IEEE Access
Multidisciplinary ⁝ Rapid Review ⁝ Open Access Journal

Received September 7, 2021, accepted September 16, 2021, date of publication September 20, 2021,
date of current version September 29, 2021.

Digital Object Identifier 10.1109/ACCESS.2021.3114324

ALFO: Adaptive Light Field Over-Segmentation

MARYAM HAMAD, (Graduate Student Member, IEEE), CAROLINE CONTI, (Member, IEEE), PAULO NUNES, (Member, IEEE), AND LUÍS DUCLA SOARES, (Senior Member, IEEE)
Instituto Universitário de Lisboa (ISCTE-IUL), 1649-026 Lisboa, Portugal
Instituto de Telecomunicações, 1049-001 Lisboa, Portugal

Corresponding author: Maryam Hamad (maryam.hamad@lx.it.pt)

This work was supported by the Fundação para a Ciência e Tecnologia (FCT)/Ministério da Ciência, Tecnologia e Ensino Superior (MCTES) through the National Funds under Project UIDB/50008/2020 and Project PTDC/EEI-COM/7096/2020.

ABSTRACT Automatic image over-segmentation into superpixels has attracted increasing attention from researchers to apply it as a pre-processing step for several computer vision applications. In 4D Light Field (LF) imaging, image over-segmentation aims at achieving not only superpixel compactness and accuracy but also cross-view consistency. Due to the high dimensionality of 4D LF images, depth information can be estimated and exploited during the over-segmentation along with spatial and visual appearance features. However, balancing between several hybrid features to generate robust superpixels for different 4D LF images is challenging and not adequately solved in existing solutions. In this paper, an automatic, adaptive, and view-consistent LF over-segmentation method based on normalized LF cues and K-means clustering is proposed. Initially, disparity maps for all LF views are estimated entirely to improve superpixel accuracy and consistency. Afterwards, by using K-means clustering, a 4D LF image is iteratively divided into regular superpixels that adhere to object boundaries and ensure cross-view consistency. Our proposed method can automatically adjust the clustering weights of the various features that characterize each superpixel based on the image content. Quantitative and qualitative results on several 4D LF datasets demonstrate outperforming performance of the proposed method in terms of superpixel accuracy, shape regularity and view consistency when using adaptive clustering weights, compared to the state-of-the-art 4D LF over-segmentation methods.

INDEX TERMS Automatic segmentation, adaptive light field over-segmentation, superpixels.

I. INTRODUCTION

Image segmentation is a process of dividing the scene into several coherent regions according to some criteria. Image segmentation aims at minimizing intra-variance and maximizing inter-variance among regions [1]. Several image processing and computer vision applications rely on image segmentation in different fields, such as medical imaging [2], autonomous vehicle navigation [3], and face or optical character recognition [4]. Available image segmentation algorithms in the literature require different levels of supervision to suit different types of applications. These algorithms can be classified into supervised [5], semi-supervised [6], and unsupervised (automatic) [7], [8], based on the need for pre-trained labels or human interactions.

Image over-segmentation divides the scene into uniform regions with similar visual characteristics, such as color or texture to obtain superpixels [9]. Most existing image over-segmentation methods belong to the unsupervised image segmentation class and can be categorized as clustering-based methods and graph-based methods [9]. Recently, researchers have also been attempting to exploit deep learning techniques to generate image over-segmentations for 2D images [10], [11]. These image over-segmentation methods, in [10], [11], belong to the supervised image segmentation class and have shown to achieve superior performance. However, preserving all image boundaries during the over-segmentation could be challenging, since the used ground truth labels for training are usually segmented in a more semantically meaningful level (e.g., object level). Additionally, although their performance is competitive compared to unsupervised methods, the generalization of the network to over-segment different datasets is still a challenge to be further studied.

By creating homogenous regions that involve local perceptually meaningful information (i.e., superpixels), subsequent

The associate editor coordinating the review of this manuscript and approving it for publication was Fahmi Khalifa.

image analysis and processing are facilitated [8]. A recent trend in computer vision and image processing applications is to process an image at the superpixel-level representation instead of the pixel-level representation. As an example, in image compression, superpixels can be used to reduce coding overhead by minimizing the number of regions that need to be coded [12], [13]. Additionally, superpixels can be used in object tracking [14], object segmentation [15], and saliency detection [8], [16].

As for 2D images, in 4D Light Field (LF) images, the superpixel concept can be also exploited to divide the various views into smaller regions. However, 4D LFs comprise spatial as well as angular scene information, since they capture the scene from different perspectives by using a camera array, a moving camera gantry or a single camera equipped with a microlens array in front of the sensor [17], [18]. Therefore, in 4D LF images the superpixel-level representation should correspond to regions that are coherent not only spatially but also angularly across views. In 4D LF processing, superpixel-level representation facilitates the propagation of subsequent processing tasks from a reference view into other views; hence, a significant reduction in computational complexity can be achieved. Furthermore, superpixel-level representation using appropriate LF superpixels, that consider angular and spatial geometry, helps ensure cross-view consistency, which is a critical property in 4D LF processing (e.g., in virtual reality applications, the 4D LF object must be accurately and consistently segmented in all views). Compared to 2D images, 4D LFs offer richer cues that can be used efficiently to significantly improve the robustness of image segmentation, such as depth information. In general, when traditional 2D segmentation is applied to 4D LFs, the cross-view information is not considered to resolve object occlusions, thus resulting in inconsistent or inaccurate image segmentations. Therefore, 4D LF over-segmentation solutions should aim at achieving superpixel cross-view consistency (e.g., without flickering borders or sudden shifts in border positions when the angular perspective is changed) in addition to other properties such as compactness (e.g., superpixel-shape regularity) and segmentation accuracy by adhering to object boundaries. Currently, there are only a few 4D LF over-segmentation solutions in the literature that tackle the above 4D LF superpixel challenges. Existing solutions for 4D LF over-segmentation can be classified as clustering-based methods [8], [19], [20] and graph-based methods [21], depending on the used approach. However, independently of the followed approach, they all suffer from two important limitations.

The first such limitation is the fact that the used parameters for clustering or graph optimization are empirically tuned to the specific set of tested images. Consequently, it may be very time-consuming, and it may not lead to an optimal set of parameters considering the actual content of each view. Moreover, when features of different nature (such as color, position, and depth) are used, the difference in range between them is not adequately considered. As a result, the superpixel accuracy and consistency may be negatively affected. A possible way to overcome this limitation is to use a content-adaptive algorithm that adjusts over-segmentation parameters. The adaptive algorithm can use the feedback values from previous iterations to dynamically adjust the parameters for better performance. This type of solutions has been proposed for adapting the used weights for segmentation or graph optimization in 2D superpixel segmentation algorithms with promising results, e.g., [22], [23]. Given the similarities between 2D and 4D LF image segmentation, a similar approach can be followed for LF images.

The second limitation is the fact that the angular information is currently not being fully exploited. In some cases [8], [19], only a sparse estimation of the disparity (i.e., the displacement of a point between different views, which is inversely proportional to the depth) is used for projecting superpixels from the central view to all other LF views. When a sparse or roughly estimated disparity is used for centroid projection, actual corresponding positions in other views may not be computed accurately, hence may generate inconsistent superpixels across views. Additionally, since in most existing solutions the disparity is used merely for projection and not for clustering, this seriously limits the ability to segment regions with the same visual appearance at different depths. In other cases [20], the central horizontal and vertical views are used to guide the segmentation and propagation, which may affect the accuracy or consistency in the off-central views.

To deal with the two limitations above, this paper proposes an adaptive view-consistent 4D LF over-segmentation method that belongs to the clustering-based over-segmentation class. The two main contributions of the proposed method are:

- **Automatic LF over-segmentation with adaptive clustering weights** – In the proposed method, the used features are first normalized using the min-max normalization method for proper feature weighting, preventing unbiased clustering and leading, this way, to a robust segmentation. Additionally, the clustering weights are adjusted adaptively based on the 4D LF content. For that, the discriminability measure proposed for 2D images [22] is adapted to compute the contribution of the used features and adjust the clustering weights accordingly. To the best of the authors' knowledge, this is the first 4D LF method that generates content-based adaptive 4D LF superpixels based on K-means clustering. Experiments and the dynamic results in the supplemental materials show outperforming results quantitively and qualitatively when adjusting the weights based on image content compared to the existing solutions that use fixed clustering weights.
- **Adaptive clustering based on a robust hybrid feature set** – The proposed method belongs to the clustering-based class using a bottom-up clustering approach with hybrid clustering features. Angular and spatial LF information is included to improve the

accuracy and cross-view consistency of the generated superpixels. The recent 4D view-consistent depth estimation method [24] that estimates per-pixel disparity is used during the superpixel segmentation as a discriminable feature, besides position and visual appearance. Exploiting per-pixel disparity for clustering and projecting improves the qualitative and quantitative results in terms of accuracy significantly and ensures view consistency.

The remainder of the paper is organized as follows. Section II briefly reviews the related work on 4D LF over-segmentation available in the literature. Section III describes the proposed Adaptive LF Over-segmentation (ALFO) method in detail, while Section IV evaluates its performance through a series of experimental results. Section V discusses some remaining limitations. Finally, Section VI concludes the paper with some final remarks and proposes directions for future work.

II. RELATED WORK

Superpixels have attracted increasing attention since their naming in 2003 [25]. Several over-segmentation solutions for obtaining superpixels in 2D images have already been proposed; a comprehensive review can be found in [26]. For 4D LF images, unsupervised over-segmentation solutions have been proposed and can be classified as either clustering-based or graph-based 4D LF over-segmentation methods.

A. CLUSTERING-BASED 4D LF OVER-SEGMENTATION

In this class, the image is segmented by defining centroids (a.k.a. seeds) to guide the segmentation, with each pixel being grouped into the nearest centroid based on some criteria. The existing solutions use K-means clustering to generate the 4D LF superpixels, where K is the number of superpixels.

Initially, Hog *et al.* [8] introduced the concept of superrays to achieve superpixel segmentation for LFs. Using K-means clustering, the 2D square grid of the central view is projected to the other LF views, based on a roughly estimated disparity for the central view centroids only. Afterwards, the pixels are assigned to the nearest superray based merely on color and position features. During the clustering, the color and position of each centroid are updated. However, the centroid disparity is never updated even when the centroid position is changed. The clustering is iteratively applied until convergence is reached. Finally, a cleaning step is needed to smooth the labeling. In [27], the authors extended the work to handle LF video by including the temporal dimension. Although their proposed solution has a fast execution time, the resulting superrays are not always consistent across views [20], [21].

Zhu *et al.* [28], [19] proposed a robust superpixel Light Field SuperPixel (LFSP) segmentation method. Given the depth map of the central view, they first perform a 2D K-means superpixel segmentation for the central view using a 2D superpixel algorithm. Then, the result is projected into the entire LF based on the central view depth map. Lastly, after clustering, the segmentation boundaries are optimized using the Block Coordinate Descent algorithm (i.e., an optimization algorithm that sequentially minimizes a multivariable function along one direction at a time to find the minimum of that function) [19] to preserve boundaries for occluded objects. Since the depth map is used to segment the central view only, the objects in the off-central views that are occluded in the central view may not be segmented properly across views.

Khan *et al.* [20] proposed a View-Consistent Light Field Superpixel (VCLFS) segmentation with implicit disparity estimation based on Epipolar Plane Images (EPIs) (i.e., the unique 2D spatio-angular slice of the LF. Each EPI contains several oriented lines, and the slope of these lines is associated with the disparity) [29]. They use two stacks of the central horizontal and central vertical views independently to generate the EPIs. Each pair of lines in an EPI represents a segment; hence, cross-view consistency can be enforced by propagating the labels through these lines. After applying the segmentation in the EPI space, they use K-means clustering by combining the angular segmentations in horizontal and vertical EPIs into the central view. Labels are then propagated to all off-central views in the 4D LF using per-pixel disparity. Finally, unlabeled pixels are assigned to the label of the nearest neighbor in each view independently. Although the disparity is exploited during clustering in this solution, in some cases, such as for non-Lambertian or occluded objects in the central EPIs, not all the superpixels are view consistent. In all the mentioned solutions, fixed values are used for the clustering weights and most of them also fixed the number of iterations, independently of the content.

B. GRAPH-BASED 4D LF OVER-SEGMENTATION

In this class, the image is represented as a weighted undirected graph. Each pixel is considered as a graph node. Afterwards, graph optimization techniques are used to separate the graph into sub-graphs to generate superpixels based on the edge weights between the nodes. Due to the huge number of pixels in 4D LF images, graph-based solutions are generally complex in terms of the used resources and execution time.

Li and Heidrich [21] proposed a Hierarchical and View-invariant LF Segmentation (HVLFS) method. Given the estimated depth for all LF views, they use 4D graph segmentation by applying greedy heuristic optimization to maximize the entropy rate in a 4D weighted undirected graph. The proposed method generates hierarchical superpixels with different sizes based on the user input. This solution exploits several features, such as depth and texture, and no centroids projection is used. Due to the huge graph structure, the authors proposed several optimization techniques and data structures to reduce the complexity, such as disjoint trees and max heap structure. However, they also mentioned some limitations regarding the need for normalizing the weight values of the used optimization function. Moreover, a massive amount of computing resources is needed for dense LF segmentation.

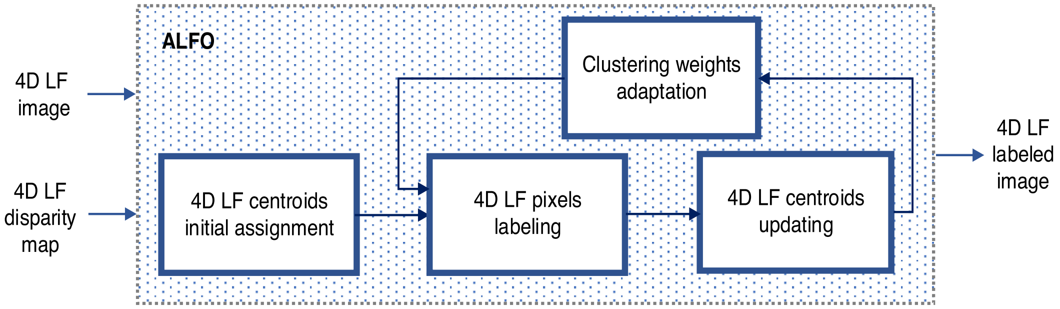

FIGURE 1. Overview of the proposed ALFO method. Given a 4D LF image and the corresponding disparity maps for all views, initial centroids, characterized by distinct features, are assigned in a reference view. Next, the 4D LF superpixel segmentation is achieved by iteratively applying K-means clustering, including pixel labeling, centroids updating and clustering weights adaptation, until convergence is reached.

To the best of our knowledge, to date, these are the existing solutions that address the 4D LF superpixel segmentation problem. All these solutions rely on several fixed parameters for different input images during the K-means clustering or the graph optimization, without considering the relative importance of various features for each image. Additionally, the used features are not normalized before the clustering; hence, they cannot be weighted properly, and the superpixels may not be generated optimally.

III. PROPOSED METHOD

The proposed ALFO method aims at generating 4D LF superpixels that respect visual appearance, compactness, occlusions, and cross-view consistency. The proposed method consists of four major stages as shown in Fig. 1. To generate the 4D LF superpixels, firstly, the disparity of all 4D LF views are estimated entirely (i.e., for each pixel) using the View-consistent 4D Light Field Depth Estimation algorithm proposed in [24]. Given the input LF image, the estimated disparity for all views, and the grid step size, the central view is selected to initialize the centroids and assign them the initial feature values (i.e., position, color and disparity) extracted from the central view of the 4D LF image and the central disparity map in the grid spatial coordinates. Next, the centroids are projected to each view using the disparity (i.e., the disparity from the central view to other views) to ensure consistency across views. After that, the K-means clustering is applied for each view in the 4D LF to assign a label for each pixel according to its "nearest" centroid, considering all the features. The features of all centroids are updated iteratively by back-projecting the pixels that belong to each superpixel from all LF views into the central view. Finally, to optimize the segmentation, the used clustering weights are adapted according to the content of each image and the generated superpixels in the current iteration. Each stage in Fig. 1 will be detailed in the following sub-sections and the main notations used in this paper are summarized in Table 1.

TABLE 1. Main notations used in this paper.

Symbol	Definition
$I(x,y,u,v)$	A 4D light field image with x,y spatial coordinates and u,v angular coordinates
ref	Short form for the angular coordinates of the reference view; in this paper, refers to the central view of I, I^{ref}
K	Number of superpixels
S_{size}	Grid step size (a.k.a., superpixel size)
c	Centroid index of a superpixel, where $c \in \{1,\ldots,K\}$
Ω_c	Searching window centered at centroid c, with size equal to $(4 \times S_{size})^2$
$\lvert A \rvert$	Cardinality of set A
$\mathbf{p}^{u,v}$	A pixel in (u,v) view with spatial position (x,y)
$\mathbf{c}^{u,v}$	A projected centroid from ref view into (u,v) view
$\mathbf{c}^{ref}$	An original centroid in ref view
$d_{hor,\mathbf{p}}^{(u,v)\to(u\prime,v\prime)}$, $d_{ver,\mathbf{p}}^{(u,v)\to(u\prime,v\prime)}$	Horizontal and vertical disparities, respectively, of pixel $\mathbf{p}^{u,v}$ from view (u,v) to view (u',v')
F	Clustering feature set, where $F = \{p,l,a,b,d\}$, consists of relative position p, three color channels l,a,b in CIELAB color space and disparity d
S_c	A superpixel in 4D space
$S_c^{u,v}$	A 2D superpixel slice of S_c in (u,v) view
B_c^{ref}	Set of all back-projected pixels that belong to superpixel S_c from all views into the ref view
$D_f(\mathbf{p}^{u,v},\mathbf{c}^{u,v})$	Distance between pixel $\mathbf{p}$ and centroid $\mathbf{c}$ in (u,v) view according to feature f, where $f \in F$
WSV_f	Within superpixel variance of feature f, where $f \in F$
w_f	Clustering feature weight of feature f, where $f \in F$

A. 4D LF CENTROIDS INITIAL ASSIGNMENT

Initially, the 4D LF image (represented as a 2D array of 2D views) is converted to CIELAB color space. This color space was designed to approximate the human visual perception; thus, it is typically used in image segmentation. After that, a reference view (e.g., central view) is selected to initialize the clustering centroids in a grid. A uniformly distributed grid is used where the center of each grid square represents a centroid, and the initial distance between two centroids is defined as the grid step size, S_{size}, as illustrated in Fig. 2. The value of S_{size} is defined by the user, or a default value (e.g., 20 pixels) can be used to generate superpixels that adhere well to the boundaries. S_{size} is commonly referred as the superpixel size in the literature [7], [20].

After generating the centroids grid, each centroid will be characterized by several features, namely relative position,

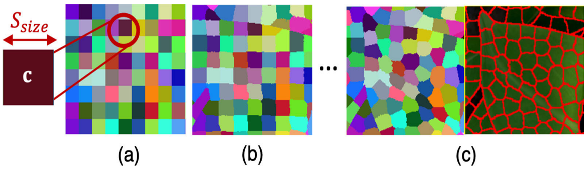

FIGURE 2. Visual representation of the clustering iterations: a) initial square grid in the central view only. Each square represents a superpixel and the center point of each square represents its centroid. In (a), for illustration, all pixels are labeled, however, initially, only centroids have labels; b) labeling result after the first iteration; c) final labeling output.

color and disparity. However, due to the differences in the hybrid features ranges, the used features are normalized to properly weight them in the next stages. The min-max normalization [30] is used as in (1):

$$\omega_{norm} = \frac{\omega - \omega_{min}}{\omega_{max} - \omega_{min}}, \tag{1}$$

where ω_{norm} is the normalized value, ω is the current value and ω_{min}, ω_{max} are the minimum and maximum values in the dataset, respectively. For LF images, the MATLAB conversion from RGB color space to CIELAB color space is used, and the CIELAB LF image is normalized to the range of [0, 1] using the color space ranges, namely [0, 100] for l channel, and $[-100, 100]$ for a and b channels. These ranges are obtained from MATLAB documentation [31]. To normalize the disparity feature, the maximum and minimum values from the dense 4D LF dataset are used. Although the used test images in our experiments are within the disparity range of $[-2.25, 2.25]$ pixels for horizontally adjacent views, we considered a larger range than the used test images to ensure robust over-segmentation for other dense LF datasets available with disparity values up to $[-4, 4]$ [32]. The position feature normalization will be detailed later in Sub-section C. To exploit the 4D LF cues in segmentation, each pixel is characterized by its color and disparity values, according to its location (x, y, u, v), where (x, y) are the spatial coordinates and (u, v) are the angular coordinates.

B. 4D LF PIXELS LABELING

Like state-of-the-art 4D LF superpixel methods, we assume the centroids in the central view also exist in all other 4D LF views. Given the disparity maps for all 4D LF views and the initial centroids in the central view, the K-means clustering is applied to each view by first projecting the centroids from the central view into each view, as in (2):

$$\begin{aligned} c_x^{u,v} &= c_x^{ref} + d_{hor,\mathbf{c}}^{ref\rightarrow(u,v)}, \\ c_y^{u,v} &= c_y^{ref} + d_{ver,\mathbf{c}}^{ref\rightarrow(u,v)}, \end{aligned} \tag{2}$$

where $(c_x^{u,v}, c_y^{u,v})$ are the spatial coordinates of the projected centroid using the disparity of the reference centroid located at (c_x^{ref}, c_y^{ref}), and $(d_{hor,\mathbf{c}}^{ref\rightarrow(u,v)}, d_{ver,\mathbf{c}}^{ref\rightarrow(u,v)})$ are the horizontal and vertical disparities from the reference view $ref = \left(u_{ref}, v_{ref}\right)$ to view (u, v), respectively. Since the used disparity estimation method generates per-pixel disparities from each view to its right horizontal adjacent view, and considering uniformly sampled LF, the disparity value is computed as in (3) [19]:

$$\begin{aligned} d_{hor,\mathbf{c}}^{ref\rightarrow(u,v)} &= d_{\mathbf{c}} \times \left(u - u_{ref}\right), \\ d_{ver,\mathbf{c}}^{ref\rightarrow(u,v)} &= d_{\mathbf{c}} \times \left(v - v_{ref}\right), \end{aligned} \tag{3}$$

where $d_{\mathbf{c}}$ is the disparity of the centroid from each view to its right horizontal adjacent view and (u_{ref}, v_{ref}) are the angular coordinates of the *ref* view. However, if the camera baselines are different for horizontal and vertical directions (e.g., the LF is captured by a camera array), in this case, camera parameters (extrinsic and intrinsic matrices) should be considered [19]. The projected centroid $(c_x^{u,v}, c_y^{u,v})$ may belong to R^2, however, in the used datasets we only have color and disparity values for integer positions. To access these features from the projected centroid, the color and disparity values are obtained by rounding the coordinates to ensure integer indexing belonging to z^2. Notice that the normalized unrounded values of the position and disparity are used for clustering and clustering weights adaptation. Unnormalized values are only used for projection.

To improve the clustering performance, searching is performed in a small window, Ω_c, with size $(4 \times S_{size})^2$ around each centroid in each view. The searching window enforces spatial connectivity and improves the performance since most 4D LF superpixels have a local slice in each view [7] (i.e., are non-occluded). As shown in Fig. 3, for narrow baselines (e.g., when $d_{\mathbf{c}} < S_{size}$), each centroid in the reference view is assumed to exist in all views with a slight disparity. The solid arrows describe the projection of the centroids from the reference view into other views based on the disparity of the centroid. After projecting from the reference view into all other LF views, for each pixel, let F represents the set of clustering features $\{p, l, a, b, d\}$, where p stands for relative position, l, a, b for the three color channels in the CIELAB color space and d for the average of the horizontal and vertical disparities, respectively. Each pixel in all LF views is then assigned to the "nearest" superpixel according to the weighted distance, D_w, as in (4)-(9):

$$D_p(\mathbf{p}, \mathbf{c}) = \sqrt{\frac{(p_x - c_x)^2 + \left(p_y - c_y\right)^2}{8 \times S_{size}^2}}, \tag{4}$$

$$D_l(\mathbf{p}, \mathbf{c}) = \sqrt{\left(l_{\mathbf{p}} - l_{\mathbf{c}}\right)^2}, \tag{5}$$

$$D_a(\mathbf{p}, \mathbf{c}) = \sqrt{\left(a_{\mathbf{p}} - a_{\mathbf{c}}\right)^2}, \tag{6}$$

$$D_b(\mathbf{p}, \mathbf{c}) = \sqrt{\left(b_{\mathbf{p}} - b_{\mathbf{c}}\right)^2}, \tag{7}$$

$$D_d(\mathbf{p}, \mathbf{c}) = \sqrt{\left(d_{\mathbf{p}} - d_{\mathbf{c}}\right)^2}, \tag{8}$$

$$\begin{aligned} D_w\left(\mathbf{p}, \mathbf{c}\right) = {} & w_p \times D_p^2 + w_l \times D_l^2 + w_a \times D_a^2 \\ & + w_b \times D_b^2 + w_d \times D_d, \end{aligned} \tag{9}$$

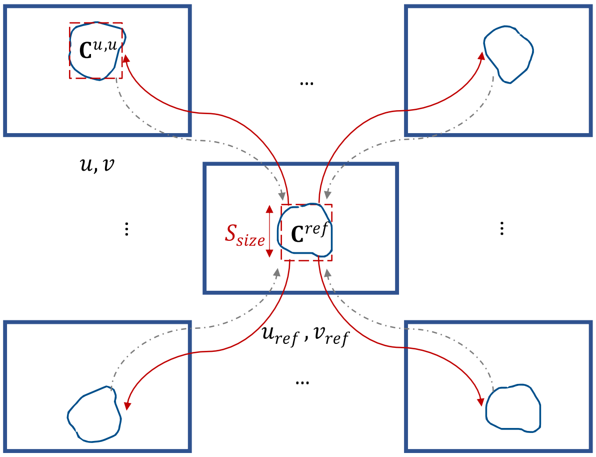

FIGURE 3. **Assuming all centroids in the reference view exist in all other views, the projection of a centroid from the reference view into other views is illustrated by the solid red arrows. Similarly, back-projection of all pixels that belong to a superpixel from all other views into the reference view is illustrated by the dashed arrows.**

where w_p is the relative position clustering weight, w_l, w_a, w_b are the color clustering weights, w_d is the disparity clustering weight, $\mathbf{p}$ represents each pixel that belongs to the searching window centered on centroid $\mathbf{c}$ and D_p, D_l, D_a, D_b, D_d are the relative position, color and disparity distances between each pixel $\mathbf{p}$ and a centroid $\mathbf{c}$, respectively. Note that D_d is not squared in (9) as will be detailed in Section IV. To normalize the relative position feature, D_p is divided by $8 \times S_{size}^2$, by considering the minimum distance to be zero and the maximum distance to be $2 \times S_{size}$, for both x and y coordinates.

In the first iteration, all the weights are initialized with same value, equal to $1/|F|$, where $|F|$ is the number of the used clustering features. After extensive testing, we noticed that the values of the initial weights do not significantly impact the final clustering weights. Notice that the used weights must be in the (0, 1) range and the summation of all weights is equal to one. Let $S = \{S_1, \ldots, S_k\}$ represents the set of all superpixels, the over-segmentation can be considered as an energy minimization problem as in (10):

$$E = \arg\min_{S} \sum_{c=1}^{K} \sum_{u=1}^{N_u} \sum_{v=1}^{N_v} \sum_{\mathbf{p} \in S_c^{u,v}} D_w(\mathbf{p}^{u,v}, \mathbf{c}^{u,v}), \quad (10)$$

where K is the number of superpixels, N_u, N_v are the horizontal and vertical dimensions of the LF array of views, respectively.

C. 4D LF CENTROIDS UPDATING

After assigning each pixel in all 4D LF views to the "nearest" superpixel (in terms of D_w), the clustering feature set, F, for each centroid in the central view is updated iteratively as described in this section.

The average value of the color channels from all pixels that belong to that superpixel, considering the entire 4D space, are assigned to each centroid. However, since in each iteration, all centroids in the central view are projected to all 4D LF views, only the relative position of each centroid is updated. To update the relative position of the centroids in the central view, all the pixels that belong to a given superpixel in each view are back-projected into the central view using the disparity of each pixel (see Fig. 3 dashed arrows), as in (11):

$$\begin{aligned} p_x^{ref} &= p_x^{u,v} + d_{hor,\mathbf{p}}^{(u,v)\to ref}, \\ p_y^{ref} &= p_y^{u,v} + d_{ver,\mathbf{p}}^{(u,v)\to ref}, \\ (c_x^{ref}, c_y^{ref}) &= \frac{1}{\left|B_c^{ref}\right|} \times \left(\sum_{\mathbf{p} \in B_c^{ref}} p_x^{ref}, \sum_{\mathbf{p} \in B_c^{ref}} p_y^{ref} \right), \end{aligned} \quad (11)$$

where $\left(p_x^{ref}, p_y^{ref}\right)$ are the back-projected spatial coordinates of the pixel using its horizontal and vertical disparities $d_{hor,\mathbf{p}}^{(u,v)\to ref}$, $d_{ver,\mathbf{p}}^{(u,v)\to ref}$ from view (u, v) into the *ref* view, B_c^{ref} is the set of all back-projected pixels that belong to superpixel S_c from all views into the *ref* view, with $c \in \{1, \ldots, K\}$, and (c_x^{ref}, c_y^{ref}) are the updated spatial coordinates of the centroid in the *ref* view. In contrast to the solution described in [8], where the pixels of all views are back-projected into the central view using the same coarse estimated disparity of the central view centroids only, we use the estimated disparity values of each pixel that belong to the corresponding superpixel in the 4D space to properly back-project into the central view. Similarly, as in (3), according to the used disparity estimation method, the disparity from any view (u, v) to the *ref* view is computed as in (12):

$$\begin{aligned} d_{hor,\mathbf{c}}^{(u,v)\to ref} &= d_{\mathbf{c}} \times \left(u_{ref} - u\right), \\ d_{ver,\mathbf{c}}^{(u,v)\to ref} &= d_{\mathbf{c}} \times \left(v_{ref} - v\right). \end{aligned} \quad (12)$$

After that, the spatial position of each centroid is determined as the average pixel coordinates of all pixels that belong to the given superpixel. The back-projection step is used to update the centroids positions in the *ref* view without being affected by the slight disparity across views (e.g., if actual positions of all pixels are considered).

Finally, after updating the positions of the centroids, the disparity value of each centroid needs to be updated as well. Given the estimated disparity maps, each centroid disparity is updated using the disparity value of the updated position (rounded to integer positions) from the disparity map. The actual disparity in the updated centroid position is used in our method instead of computing the average disparity of all pixels in a superpixel. This approach ensures a robust projection of a given centroid from the reference view into other views in the next iteration. Different from the proposed solution in [20], where the average disparity of all pixels that belong to each superpixel is considered to update the disparity of each centroid. Additionally, the centroid disparity is never updated in the proposed solution in [8], even when a centroid changed its position, which may affect the projection accuracy, hence degrading the superpixels consistency.

D. CLUSTERING WEIGHTS ADAPTATION

Due to the different nature of the used features, fixing clustering weights for all image types without considering their content is a non-trivial, time-consuming task and may generate non-optimal over-segmentations. To improve over-segmentation flexibility and robustness, and to overcome this drawback, which prevails in the existing 4D LF superpixel solutions, adaptive clustering weights are used in our proposed method. The technique considered here was inspired by the adaptation technique proposed in [22] for 2D clustering to adapt the K-means clustering weights iteratively based on their within-cluster variance. As proposed in [22], the principle of feature discriminability states that the features with the smaller sum in within-superpixel variance (i.e., the total sum of the feature distances from each pixel to its centroid in all superpixels) are more distinguishable. Therefore, they can be assigned larger weights to guide the segmentation. To compute the discriminability of each clustering feature, after each K-means iteration and after all the 4D LF centroids are updated, the normalized within-superpixel variance for each feature f is computed by using (13):

$$WSV_f = \sum\nolimits_{c=1}^{K} \sum_{u=1}^{N_u} \sum_{v=1}^{N_v} \sum_{\mathbf{p}\in S_c^{u,v}} D_f\left(\mathbf{p}^{u,v}, \mathbf{c}^{u,v}\right)^2, \tag{13}$$

where K is the number of superpixels, N_u, N_v are the horizontal and vertical dimensions of LF array of views, respectively, $S_c^{u,v}$ is a 2D slice of superpixel S_c in view (u, v), $\mathbf{p}$ represents each pixel that belongs to the superpixel S_c in all 4D LF views, D_f is the feature distance from each pixel $\mathbf{p}^{u,v}$ and the projected centroid $\mathbf{c}^{u,v}$ in view (u, v), and $f \in F$. In [22], WSV_f is then divided by the range of feature f in a given image to normalize it. However, during clustering, in [22], the used features are not normalized, and range differences are not considered. Different from [22], in this paper, the clustering features are normalized initially, hence, WSV_f, is computed based on normalized features, and for proper weighting, the normalized features are also used during clustering.

Initially, all feature clustering weights, are assigned to $1/|F|$. After that, we iteratively update the clustering weights according to the generated superpixels of the current iteration. Based on [22], features with smaller values of WSV_f are coherent among the superpixel, and can generate a compact grouping for similar pixel values. Hence, to optimize the clustering weights, a higher weight value is assigned to the feature with small WSV_f value, as in (14):

$$w_f = \frac{1}{\sum_{t\in F}\left(WSV_f / WSV_t\right)^{\frac{1}{|F|-1}}}, \tag{14}$$

where t is a feature that belongs to the features array F. The summation of all the clustering weights should be equal to 1 in all iterations.

Since the proposed method is adaptive, the number of K-means iterations is content-dependent as well. After each iteration, the average displacement of all centroids is computed by finding the Euclidian distance between the centroid previous position and the updated one in the *ref* view as in (15):

$$D_{avg} = \frac{1}{K}\sum_{c=1}^{K}\sqrt{\left(c_{x'}^{ref} - c_{x}^{ref}\right)^2 + \left(c_{y'}^{ref} - c_{y}^{ref}\right)^2}, \tag{15}$$

where $(c_{x'}^{ref}, c_{y'}^{ref})$ and $(c_{x}^{ref}, c_{y}^{ref})$ are, respectively, the previous and updated spatial coordinates of each centroid in the *ref* view, and K is the number of superpixels. The 4D LF superpixel segmentation will iterate until D_{avg} reaches 0.5% of S_{size} (i.e., the grid step size), or until it reaches the maximum number of iterations (e.g., 20 iterations).

According to the image dimensions and grid shape or step size, the approximate number of generated 4D LF superpixels, K, can be computed and rounded from (16), where S_{size} is the grid step size, and $\left|I^{ref}\right|$ is the number of pixels in the *ref* view:

$$K \approx \frac{\left|I^{ref}\right|}{S_{size}^2}. \tag{16}$$

The entire proposed algorithm is summarized in Algorithm 1.

Algorithm 1: ALFO: Adaptive Light Field Over-Segmentation

```
Input: 4D light field image, $I$, step size, $S_{size}$, and 4D light field
       disparity map, $Z$
Result: 4D light field labeled image, $L$
Initialize a 4D regular grid with step size in the reference view;
Initialize the $K$ centroids using reference view values and normalized
  features;
Initialize clustering weights to $1/|F|$;
Initialize pixel label $L(p) = 0$ for each pixel;
Initialize pixel distance $D(p) = \infty$ for each pixel;
while not converged or reached max iterations do
    $D(\mathbf{p})=\infty$;
    for each centroid $\mathbf{c} \in \{1, \ldots, K\}$ do
        for each view $(u, v) \in I$ do
            project $\mathbf{c}$ into $(u, v)$ view using (2);
            Create searching window, $\Omega_c$, around the
              projected $\mathbf{c}$;
            for each pixel $p \in \Omega_c$ do
                Compute features distance, $D_w(\mathbf{p}, \mathbf{c})$,
                  using (9);
                if $D_w(\mathbf{p}, \mathbf{c}) < \mathbf{D}(\mathbf{p})$ then
                    $L(\mathbf{p}) \leftarrow L(\mathbf{c})$;
                    $D(\mathbf{p}) \leftarrow D_w(\mathbf{p}, \mathbf{c})$;
                end
            end
        end
    end
    Update color, position and disparity for
      each $\mathbf{c}$;
    Compute within-superpixel variance, $WSV_f$,
      for each feature using (13);
    Update clustering weights, $W_f$, using (14);
end
```

IV. EXPERIMENTAL RESULTS

In this section, the proposed method is analyzed and evaluated. For this purpose, quantitative and qualitative comparisons with the state-of-the-art methods are performed. Initially, the used datasets, benchmark methods and evaluation metrics are introduced. Afterwards, the generated results and comparisons are discussed. In this analysis, visual results are presented only from top-left, central, and bottom-right LF views to show the over-segmentation consistency across the 4D LF views. Nevertheless, to visualize the entire 4D LF views and the smooth transition across views, we highly encourage the reader to see our results in the supplemental material for dynamic visualizations available online.[1]

A. DATASETS AND PARAMETER SETTINGS

To evaluate the proposed method, both synthetic and real (i.e., not synthetic) 4D LF datasets are used to obtain the experimental results. For synthetic 4D LF images, the HCI 4D LF dataset [33] is used. The HCI dataset includes both Ground Truth (GT) disparity maps and segmentation labels. Additionally, for real 4D LF images, the EPFL MMSPG dataset captured with a Lytro Illum camera [34] is used, as shown in Table 2. Due to the vignetting effects in this dataset (i.e., darkening of the edges of the captured micro-images), only the central 13 × 13 views are used, thus discarding the entirely dark views in the 4D LF corners.

TABLE 2. Image datasets used in the experimental results.

4D LF dataset	View resolution $(N_x \times N_y)$ pixels	Number of views $(N_u \times N_v)$	Thumbnails
HCI benchmark dataset [33]: Buddha, Papillon, Horses and StillLife	768×768 except for horses: 1024×576	9×9	
MMSPG LF images dataset [34]: Friends1, Sphynx, Bikes, and Sophie and Vincent 3	625×434	15×15	

It is worth highlighting that our method does not use any empirically set clustering weights or any post-processing optimization (e.g., to regularize the superpixel borders across views) or cleaning (e.g., to remove sparse pixels that are labeled wrongly). Solely the maximum number of iterations is set empirically. The maximum number of iterations is set to 20 to ensure robust segmentation even for complex texture images. As illustrated in Fig. 4, the average displacement of the centroids, D_{avg}, converges after 10-15 iterations and goes, usually, below 0.5% of the superpixel size before 20 iterations (see the threshold line in Fig. 4). Moreover, we noticed that the results were not significantly improved when the clustering is terminated based on this threshold value compared to the maximum number of iterations. The superpixel size is assigned by the user to control the generated superpixel size according to the desired application. In our experiments, several superpixel sizes are tested and the central view is used as a clustering reference view.

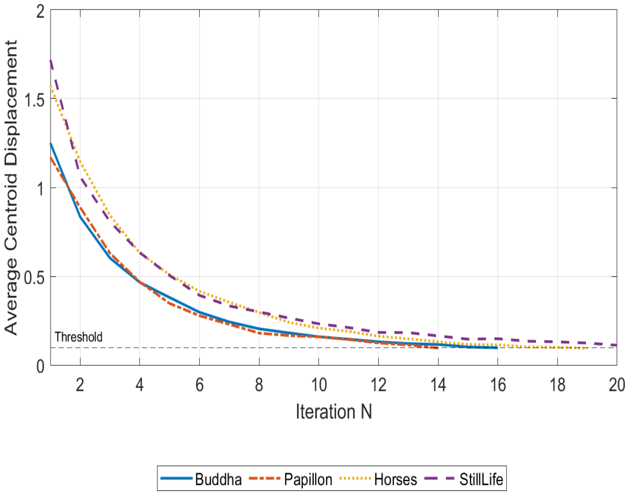

FIGURE 4. Average displacement of centroid spatial coordinates, D_{avg}, in pixels, along the number of iterations. $S_{size} = 20$.

B. BENCHMARK METHODS

To compare our method with the state-of-the-art methods presented in Section II, we used the open-source software provided by the authors of the LFSP [19] and the VCLFS [20] methods. For the LFSP method, we used the depth estimation algorithm proposed in [35] applied for central view only, as defined in the LFSP proposal. To compare with the Superray method [8], we used the superray software that was implemented and used in [36], since the original software of the Superray method [8] is not publicly available. To generate the superrays, several parameters are needed to be assigned, such as disparity range between two adjacent LF views, and compactness weight (e.g., a weight that controls superpixel compactness and balances between color and position features during the clustering). The disparity range is obtained from the used estimated disparity in [24] for each test image independently, and the compactness weight is set to 10 for better results for different superpixel sizes. For the HCI dataset, several superpixel sizes were tested for all the mentioned solutions (i.e., {15, 20, 25, 30, 35, 40}). For the MMSPG LF dataset, since there is no labeling GT available, only $S_{size} = 20$ was tested, as detailed below. Finally, we compared our proposed method with the HVLFS method [21] using the 4D LF labeled images from the HCI dataset provided by the author, with average superpixel sizes belong to [10, 45].

[1] Higher quality versions at https://github.com/MaryamHamad/ALFO

C. EVALUATION METRICS

In 2D superpixel methods, there is, usually, a requirements trade-off between compactness (e.g., shape regularity) and accuracy including boundary adherence [7]. In addition to these requirements, 4D LF superpixels should also be consistent across views (e.g., to have coherent shape and no flickering borders or sudden shifts in border position when the angular perspective is changed). To evaluate these characteristics quantitatively, the following metrics are considered [20]:

1) ACCURACY AND COMPACTNESS METRICS

- **Achievable Accuracy (AA)** – Since the GT labels, L_{GT}, are segmented at the object-level with n segments, each superpixel in the labeled image, L, is assigned to the label of the L_{GT} segment that has the largest overlap with the current superpixel. Afterwards, the accuracy is measured as follows [22]:

$$AA = \frac{1}{N_{u,v}} \sum_{u,v} \left\{ \frac{1}{|I^{u,v}|} \sum_{c=1}^{K} \max_{j} \left| S_c \cap G_j \right| \right\}, \quad (17)$$

where $N_{u,v}$ is the number of all 4D LF views, $\left|I^{(u,v)}\right|$ is the number of pixels in a single LF view, (u, v) are the angular coordinates for all LF views, K is the number of superpixels, S_c is a superpixel in L and G_j is the jth segment in L_{GT}, with $j = \{1, \ldots, n\}$. A higher value indicates better accuracy.

- **Boundary Recall (BR)**– Given the GT boundary image, B_{GT}, let True Positive, TP, and False Negative, FN, represent the number of boundary pixels (i.e., pixels that represents image edges) in the superpixel labeled image, L, with respect to B_{GT}. Then, the boundary recall is computed as follows [37]:

$$BR = \frac{TP}{TP + FN}, \quad (18)$$

where TP is the number of boundary pixels in B_{GT} that share boundary pixels with L within chessboard distance, β, in pixels, FN is the number of boundary pixels in B_{GT} that do not share any boundary pixels with L within distance β, where β is set to 2 as in [20]. A higher value of BR indicates better adherence to the object boundaries.

- **Under-segmentation Error (UE)**– This metric computes the percentage of superpixels that overlap GT segment borders as follows [37]:

$$UE_{u,v} = \sum_{j=1}^{n} \frac{\sum_{S_c : S_c \cap G_j = \emptyset} \min\left(\left|S_c^{IN}\right|, \left|S_c^{OUT}\right|\right)}{\left|G_j\right|},$$
$$UE = \frac{1}{N_{u,v}} \sum_{u,v} \frac{UE_{u,v}}{|I^{u,v}|}, \quad (19)$$

where n is the number of segments in GT labels, and S_c^{IN}, S_c^{OUT} represent the inside and outside parts of a superpixel that are divided by a GT label segment G_j, $\left|S_c^{IN}\right|$, $\left|S_c^{OUT}\right|$, $\left|G_j\right|$, represent the number of pixels in each segment, $N_{u,v}$ is the number of 4D LF views and $|I^{u,v}|$ is the number of pixels in a single LF view. This metric evaluates the quality of segmentation based on the requirement that a superpixel should overlap with only one object. A lower value of UE indicates that the superpixels are less likely to flood over the GT segment borders, hence indicates improved accuracy.

- **Compactness (CP)** – This metric measures superpixel boundary curvature as follows [20]:

$$CP = \frac{1}{N_{u,v}} \sum_{u,v} \sum_{S_c \in S} \frac{4\pi A_{S_c} \left|S_c\right|}{|I^{u,v}| \, P_{S_c}^2}, \quad (20)$$

where $N_{u,v}$ is the number of 4D LF views, S is the set of superpixels in labeled image, L, $|I^{u,v}|$ is the number of pixels in a single LF view, A_{S_c} and P_{S_c} are the area and perimeter of superpixel S_c, respectively, and $|S_c|$ is the number of pixels in S_c. Larger CP values indicate smoother borders of superpixels and better regulation in superpixel size across views.

2) ANGULAR SIMILARITY AND CONSISTENCY METRICS

- **Self-Similarity (SS)**– As defined in [19], centroids are back-projected from each view into the *ref* view using the GT disparity. The self-similarity error computes the average distance between the back-projected centroids from all views and the centroids in the central view, the approach in [20] is used as follows:

$$SS = \frac{1}{K} \sum_{c=1}^{K} \left\{ \frac{1}{N_{u,v}} \sum_{u,v} \sqrt{\left(\mathbf{c}_{c,(u,v)}^{ref} - \mathbf{c}_{c}^{ref}\right)^2} \right\}, \quad (21)$$

where K is the number of superpixels, $N_{u,v}$ is the number of 4D LF views, $\mathbf{c}_{c,(u,v)}^{ref}$ is the back-projected centroid from view located in angular coordinate (u, v) into *ref* view, and $\mathbf{c}_{c}^{ref}$ is the original centroid in the *ref* view. A smaller SS error indicates better consistency.

- **Number of Labels per Pixel (LP)**– This metric computes the average number of labels per pixel in the *ref* view by projecting the labels from the *ref* view to other views via GT disparity as follows [20]:

$$LP = \frac{1}{\left|I^{ref}\right|} \sum_{u,v} \sum_{\mathbf{p} \in I^{ref}} \mathbb{1}\left(L(\mathbf{p}^{u,v}) \neq L(\mathbf{p}^{ref})\right), \quad (22)$$

where $\left|I^{ref}\right|$ is the number of pixels in the *ref* view, L represents the superpixel labeled image, $\mathbf{p}^{u,v}$ represents a projected pixel in view (u, v), $\mathbf{p}^{ref}$ represents a pixel in the *ref* view, $\mathbb{1}()$ is a binary indicator and $\mathbb{1}\left(L(\mathbf{p}^{u,v}) \neq L(\mathbf{p}^{ref})\right) = 1$ indicates that the label of the projected pixel $\mathbf{p}^{u,v}$ has a different label value compared to its label value in the *ref* view. This metric discards the pixels from other views that are occluded in the central view to simplify the computation. A smaller LP error indicates better consistency across views because the corresponding pixels that belong to the same superpixel have the same label across views.

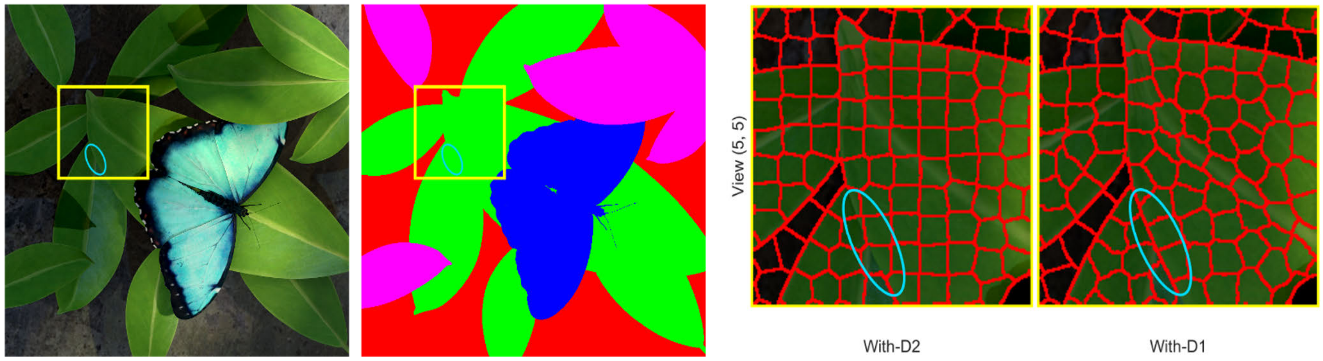

FIGURE 5. Visual results for Papillon test image, with and without using the squared disparity distance in the clustering weighted distance, (With-D2), (With-D1), respectively. Portions of the central view (5, 5) are selected and highlighted on both the test images and the corresponding ground truth label images. The blue oval highlights higher segmentation accuracy in (With-D1) where the overlapping leaves are robustly segmented. $S_{size} = 20$.

D. VISUAL AND QUANTITATIVE RESULTS

In this section, we firstly compare our results with two different versions of the proposed ALFO method, to study the influence of clustering weights adaptation stage and the used disparity map on the performance. In the first version, the used clustering weights are fixed and not adjusted during clustering to study the clustering weights adaptation stage impact. In the second version, the GT disparity is used instead of the estimated disparity, that is used in our proposed method, to study the influence of using an accurate 4D LF disparity map. Quantitative and qualitative results are generated for both versions and compared to the proposed ALFO method. Next, the performance of the proposed method ALFO is evaluated and compared with the benchmark methods.

1) ABLATION STUDIES

Before discussing the two versions of the proposed ALFO method, it is worth to present some intermediate results that justify the weighted distance in (9), where the distances are squared for all features but not for the disparity. Therefore, in this experiment, instead of (9), the following distance is used:

$$D''_w(\mathbf{p},\mathbf{c}) = w_p \times D_p^2 + w_l \times D_l^2 + w_a \times D_a^2 + w_b \times D_b^2 + w_d \times D_d^2, \qquad (23)$$

where the disparity distance is squared, aiming to study its influence on the results. As can be seen in Fig. 5, the overlapping leaves are not segmented robustly when squaring the disparity and the superpixels are not adhering to the light green leaf vein. Although the consistency metrics do not significantly differ in both cases (see Fig. 6 for average quantitative evaluation and Table 3 for specific superpixel size (i.e., 20) where the best results are highlighted with bold font style), the accuracy metrics are noticeably decreased when squaring the disparity, especially for large superpixel sizes. The accuracy is reduced due to the superpixel-flooding over the true object boundaries in the image when the color and position are not enough to segment different regions. While the used features are normalized within [0, 1] range, keeping the disparity unsquared in (9) imposes stronger penalty on disparity feature. Hence, the method will avoid clustering across occlusions and accurately segment overlapping objects with same color but different depths. This approach is also used in [20] where a high weight is assigned to penalize the disparity feature compared to other used features.

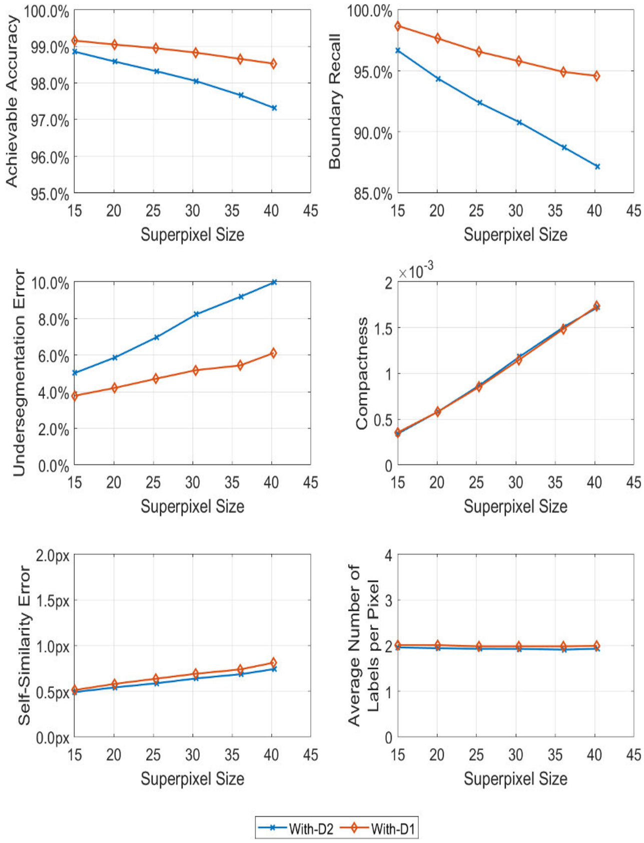

FIGURE 6. Average quantitative evaluation on all LF images of the HCI 4D LF dataset, (With-D2), (With-D1) indicate with and without using the squared disparity distance in the clustering weighted distance, respectively.

Furthermore, we evaluate the proposed ALFO method by implementing two different versions, considering two different test conditions:

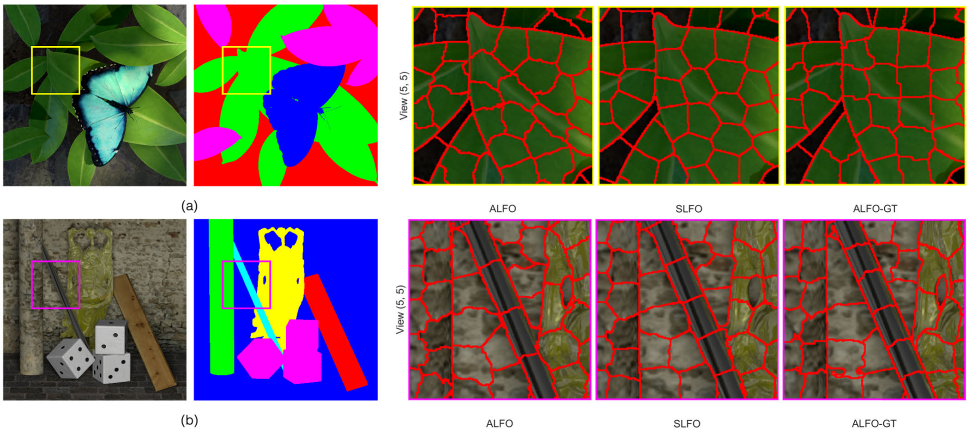

FIGURE 7. Visual results for two test images of the HCI 4D LF dataset for different test conditions of the proposed ALFO method, namely SLFO and ALFO-GT. Portions of the central view (5, 5) are selected and highlighted on both the test images and the corresponding ground truth label images. Adaptive clustering weights with good disparity maps can robustly segment challenging regions, e.g., the silver non-Lambertian region in (b). $S_{size} = 35$.

TABLE 3. Average quantitative evaluation on all LF images of the HCI 4D LF dataset (for superpixel size 20).

	With-D2	With-D1
Achievable accuracy	98.58%	**99.05%**
Boundary recall	94.35%	**97.65%**
Under-segmentation error	0.06	**0.04**
Compactness	0.6×10^{-3}	$\mathbf{0.6 \times 10^{-3}}$
Self-similarity error	**0.54**	0.58
Number of labels per pixel	**1.94**	2.01

TABLE 4. Average quantitative evaluation on all LF images of the HCI 4D LF dataset (for superpixel size 20).

	SLFO	ALFO-GT	ALFO
Achievable accuracy	98.88%	**99.28%**	99.05%
Boundary recall	95.84%	**98.39%**	97.65%
Under-segmentation error	0.05	**0.03**	0.04
Compactness	0.6×10^{-3}	0.5×10^{-3}	$\mathbf{0.6 \times 10^{-3}}$
Self-similarity error	**0.55**	0.63	0.58
Number of labels per pixel	**1.94**	2.14	2.01

- **Static LF Over-segmentation (SLFO)**– This version consists in not using the clustering weights adaptation stage during the K-means clustering. Alternatively, fixed weights (e.g., initial clustering weights) are used and not changed during clustering. Equal clustering weights are used for SLFO to study the influence of the adaptation stage where the initial weights are adjusted.
- **ALFO using GT disparity (ALFO-GT)**– This version consists in using the GT disparity instead of the estimated one for the HCI 4D LF dataset to study the influence of disparity accuracy on the clustering and projection.

Notice that we normalized the used features as described in Section III in all versions. Several superpixel sizes are used to obtain the quantitative results, however, for visual results, superpixels with S_{size} equal to 35 is presented in Fig. 7 for better visual comparison.

According to the visual results shown in Fig. 7, the quantitative results in the form of plot presented in Fig. 8 and the numerical quantitative results for superpixel size 20 in Table 4 (highlighting the best results in bold font style), we may conclude that a significant improvement is achieved on the *AA*, *BR*, *UE* metrics when using adaptive clustering weights associated with accurate disparity maps (i.e., GT disparity maps) as in ALFO-GT. As can be seen in Fig. 7, some challenging regions can be segmented more robustly using ALFO-GT compared to other versions. However, using fixed weights for all test images, without adjusting the weights based on the image content, may generate wrong segmentation (e.g., see the overlapping leaves in Fig. 7a and the small hole in the gold region in Fig. 7b). The SLFO version shows higher *CP* for large superpixel sizes compared to other versions, without genuinely adhering to the borders. According to consistency metrics *SS* and *LP*, no significant difference is noticed since the used consistency metrics consider the non-occluded regions in the central view, where the used disparity has high accuracy in these regions, but some ambiguity exists in the occluded ones.

2) COMPARISON TO BENCHMARK SOLUTIONS

Before comparing our results to the existing methods, it is important to mention that our method does not require any post-processing optimization, since the centroid projection across views is applied robustly by using per-pixel disparity and the clustering weights are optimized in each iteration. In most existing methods, a post-processing stage is needed

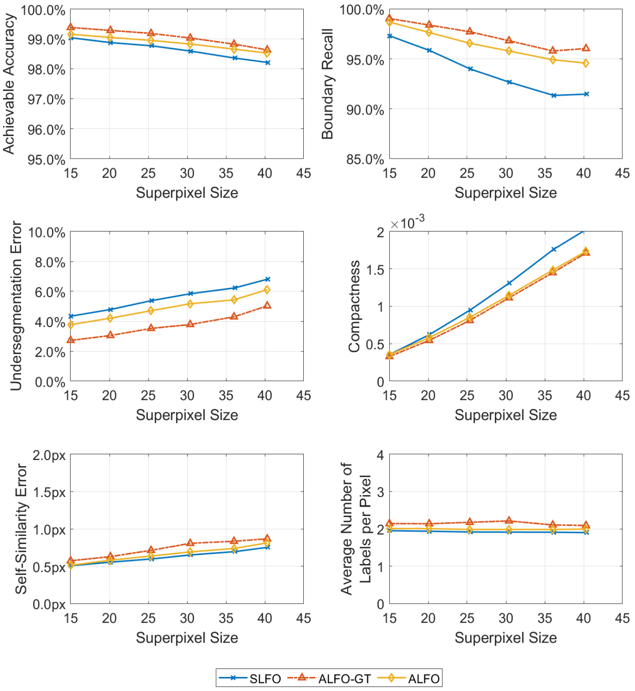

FIGURE 8. Average quantitative evaluation on all LF images of the HCI 4D LF dataset for different test conditions of the proposed method, namely SLFO, ALFO-GT and the proposed ALFO method. Adaptive clustering weights and good disparity maps can improve the segmentation performance.

to remove sparse labels that are wrongly propagated or to smooth superpixels borders and enforce spatial or angular connectivity across views. In our experiments, we compared with other methods without disabling their post-processing step. As shown in Fig. 9, the used clustering weights are adapted based on the image content and adjusted in each iteration until the final weights are reached when the segmentation terminates (see Table 5).

According to the initial values of the used clustering weights, several tests using different initial weights (e.g., giving a higher weight for one feature compared to other features) are applied. We noticed that the initial clustering weights are not crucially impacting the final clustering weights, such as when these weights ($w_l = 0.2, w_a = 0.15, w_b = 0.15, w_p = 0.1, w_d = 0.4$) are used as initial weights, and S_{size} is set to 20, the final clustering weights percentage change on the HCI dataset is less than or equal to 2.0% of the final weights when using equal initial clustering weights, without any significant change on the quantitative evaluation metrics.

To compare our results with the existing methods, different superpixel sizes are used for all methods. However, since we only could obtain labels of the HVLFS method for specific sizes, only the available sizes in the used size range are used in our comparisons. Due to the post-processing stage in some methods, the size of the generated superpixels can be different from the input size (e.g., in some solutions, some superpixels are removed if their sizes, after the segmentation is completed, are smaller than a given threshold). For this reason, and for fair comparisons, the average size of the generated superpixel in each image is used instead of the input superpixel size. The average performance of all the LF images of the HCI dataset is presented in Fig. 10 and per-image performance is presented in Fig. 11.

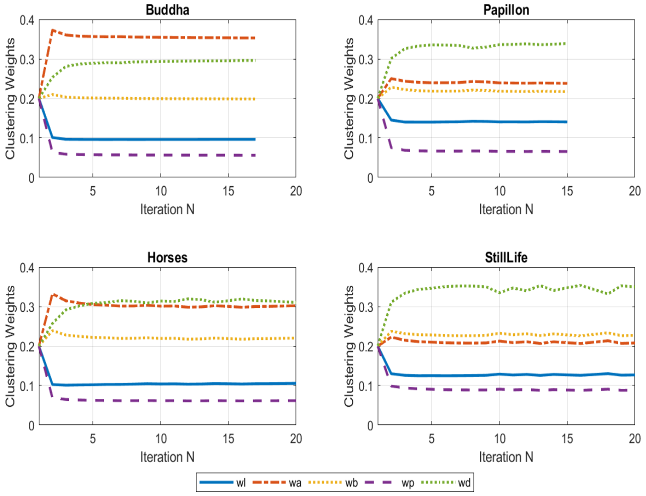

FIGURE 9. Clustering weights adaptation along the number of iterations for different test images. The included weights are w_l, w_a, w_b for color channels, w_p for relative position and w_d for the disparity. $S_{size} = 20$.

TABLE 5. Final clustering weights for different features and test images.

Test image	w_l	w_a	w_b	w_p	w_d
Buddha	0.095	0.355	0.198	0.055	0.297
Papillon	0.140	0.240	0.217	0.064	0.339
Horses	0.105	0.303	0.220	0.061	0.311
StillLife	0.126	0.206	0.226	0.087	0.355

The quantitative evaluation and the visual results in Fig. 10, Fig. 11, Fig. 12 and Table 6 (bold font style for best results) can be summarized based on evaluation metrics as follows:

- **Achievable accuracy** – Our proposed method achieves outperforming average *AA* for all superpixel sizes compared to the benchmark methods. The importance of using the disparity feature during the clustering can be observed in Fig. 12b and Fig. 12c, where the overlapping regions share the same color information; hence it cannot be accurately segmented in the Superrays or LFSP methods. The HVLFS method accurately segmented the leaves in Fig. 12b since the depth information is used during the clustering. However, in Fig. 12c, the method fails to segment the horses' heads correctly due to the limitation in balancing the importance of the used features to generate robust segmentation.
- **Boundary recall** – Our proposed method achieves outperforming average *BR* compared to the benchmark methods and competitive results to the VCLFS method.

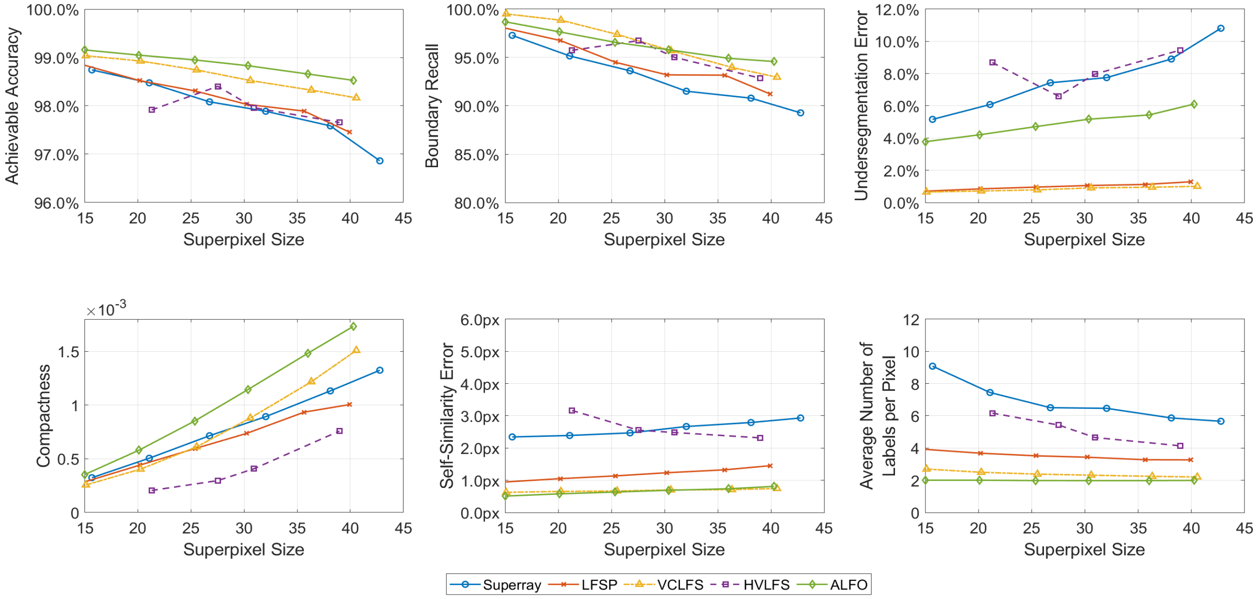

FIGURE 10. Average quantitative evaluation on all LF images of the HCI 4D LF dataset for different 4D LF superpixel segmentation methods.

TABLE 6. Average quantitative evaluation on all LF images of the HCI 4D LF dataset (for superpixel size 20).

	Superray	LFSP	VCLFS	HVLFS	ALFO
Achievable accuracy	98.48%	98.52%	98.93%	97.92%	**99.05%**
Boundary recall	95.16%	96.75%	**98.85%**	95.74%	97.65%
Under-segmentation error	0.06	**0.01**	0.01	0.08	0.04
Compactness	0.5×10^{-3}	0.4×10^{-3}	0.4×10^{-3}	0.2×10^{-3}	$\mathbf{0.6 \times 10^{-3}}$
Self-similarity error	2.39	1.05	0.65	3.17	**0.58**
Number of labels per pixel	7.45	3.68	2.50	6.17	**2.01**

Our results are competitive to the VCLFS method since the per-pixel disparity is used during the clustering in both methods. In Fig. 12a, our results recall boundaries across views even in the small black circus. Moreover, in Fig. 12c, only our method and the VCLFS method adhere to the actual boundaries of the horses.

- **Under-segmentation error** – Our proposed method achieves outperforming *UE* compared to the Superrays and HVLFS methods. However, the LFSP and VCLFS methods achieve lower UE error (e.g., each superpixel is less likely to include more than one object) but not necessarily with better accuracy or compactness as mentioned above and can be seen visually in Fig. 12.
- **Compactness** – Our proposed method achieves outperforming *CP* for all superpixel sizes compared to the benchmark methods. Our method encourages spatial and angular connectivity through robust projection and local searching. Moreover, the adaptation stage adjusts the clustering weight of the position, hence can control the superpixel boundaries to be smoother and more coherent across views. As can be seen in the yellow ball in Fig. 12d, where our results show more regular shapes and smoother borders.
- **Self-similarity and number of labels per pixel** – Our proposed method achieves outperforming *SS* and *LP* compared to the benchmark methods and competitive results to the VCLFS method. Superpixel consistency can be clearly noticed from the dynamic results in the supplemental material, where the flickering and label change across views can be noticed easily. Visually, our results preserve angular consistency and the superpixels borders are less likely to flicker, when changing the angular perspective, compared to the benchmark methods. We tried to show the consistency metrics by presenting the same patch from different LF views. As in Fig. 12 for all images, our results are consistent and similar across views. Generating consistent superpixels is a crucial requirement for subsequent editing tasks.

For the real LF images dataset, since there are no GT segmentation labels available, we only make a visual comparison of our method, the Superrays, LFSP, and VCLFS methods for various representative test images, S_{size} is set to 20. The HVLFS method is not evaluated in this experiment

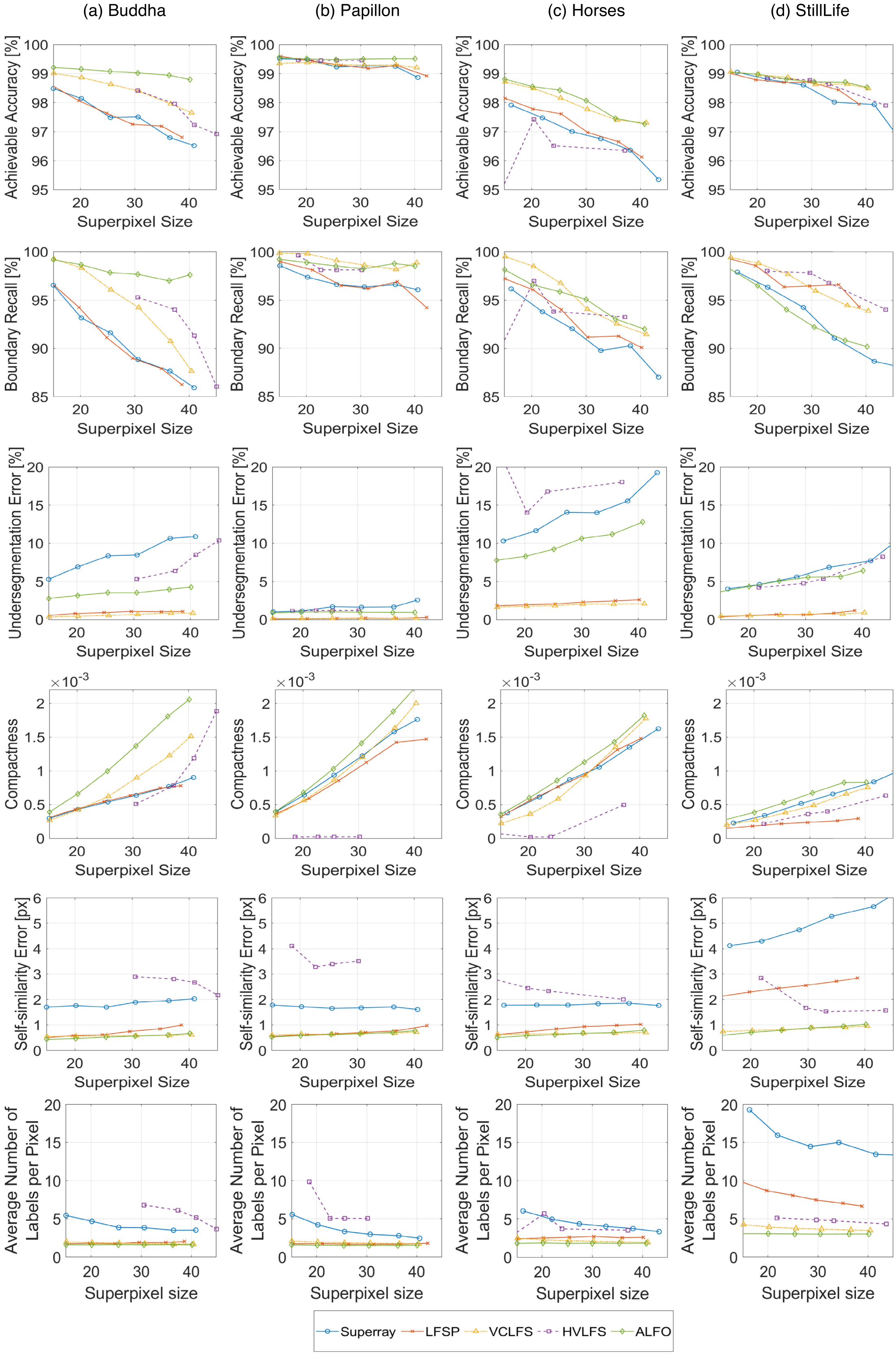

FIGURE 11. Per-image quantitative evaluation on the HCI 4D LF dataset for different 4D LF superpixel segmentation methods.

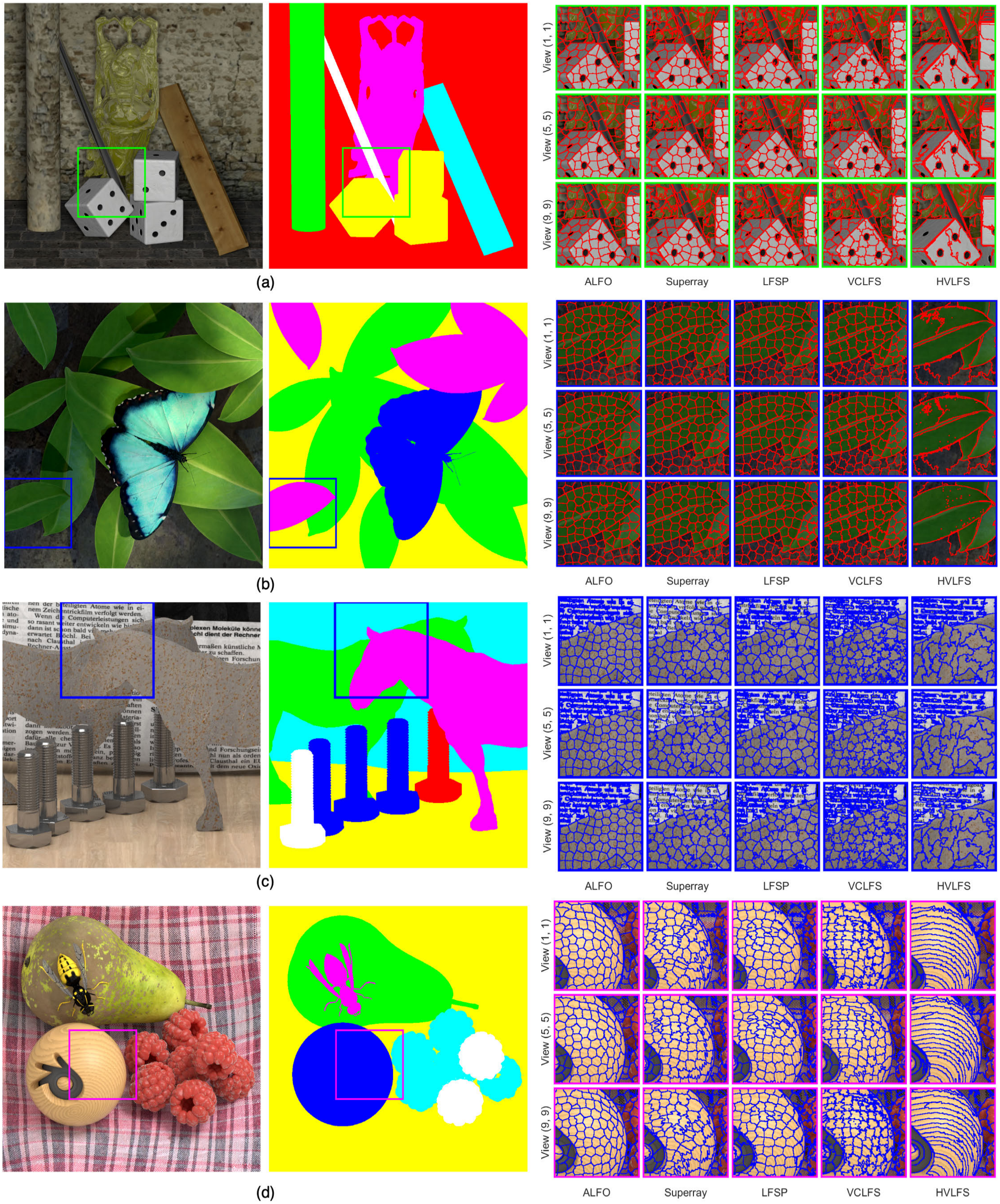

FIGURE 12. Visual results to evaluate accuracy, compactness and consistency across views for the proposed ALFO, Superray, LFSP, VCLFS and HVLFS methods on the HCI 4D LF dataset. Challenging regions (highlighted on both the test images and the corresponding ground truth label images) are selected to show the importance of the adaptive clustering weights: a) non-Lambertian and shaded regions; b) overlapping leaves with the same color and different depths; c) a complex background and overlapping cardboard horses sharing the same texture; d) a spherical region with non-even lighting. As can be seen, our method can robustly and adaptively segment similar color regions with different depths and reduce the flickering around superpixels, hence generates not only superpixels that are compact but also accurate and consistent across views. $S_{size} = 20$.

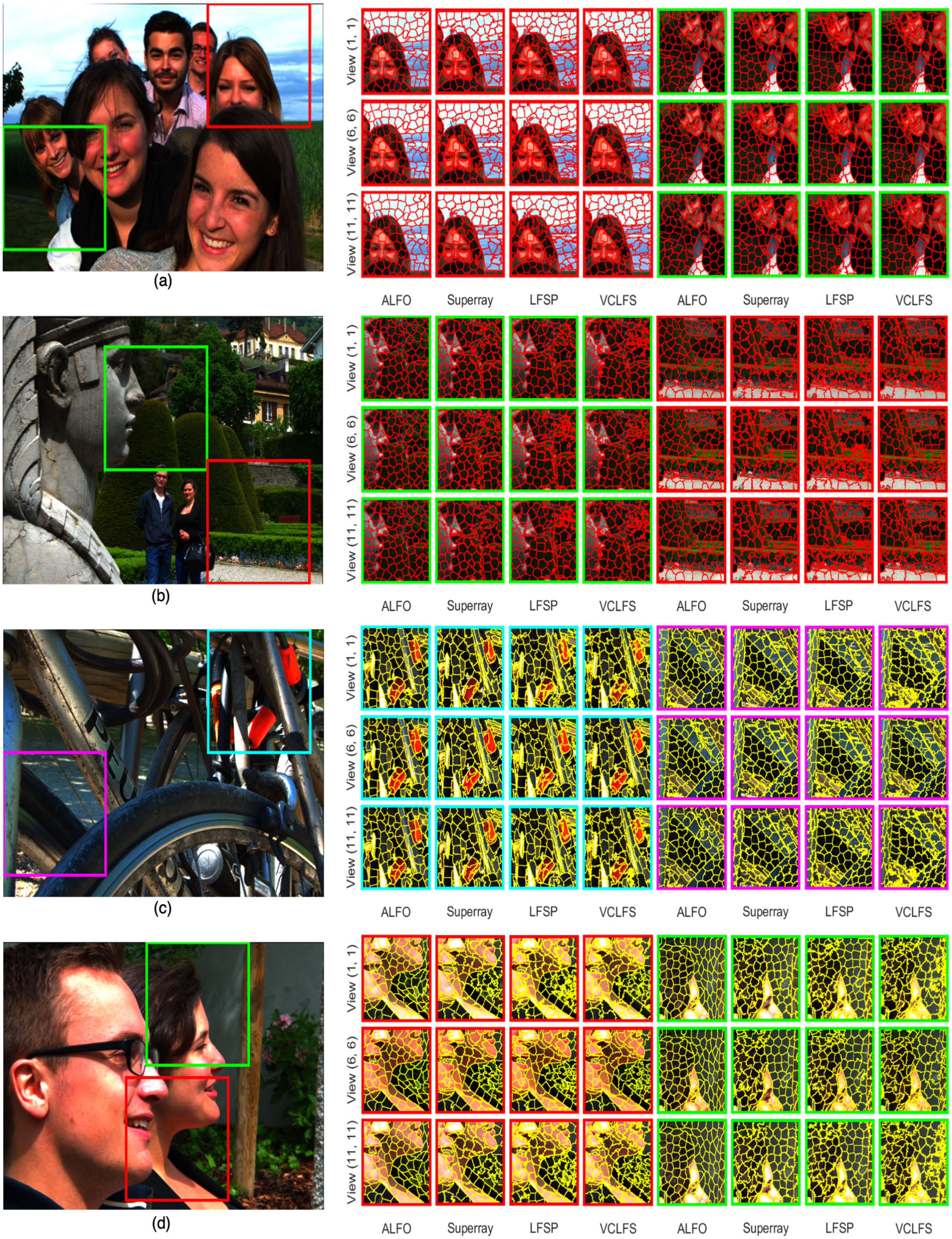

FIGURE 13. Visual results to evaluate accuracy, compactness and consistency across views for the proposed ALFO, Superray, LFSP and VCLFS methods on the MMSPG LF dataset. In real 4D LF images, the noise existence and non-even lighting generally can affect the segmentation accuracy or generate flickering borders around superpixels. However, with the adaptive clustering weights to optimize the segmentation based on the content of each image, our method shows compact and consistent superpixels compared to the benchmark methods. Due to the vignetting issue in the corner views of this dataset, only the central 13 × 13 views are used to generate superpixels. $S_{size} = 20$.

since only the labels of the HCI 4D LF dataset are available. We strongly encourage the reader to see the dynamic results in the supplemental material, where the performance in terms of accuracy and cross-view consistency can be noticed easily. As can be seen in Fig. 13, the existence of complex texture and noise in the real LF image can affect the regularity and accuracy of superpixels in the existing solutions, where the borders of superpixels may flicker across views. However, our results generate more compact and accurate superpixels as shown in Fig. 13, where the superpixels in the woman's hair, the trees in the background, the bike parts and in the face patch are more regular and consistent when compared to other methods. In Fig. 13c, a challenging region with non-even lighting and a non-Lambertian object are selected. Our results show better consistency, which can be observed from the red parts in Fig. 13c. However, the light in the floor in Fig. 13c (see pink square) is different across views and, hence, may lead to inconsistent superpixels, as is the case for the benchmark results. More results for real LF images can be found in the dynamic results available in the supplemental material. In general, for complex textures in real LF images, our proposed method can balance between compactness, accuracy and cross-view consistency instead of generating superpixels that are extremely sensitive to color changes with irregular or flickering borders when changing the view perspective.

The proposed method is implemented using MATLAB on a desktop computer with Intel i7 4 GHz processor and 32 GB RAM. Our implementation is not optimized and, for this reason, consumes more time, compared to the benchmark methods, since the clustering is performed for each light field view and not merely propagated from the central view as in some existing solutions. The average computational cost (i.e., execution time in seconds) of generating 4D LF superpixels for all LF views is presented in Table 7 for different superpixel sizes and datasets. The computational cost of the HVLFS method is not included since we only have the generated results from the author but not the software implementation. Our implementation takes more time for images with complex textures since it requires more clustering iterations due to the frequent adjusting of the clustering weights and the labels of the pixels until convergence is reached. Additionally, in most test images, it requires more time for smaller superpixel sizes since the clustering includes more superpixels and requires more comparisons to assign the accurate label for each pixel according to the corresponding superpixel. Since K-means clustering in local searching can be parallelized, as shown in [7] for the proposed 2D superpixel method, it is expected that our method can be further optimized, especially considering that clustering is done independently in each view.

TABLE 7. Average Segmentation time in seconds for different over-segmentation methods.

S_{size}	dataset	ALFO	Superray	LFSP	VCLFS
20	HCI	746.65	108.55	91.25	222.09
20	MMSPG	616.20	59.38	58.37	125.56
40	HCI	813.75	83.25	71.84	178.97
40	MMSPG	541.52	48.85	48.01	104.81

V. DISCUSSION AND LIMITATIONS

The proposed ALFO method produces competitive results in several challenging cases such as overlapping objects with the same color but different depths (see Fig. 12c), and can segment accurately, consistently and adaptively the small parts that are smaller than the initial/target superpixel size (see the dice black dots in Fig. 12a) without any need for post-processing smoothing or cross-view regularization steps, when compared to most of the existing methods. Additionally, using disparity values for each pixel during the over-segmentation helps in improving the superpixel accuracy and consistency for non-Lambertian objects where the color can change according to each view perspective. The mentioned advantages can be noticed in the dynamic results in the supplemental material where the superpixels are accurately adhering to the boundaries and not flickering across views.

However, the ALFO method still has some limitations that can be further improved. First, in real LF images, where the disparity maps are affected by noise or non-even lighting across views, ALFO may generate an imprecise segmentation and superpixels may not adhere well to the boundaries when there are disparity ambiguities. Hence, better disparity maps will lead to better performance. Second, non-Lambertian objects have a non-uniform appearance across views due to the non-even lighting in each view perspective. In the EPI space, these non-Lambertian objects present more complex and non-linear features, characterized by curved lines [38]. Although, in our method, we are not enforcing superpixel consistency in the EPI space by exploiting the assumption of linearity in EPI lines (as in other light field over-segmentation methods [19]–[21]), our method may still generate inaccurate or inconsistent segmentation in some non-Lambertian areas. The mentioned limitations can be noticed in the dynamic results for all views in the supplemental material where, in some regions that include a metallic material or non-even lighting, the superpixels may not adhere to the borders accurately across views. Third, our implementation, including K-means clustering, is not optimized and may take more time compared to other methods. However, K-means clustering in local searching can be parallelized, as shown in [7] for the proposed 2D superpixel method and in [8] for 4D LF images; hence, it is expected that our method can be further optimized to generate faster over-segmentation and reduce the overall subsequent editing complexity (this optimization is out of scope of the present work). Finally, similarly to the benchmark methods, we assume that the centroids in the central view exist in other views. Since this assumption may not hold for LF images captured by wide baseline cameras, where new centroids can exist in other views and some centroids in the central view are completely occluded in other views, our method may fail to segment this type of sparse LF images accurately.

VI. CONCLUSION

In this paper, we proposed an automatic content-adaptive LF over-segmentation method. Using hybrid and normalized 4D LF features along with adaptive clustering weights, our method achieves a robust balance between accuracy, compactness and cross-view consistency of superpixels. More precisely, the estimated disparity for entire 4D LF views is used jointly with color and position features during clustering to overcome the limitation in some challenging regions where color information is not enough for segmentation. Due to the different nature and ranges of the used features, the clustering weights are adapted to the given content iteratively until convergence is reached. Experimental results showed competitive results, quantitatively and visually outperforming the benchmark methods, without requiring any empirical assignment for the clustering weights or any post-processing optimization. Additionally, it was shown that the proposed ALFO method can benefit from accurate disparity maps and the performance is relatively independent of the initial clustering weights adopted.

In the future, we will apply the proposed method in different applications, such as object segmentation and saliency detection. Additionally, we will further consider adapting the final superpixel size to generate an adequate number of superpixels based on the image content. Furthermore, we will exploit deep learning techniques to generate superpixels for 4D LF images, since it has shown promising results for 2D over-segmentation.

ACKNOWLEDGMENT

The authors would like to thank Dr. Mira Rizkallah for providing her re-implementation of the Superray software and Dr. Rui Li for providing his generated labels from his method for them to compare with. They would also like to thank Numair Khan for publishing the software of all the used evaluation metrics that facilitated their comparisons.

REFERENCES

[1] A. B. M. Faruquzzaman, N. R. Paiker, J. Arafat, Z. Karim, and M. A. Ali, "Object segmentation based on split and merge algorithm," in *Proc. IEEE Region Conf.*, Hyderabad, India, Nov. 2008, pp. 1–4.

[2] T. Shen and Y. Wang, "Medical image segmentation based on improved watershed algorithm," in *Proc. IEEE 3rd Adv. Inf. Technol., Electron. Autom. Control Conf. (IAEAC)*, Chongqing, China, Oct. 2018, pp. 1695–1698.

[3] I. Cinaroglu and Y. Bastanlar, "Image based localization using semantic segmentation for autonomous driving," in *Proc. 27th Signal Process. Commun. Appl. Conf. (SIU)*, Sivas, Turkey, Apr. 2019, pp. 1–4.

[4] C. J. Mathew, R. C. Shinde, and C. Y. Patil, "Segmentation techniques for handwritten script recognition system," in *Proc. Int. Conf. Circuits, Power Comput. Technol.*, Nagercoil, India, Mar. 2015, pp. 1–7.

[5] W. Byeon and T. M. Breuel, "Supervised texture segmentation using 2D LSTM networks," in *Proc. IEEE Int. Conf. Image Process. (ICIP)*, Paris, France, Oct. 2014, pp. 4373–4377.

[6] X. Lv, X. Wang, Q. Wang, and J. Yu, "4D light field segmentation from light field super-pixel hypergraph representation," *IEEE Trans. Vis. Comput. Graphics*, vol. 27, no. 9, pp. 3597–3610, Sep. 2021.

[7] R. Achanta, A. Shaji, K. Smith, A. Lucchi, P. Fua, and S. Süsstrunk, "SLIC superpixels compared to state-of-the-art superpixel methods," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 34, no. 11, pp. 2274–2282, Nov. 2012.

[8] M. Hog, N. Sabater, and C. Guillemot, "Superrays for efficient light field processing," *IEEE J. Sel. Topics Signal Process.*, vol. 11, no. 7, pp. 1187–1199, Oct. 2017.

[9] D. Stutz, A. Hermans, and B. Leibe, "Superpixels: An evaluation of the state-of-the-art," *Comput. Vis. Image Underst.*, vol. 166, pp. 1–27, Jan. 2018.

[10] V. Jampani, D. Sun, M.-Y. Liu, M.-H. Yang, and J. Kautz, "Superpixel sampling networks," in *Proc. Eur. Conf. Comput. Vis. (ECCV)*, Munich, Germany, 2018, pp. 352–368.

[11] F. Yang, Q. Sun, H. Jin, and Z. Zhou, "Superpixel segmentation with fully convolutional networks," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Seattle, WA, USA, Jun. 2020, pp. 13961–13970.

[12] X. Luo, "Image compression via K-means and SLIC superpixel approaches," in *Proc. 4th Int. Conf. Mach., Mater. Inf. Technol. Appl.*, Paris, France, 2016, pp. 1008–1012.

[13] C. Conti, L. D. Soares, and P. Nunes, "Dense light field coding: A survey," *IEEE Access*, vol. 8, pp. 49244–49284, 2020.

[14] D. Yeo, J. Son, B. Han, and J. H. Han, "Superpixel-based tracking-by-segmentation using Markov chains," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Honolulu, HI, USA, Jul. 2017, pp. 511–520.

[15] X. Xie, G. Xie, X. Xu, L. Cui, and J. Ren, "Automatic image segmentation with superpixels and image-level labels," *IEEE Access*, vol. 7, pp. 10999–11009, Jan. 2019.

[16] Y. Yan and J. Zhu, "Saliency detection based on superpixel correlation and cosine window filtering," *Multimedia Tools Appl.*, vol. 78, no. 15, pp. 21205–21221, Aug. 2019.

[17] M. Levoy and P. Hanrahan, "Light field rendering," in *Proc. 23rd Annu. Conf. Comput. Graph. Interact. Techn.*, 1996, pp. 31–42.

[18] G. Wu, B. Masia, A. Jarabo, and Y. Zhang, "Light field image processing: An overview," *IEEE J. Sel. Topics Signal Process.*, vol. 11, no. 7, pp. 926–954, Oct. 2017.

[19] H. Zhu, Q. Zhang, Q. Wang, and H. Li, "4D light field superpixel and segmentation," *IEEE Trans. Image Process.*, vol. 29, pp. 85–99, 2020.

[20] N. Khan, Q. Zhang, L. Kasser, H. Stone, M. H. Kim, and J. Tompkin, "View-consistent 4D light field superpixel segmentation," in *Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV)*, Seoul, South Korea, Oct. 2019, pp. 7810–7818.

[21] R. Li and W. Heidrich, "Hierarchical and view-invariant light field segmentation by maximizing entropy rate on 4D ray graphs," *ACM Trans. Graph.*, vol. 38, no. 6, pp. 1–15, Nov. 2019.

[22] X. Xiao, Y. Zhou, and Y.-J. Gong, "Content-adaptive superpixel segmentation," *IEEE Trans. Image Process.*, vol. 27, no. 6, pp. 2883–2896, Jun. 2018.

[23] R. Uziel, M. Ronen, and O. Freifeld, "Bayesian adaptive superpixel segmentation," in *Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV)*, Seoul, South Korea, Oct. 2019, pp. 8469–8478.

[24] N. Khan, M. H. Kim, and J. Tompkin, "View-consistent 4D light field depth estimation," in *Proc. Brit. Mach. Vis. Conf. (BMVC)*, 2020.

[25] X. Ren and J. Malik, "Learning a classification model for segmentation," in *Proc. 9th IEEE Int. Conf. Comput. Vis.*, Nice, France, vol. 1, Oct. 2003, pp. 10–17.

[26] M. Wang, X. Liu, Y. Gao, X. Ma, and N. Q. Soomro, "Superpixel segmentation: A benchmark," *Signal Process. Image Commun.*, vol. 56, pp. 28–39, Aug. 2017.

[27] M. Hog, N. Sabater, and C. Guillemot, "Dynamic super-rays for efficient light field video processing," in *Proc. Brit. Mach. Vis. Conf. (BMVC)*, Newcastle, U.K., 2018, pp. 1–12.

[28] H. Zhu, Q. Zhang, and Q. Wang, "4D light field superpixel and segmentation," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Honolulu, HI, USA, Jul. 2017, pp. 6709–6717.

[29] R. C. Bolles, H. H. Baker, and D. H. Marimont, "Epipolar-plane image analysis: An approach to determining structure from motion," *Int. J. Comput. Vis.*, vol. 1, no. 1, pp. 55–57, 1987.

[30] J. Han, M. Kamber, and J. Pei, "Data pre-processing," in *Data Mining: Concepts and Techniques*, 3rd ed. San Francisco, CA, USA: Morgan Kaufmann, 2011, ch. 3, pp. 114–115.

[31] Mathworks. *MATLAB Function to Convert RGB to CIE 1976*. Accessed: May 1, 2021. [Online]: Available:https://www.mathworks.com/help/images/ref/rgb2lab.html

[32] J. Shi, X. Jiang, and C. Guillemot, "A framework for learning depth from a flexible subset of dense and sparse light field views," *IEEE Trans. Image Process.*, vol. 28, no. 12, pp. 5867–5880, Dec. 2019.

[33] S. Wanner, S. Meister, and B. Goldlüecke, "Datasets and benchmarks for densely sampled 4D light fields," *Vis., Model. Vis.*, vol. 13, pp. 225–226, Sep. 2013.

[34] M. Rerabek and T. Ebrahimi, "New light field image dataset," in *8th Int. Conf. Qual. Multimedia Exper. (QoMEX)*, Lisbon, Portugal, 2016, pp. 1–2.

[35] H. Zhu, Q. Wang, and J. Yu, "Occlusion-model guided antiocclusion depth estimation in light field," *IEEE J. Sel. Topics Signal Process.*, vol. 11, no. 7, pp. 965–978, Oct. 2017.

[36] M. Rizkallah, X. Su, T. Maugey, and C. Guillemot, "Geometry-aware graph transforms for light field compact representation," *IEEE Trans. Image Process.*, vol. 29, pp. 602–616, 2020.

[37] P. Neubert and P. Protzel, "Superpixel benchmark and comparison," in *Proc. Forum Bildverarbeitung*, 2012, pp. 1–12.

[38] Fachada, D. Bonatto, M. Teratani, and G. Lafruit, "Light field rendering for non-lambertian objects," in *Proc. Electron. Imag. Symp.*, 2021, pp. 54-1–54-8.

MARYAM HAMAD (Graduate Student Member, IEEE) received the B.E. degree in computer systems engineering (CSE) from Palestine Technical University-Kadoorie (PTUK), Palestine, in 2018, covered by an excellence scholarship. She is currently pursuing the fully granted Ph.D. degree with the Instituto Universitário de Lisboa (ISCTE-IUL), Portugal. During her degree, she spent one semester as an Exchange Student with Middle East Technical University (METU) with ERASMUS+ Program, Turkey. She completed her professional internship in information science and technology with IAESTE Program as a Researcher with the Multimedia Signal Processing Group, Instituto de Telecomunicações, Portugal, where she is also a Researcher. Her current research interests involve immersive visual technologies, such as light field imaging, digital image processing, and computer vision. She is a member of the IEEE Women in Engineering Society, the IEEE Signal Processing Society, and the IEEE Young Professionals Group.

CAROLINE CONTI (Member, IEEE) received the B.Sc. degree in electrical engineering from the Universidade de São Paulo (USP), Brazil, in 2010, and the Ph.D. degree in information science and technology from the Instituto Universitário de Lisboa (ISCTE-IUL), Portugal, in 2017. She is currently a Researcher with the Multimedia Signal Processing Group, Instituto de Telecomunicações, Portugal. She is also an Assistant Professor with the Information Science and Technology Department, Instituto Universitário de Lisboa (ISCTE-IUL), Portugal. She has been a Postdoctoral Researcher with the Multimedia Signal Processing Group, Instituto de Telecomunicações. She has contributed more than 20 papers to international journals and conferences in these areas. In addition, she has participated in many national and international projects related to light field processing and coding. Her research interests include immersive visual technologies and image and video processing, including light field processing and coding. She also acts as a reviewer for various IEEE and EURASIP journals and conferences.

PAULO NUNES (Member, IEEE) graduated in electrical and computers engineering from the Instituto Superior Técnico (IST), Universidade Técnica de Lisboa, Portugal, in 1992, and the M.Sc. and Ph.D. degrees in electrical and computers engineering from IST, in 1996 and 2007, respectively. He is currently a Senior Researcher with the Multimedia Signal Processing Group, Instituto de Telecomunicações, Portugal. He is also an Associate Professor with the Information Science and Technology Department, Instituto Universitário de Lisboa (ISCTE-IUL), Portugal. He has coordinated and participated in various national and international (EU) funded projects and has acted as a Project Evaluator of the European Commission. He has contributed more than 65 papers to international journals and conferences in these areas. His current research interests include 2D/3D image and video processing and coding, namely light field image and video processing and coding. He acts often as a reviewer for various ACM, EURASIP/Elsevier, IEEE, IET, MDPI, SPIE, and Springer conferences and journals and member of the program and organizing committees of various international conferences.

LUÍS DUCLA SOARES (Senior Member, IEEE) received the Licenciatura and Ph.D. degrees in electrical and computer engineering from the Instituto Superior Técnico (IST), Universidade Técnica de Lisboa, Portugal, in 1996 and 2004, respectively. He is currently a Senior Researcher with the Multimedia Signal Processing Group, Instituto de Telecomunicações, Portugal. He is also an Associate Professor with the Information Science and Technology Department, Instituto Universitário de Lisboa (ISCTE-IUL), Portugal. His research interests include image and video coding/processing, including light field coding and processing as well as biometric recognition. He has contributed more than 65 papers to international journals and conferences in these areas (20 of which on light field coding). In addition, he has participated in the development of the MPEG-4 visual standard, as well as in several national and international projects. He is a member of the Editorial Board of the *EURASIP Signal Processing* (Elsevier) journal. In parallel, he acts as a reviewer for several IEEE, IET, and EURASIP journals and conferences.