: Occluded pixels detection flowchart

Let p_1 be a pixel that respects this condition. We also know that occluded pixels are in general near color edges. So, we use C_t, the contour of image I_t, previously estimated by the SED method, to check if p_1 verifies the condition: $M_t(p_1) == 1$. $M_t(p_1)$ is a binary image obtained by applying the morphological image processing operation of dilation on C_t^* using a 3×3 neighborhood, with:

$$C_t^*(p) = \begin{cases} 1 & \textbf{if } C_t(p) > C_{th} \\ 0 & \textbf{otherwise} \end{cases} \tag{2}$$

C_{th} is empirically fixed to 0.15. Let p_2 be a pixel verifying this condition. In order to be sure that p_2 is an occluded pixel instead of a newly exposed one, we check if its neighboring pixels, moved by $\mathrm{w}_{t\to(t+1)}$, are coming close to p_{t+1}. The considered neighboring pixels are those in the normal direction to the contour at the position of p_2. N_n pixels are considered on each side of p_2 according to that direction. We consider the pixel occluded if at least N_t neighboring pixels (out of N_n), coming from the same side of the contour and moved by the forward $\mathrm{w}_{t\to(t+1)}$ motion, came closer to p_{t+1}. For this study, $N_n = 10$ and $N_t = 8$. Algorithm 2 is used to perform this test.

Let p_c be a detected occluded pixel, its occluding pixel p_o is estimated as follows:

$$p_o = p_c + \mathrm{w}_{t\to(t+1)}(p_c) + \mathrm{w}_{(t+1)\to t}(p_c + \mathrm{w}_{t\to(t+1)}(p_c))$$

2.3. Initial Partition

Segmentation at pixel level is costly in terms of computational complexity. Therefore, and for simplicity reasons, we start with an initial partition. In order to ensure that the initial partition has minimal impact on the segmentation, we perform it by calculating the intersection of the superpixels calculated from both, the colour and motion images. We used the SLIC algorithm [15] to recover the superpixels in our study.

2.4. Segmentation and depth ordering

```
Data: $p_2 = (i, j)$ potentially occluded pixel, $\alpha$ gradient
      direction $C_t$ en $p_2$,$\mathrm{w}_{t\to(t+1)}$
Result: $occ = true$ if $p_2$ is occluded; else $occ = false$
begin
    $occ \leftarrow false$
    $(ii, jj) \leftarrow (i, j) + \mathrm{w}_{t\to(t+1)}(i, j)$
    for $s \in \{1, -1\}$ do
        $N \leftarrow 0$
        $k \leftarrow 1$
        for $k \leq N_n$ do
            $i_k \leftarrow round(i + k \cdot s \cdot \sin(\alpha))$
            $j_k \leftarrow round(j + k \cdot s \cdot \cos(\alpha))$
            $(ii_k, jj_k) \leftarrow (i_k, j_k) + \mathrm{w}_{t\to(t+1)}(i_k, j_k)$
            $d_t \leftarrow dist((i, j), (i_k, j_k))$
            $d_{t+1} \leftarrow dist((ii, jj), (ii_k, jj_k))$
            if $d_{t+1} < d_t$ then
                $N \leftarrow N + 1$
            end
        end
        if $N > N_t$ then
            $occ \leftarrow true$
        end
    end
    return $occ$
end
```

Algorithm 2: Distinction between occluded and newly exposed pixels

After the initial partition computation, the image is represented as a binary partition tree (BPT) [10] as in [9]. The BPT is formed by iteratively merging the most similar adjacent regions (R_g and R_d) in a region R_i until all regions are merged into one. We prune this BPT following these steps: we visit the BPT in a bottom-up fashion and decide for each region R_i if it's better for the final segmentation to include its two children merged or separated. A node is pruned if:

$$E_{t-1}(R_i) \leq N_O \quad \text{and} \quad E_{t+1}(R_i) \leq N_O$$

$$E_q(R_i) = \sum_{(p_c, p_o) \in \mathcal{L}_q} \Gamma(p_c, p_o)$$

where $q = t \pm 1$, $\mathcal{L}_q$ is a set of occluded/occluding pixel pairs between I_t and I_{t+1} as calculated in section 2.2

$$\Gamma(p_c, p_o) = \begin{cases} 1 & \textbf{if} \quad (p_c \in R_g \text{ and } p_o \in R_d) \text{ or } (p_c \in R_d \text{ and } p_o \in R_g) \\ 0 & \textbf{otherwise} \end{cases}$$

N_O is fixed to 30 in this work. The leafs of the pruned BPT, is the final partition segmentation. The depth ordering is then deduced using the resulting segmentation and occlusion information as proposed by [9].

3. RESULTS AND DISCUSSIONS

Results are presented in two parts. The first part evaluates the proposed optical flow estimation in section 2.1 The second part presents the results of depth order estimation.

3.1. Optical flow estimation

We apply the modified EpicFlow on the 8 sequences of the Middelbury dataset [16] for which the ground-truth flow is publicly available. The used metrics are the End Point Error (EPE) and the Angular Error (AE). Table 1 presents the results obtained by

	NW Interpolation		LA Interpolation	
	Modified EpicFlow	EpicFlow	Modified EpicFlow	EpicFlow
EPE	0.2643	0.3232	0.2829	0.3517
AE	2.9821	3.2675	3.0351	3.2665

Table 1: EpicFlow and modified EpicFlow results on publicly available ground-truth Middelbury dataset

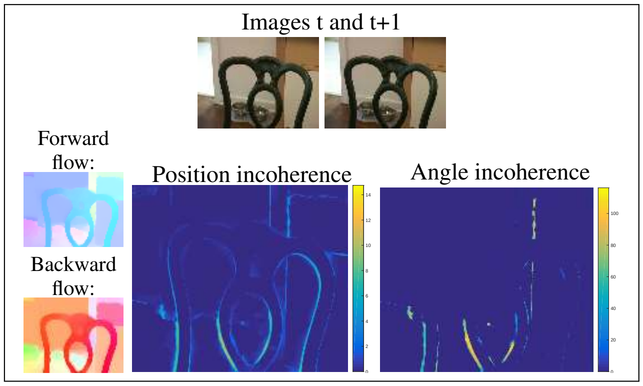

Fig. 4: Forward/Backward modified EpicFlow coherence

the EpicFlow [12] and modified EpicFlow, on the 8 sequences, with the two interpolation options: Nadaraya-Watson (NW) and locally-weighted affine (LA) used in [12]. The results of EpicFlow were generated using the published code from its authors with the parameters suggested for the Middlebury dataset [16]. We can observe that the modifications made to EpicFlow didn't deteriorate the performance and it added the Forward/Backward coherence as illustrated by Figure 4, where we can see that only pixels near edges are incoherent.

The precision provided by the edge preservation feature of the modified EpicFlow algorithm allowed us to remove the costly estimation of a quadratic parametric motion for each region used in [17] to ensure that motion edges are reliable. It allowed as well to remove a second pruning of the BPT that became unnecessary because of the highly improved precision in the computation of the occlusions.

3.2. Depth order estimation

Two datasets were used to evaluate the proposed method: the CMU dataset [11] and the BVSD [18]. The metric used to evaluate the performance of the depth order estimation is the **local consistence order** proposed in [17]. This metric is a generalization of the classic precision and recall metrics. This metric could be presented in a precision/recall coordinate system, but a result is represented by a segment instead of a point. The highest point of the segment represents the segmentation performance, and the lower one the combined effect of the segmentation and the ordering. Figure 5 shows the obtained results on the BVSD dataset. Two parameters were varied, the size S of the superpixels on the initial partition and N_{max}, the maximum number of regions allowed in the final partition. The black segment represents the result obtained using the ground truth of the segmentation as the input to the depth ordering estimation.

In [17], a global metric for the depth ordering estimation named Over Random Index (ORI) is also proposed. A positive ORI indicates that the system performs a better classification than a random classifier and an $ORI = 1$ indicates a perfect ordering. Our system obtained an $ORI = 0.27$ on the BVSD dataset. It is comparable to an $ORI = 0.25$ presented in [19] which includes the region optical flow estimation in it's calculation, resulting in a more complex process. Figure 6 presents the results for four sequences on the CMU dataset.

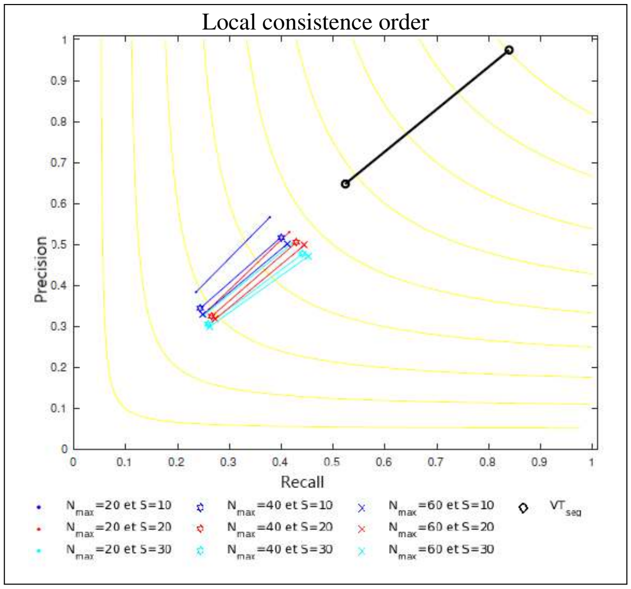

Fig. 5: Local consistence order results on the BVSD dataset

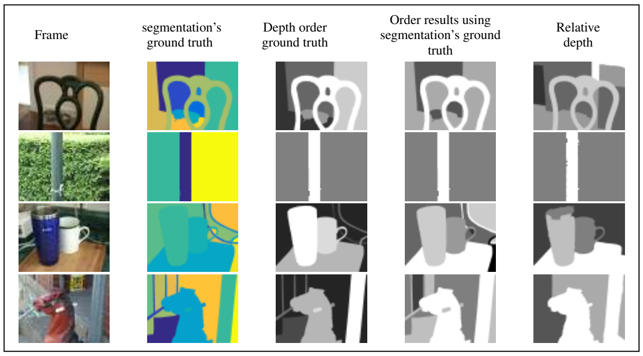
Frame
segmentation's ground truth
Depth order ground truth
Order results using segmentation's ground truth
Relative depth

Fig. 6: Results on the CMU dataset

4. CONCLUSION

In this work, we proposed a method to estimate depth ordering using motion occlusion cues. It shows that estimating a coherent forward/backward optical flow that preserves edges simplifies the depth ordering estimation while at the same time slightly improving the performance of the algorithm.

5. REFERENCES

[1] Xiaojun Huang, Lianghao Wang, Junjun Huang, Dongxiao Li, and Ming Zhang, "A depth extraction method based on motion and geometry for 2D to 3D conversion," *3rd International*

Symposium on Intelligent Information Technology Application, IITA 2009, vol. 3, pp. 294–298, 2009.

[2] Wei Liu, Yihong Wu, Fusheng Guo, and Zhanyi Hu, “An efficient approach for 2D to 3D video conversion based on structure from motion,” *Vis. Comput.*, vol. 31, no. 1, pp. 55–68, Jan. 2015.

[3] Anlong Ming, Tianfu Wu, Jianxiang Ma, Fang Sun, and Yu Zhou, “Monocular Depth-Ordering Reasoning with Occlusion Edge Detection and Couple Layers Inference,” *IEEE Intelligent Systems*, vol. 31, no. 2, pp. 54–65, 2016.

[4] James E. Cutting and Peter M. Vishton, “Chapter 3 - perceiving layout and knowing distances: The integration, relative potency, and contextual use of different information about depth*,” in *Perception of Space and Motion*, William Epstein and Sheena Rogers, Eds., Handbook of Perception and Cognition, pp. 69 – 117. Academic Press, San Diego, 1995.

[5] Amo Schodl and Irfan Essa, “Depth layers from occlusions,” *2001 IEEE Conference on Computer Vision and Pattern Recognition*, pp. 9–14, 2001.

[6] Adarsh Kowdle, Andrew Gallagher, and Tsuhan Chen, “Revisiting depth layers from occlusions,” *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pp. 2091–2098, 2013.

[7] Dorn Feldman and Daphna Weinshall, “Motion segmentation and depth ordering using an occlusion detector,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 30, no. 7, pp. 1171–1185, July 2008.

[8] Yue Feng, Jinchang Ren, and Jianmin Jiang, “Object-based 2D-to-3D video conversion for effective stereoscopic content generation in 3D-TV applications,” *IEEE Transactions on Broadcasting*, vol. 57, no. 2 PART 2, pp. 500–509, 2011.

[9] Philippe Salembier and Guillem Palou, “Depth order estimation for video frames using motion occlusions,” *IET Computer Vision*, vol. 8, no. 2, pp. 152–160, 2014.

[10] Philippe Salembier and Luis Garrido, “Binary partition tree as an efficient representation for image processing, segmentation, and information retrieval,” *IEEE Transactions on Image Processing*, vol. 9, no. 4, pp. 561–576, 2000.

[11] Andrew N. Stein and Martial Hebert, “Occlusion boundaries from motion: Low-level detection and mid-level reasoning,” *International Journal of Computer Vision*, vol. 82, no. 3, pp. 325–357, 2009.

[12] Jerome Revaud, Philippe Weinzaepfel, Zaid Harchaoui, and Cordelia Schmid, “EpicFlow : Edge-Preserving Interpolation of Correspondences for Optical Flow ,” *CVPR*, pp. 1164–1172, 2015.

[13] Philippe Weinzaepfel, Jerome Revaud, Zaid Harchaoui, and Cordelia Schmid, “DeepFlow: Large displacement optical flow with deep matching,” *Proceedings of the IEEE International Conference on Computer Vision*, pp. 1385–1392, 2013.

[14] Piotr Dollár and C. Lawrence Zitnick, “Fast edge detection using structured forests,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 37, no. 8, pp. 1558–1570, Aug 2015.

[15] Radhakrishna Achanta, Appu haji, Kevin Smith, Aurelien Lucchi, Pascal Fua, and Sabine Süsstrunk, “SLIC superpixels compared to state-of-the-art superpixel methods,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 34, no. 11, pp. 2274–2282, Nov 2012.

[16] Simon Baker, Daniel Scharstein, J. P. Lewis, Stefan Roth, Michael J. Black, and Richard Szeliski, “A database and evaluation methodology for optical flow,” *International Journal of Computer Vision*, vol. 92, no. 1, pp. 1–31, Mar 2011.

[17] Guillem Palou and Philippe Salembier, “Precision-Recall-Classification Evaluation Framework: Application to Depth Estimation on Single Images,” in *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, Zurich, 2014, vol. 8689, pp. 648–662.

[18] Patrik Sundberg, Thomas Brox, Michael Maire, Pablo Arbeláez, and Jitendra Malik, “Occlusion boundary detection and figure/ground assignment from optical flow,” pp. 2233–2240, June 2011.

[19] Guillem Palou Visa, *Monocular depth estimation in images and sequences using occlusion cues*, Ph.D. thesis, Universitat Politècnica de Catalunya, 2014.