INSTITUTO
UNIVERSITÁRIO
DE LISBOA

Light Field Processing for Immersive Systems

Maryam Faleh Awad Hamad

PhD in Information Science and Technology, specialization in Multimedia Signal Processing

Supervisors:

Doctor Luís Eduardo de Pinho Ducla Soares, Associate Professor with Habilitation,
Iscte - Instituto Universitário de Lisboa

Doctor Paulo Jorge Lourenço Nunes, Associate Professor,
Iscte - Instituto Universitário de Lisboa

October, 2024

TECNOLOGIAS
E ARQUITETURA

Department of Information Science and Technology

Light Field Processing for Immersive Systems

Maryam Faleh Awad Hamad

PhD in Information Science and Technology, specialization in Multimedia Signal Processing

Jury:
Doctor Tomás Almeida e Silva Martins Alves, Integrated Investigator, Iscte - Instituto Universitário de Lisboa (President)
Doctor Luís Alberto da Silva Cruz, Assistant Professor, Universidade de Coimbra
Doctor Sérgio Manuel Maciel de Faria, Coordinator Professor, Instituto Politécnico de Leiria
Doctor Catarina Isabel Carvalheiro Brites, Assistant Professor, Iscte - Instituto Universitário de Lisboa
Doctor Luís Eduardo de Pinho Ducla Soares, Associate Professor with Habilitation, Iscte - Instituto Universitário de Lisboa

October, 2024

"All our dreams can come true if we have the courage to pursue them."

– Walt Disney

"Success is not final, failure is not fatal: it is the courage to continue that counts."

– Winston Churchill

Acknowledgment

First and foremost, I thank almighty God for the countless blessings He has given me. I am very grateful for His wise guidance in all aspects of my life.

I would like to express my profound appreciation to my PhD supervisors, Prof. Luís Ducla Soares and Prof. Paulo Nunes. Even in tough times, their big hearts, wise advice, and positive perspectives greatly influenced my PhD journey. Their vast expertise, constructive criticism, and feedback encouraged me, empowered my critical thinking and helped me to become a better researcher. I will never forget our meetings; expected to be short but lasted for hours without realizing how time flew. I could not have hoped for better supervisors. I would also like to express my appreciation to Prof. Caroline Conti for her dedication to our weekly meetings and for inspiring me with her unique perspective on my PhD work.

I extend my gratitude to the staff at Instituto de Telecomunicações and the PhD secretariat at ISCTE-IUL who helped me with the administrative work. Special thanks to Tereza Traquinas, Sara Correia, and Fátima Estevens for their valuable help.

I would like to acknowledge the generous financial support by Fundação para a Ciência e a Tecnologia (FCT/MCTES) and Instituto de Telecomunicações for funding this PhD (under projects UIDB/50008/2020 and PTDC/EEI-COM/7096/2020).

I would like to send my thanks to my faithful friends around the world. They kept calling and supporting me overseas, especially in the loneliest times during the Covid-19 lockdown. I am also grateful to my amazing friends whom I met in Portugal and who were like family and shared with me unforgettable moments. Special thanks to Sheyma Khemiri, Alaa Nakishbndi, Menna Wajeh, and my beloved friends at Instituto de Telecomunicações. For all my dearest friends, thank you very much for being near when I needed you.

I would like to extend my thanks to the ones to whom I dedicate my Thesis, those who believed in me and are always supportive and proud, my lovely family. Special thanks to my dear parents for their unconditional love and endless support, my adorable siblings overseas Dheya, Fatima, Rawan and their families, and my loyal and supportive husband Osama and his family who all provided emotional support overseas until I achieved my Thesis.

I am also grateful for living in Portugal, which felt like home from day one. The happy colors, great weather, breathtaking scenes and the friendly people who were always welcoming and ready to help. Lastly, I am grateful to my origin country, Palestine, which taught me to be resilient, never give up or lose faith, and always see the light inside the darkness.

Resumo

A imagiologia de campo de luz (LF) é uma modalidade de imagem imersiva que tem atraído cada vez mais atenção nas últimas décadas, devido à sua capacidade de captar a intensidade e a informação sobre a direção da luz de uma cena numa matriz de quatro dimensões (4D), conhecida como LF 4D. A vasta informação incluída nos LF 4D permite ao espetador explorar a cena a partir de diferentes perspectivas, melhorando assim a perceção da profundidade e o realismo. No entanto, a promessa de uma experiência imersiva traz consigo desafios que precisam de ser investigados, nomeadamente em termos de processamento e edição.

Um exemplo desses desafios é o processamento e a edição eficientes de LFs 4D, explorando as correlações espácio-angulares e assegurando simultaneamente a precisão espacial e a consistência angular. Assim, esta Tese aborda este desafio através de vários métodos que, em conjunto, formam um *pipeline* para processar/editar eficientemente LFs 4D. Em primeiro lugar, esta Tese propõe um método eficiente de propagação de disparidades que permite calcular mapas de disparidades angularmente consistentes para todas as vistas de LF. Posteriormente, esta Tese propõe novos métodos de sobre-segmentação que se baseiam em mapas de disparidade como uma caraterística adicional de orientação para agrupar pixéis correspondentes em vistas LF em segmentos espácio-angulares. Os LFs 4D sobre-segmentados são então utilizados como uma representação intermédia que permite a segmentação dos LFs e facilita a propagação da transferência neuronal de estilo.

Nesta Tese, foi demonstrado que a representação de LFs 4D com base na sobre-segmentação permite a utilização de técnicas clássicas de corte de grafos e de redes neuronais de grafos para obter uma segmentação de LF 4D eficiente. Os métodos propostos mostraram um desempenho superior em vários aspectos, como a precisão espacial e a consistência angular.

Palavras-Chave: Campo de luz, modalidades de imagiologia imersiva, estimativa de disparidade, sobre-segmentação, segmentação de objectos, transferência neuronal de estilo, consistência angular, consistência de vista

Abstract

Light Field (LF) imaging is an immersive imaging modality that has attracted increasing attention in recent decades, due to its ability of capturing both light intensity and direction information of a scene in a Four-Dimensional (4D) array, known as the 4D LF. The rich information included in 4D LFs enables the viewer to explore the scene from different perspectives, hence, enhancing depth perception and realism. However, the promise of an immersive experience comes with challenges that need to be investigated, notably in terms of processing and editing.

One example of those challenges is to efficiently process/edit 4D LFs by exploiting the spatio-angular correlations while ensuring spatial accuracy and angular consistency. Therefore, this Thesis tackles this challenge through several methods that together form a pipeline to efficiently process/edit 4D LFs. At first, this Thesis proposes an efficient disparity propagation method that enables computing angularly consistent disparity maps for all LF views. Afterwards, this Thesis proposes novel over-segmentation methods that rely on disparity maps as an additional guiding feature to group corresponding pixels across LF views into spatio-angular segments. The over-segmented 4D LFs are then used as an intermediate representation that enables LF segmentation and facilitates neural style transfer propagation.

In this Thesis, it has been shown that representing 4D LFs based on over-segmentation allows the usage of classical graph cut and graph neural networks to achieve efficient 4D LF segmentation. The proposed processing and editing LF methods have shown outperforming performance in several aspects, such as spatial accuracy and angular consistency.

Keywords: Light field, immersive imaging modalities, disparity estimation, over-segmentation, object segmentation, neural style transfer, angular consistency, view-consistency

Table of Contents

List of Figures

List of Acronyms

2D	Two-Dimensional
3D	Three-Dimensional
4D	Four-Dimensional
7D	Seven-Dimensional
AA	Achievable Accuracy
AI	Artificial Intelligence
ALFO	Adaptive Light Field Over-segmentation
ALFO-GT	Adaptive Light Field Over-segmentation using Ground Truth disparity
AR	Augmented Reality
BP	Bad Pixels
BR	Boundary Recall
CNN	Convolutional Neural Network
CP	Compactness
DoF	Degrees of Freedom
EPI	Epipolar Plane Image
EPINet	EPI-based fully-convolutional neural Network
FoV	Field of View
GACS	Global Angular Consistency Stylization
GNN	Graph Neural Network
GraphSAGE	Graph Sampling and Aggregation
GT	Ground Truth
HMD	Head-Mounted Display
HSV	Hue, Saturation and Value
HVLFS	Hierarchical and View invariant Light Field Segmentation
IP	Inconsistent Pixels
IVS	Independent View Stylization
KLD	Kullback-Leibler Divergence

LACS	Local Angular Consistency Stylization
LBP	Local Binary Pattern
LF	Light Field
LFAC	Light Field Angular Consistency
LFEC	Light Field Epipolar Consistency
LFSGNN	LF Segmentation method using Graph Neural Networks
LFSP	Light Field SuperPixel
LiDAR	Light Detection and Ranging
LLFAC	Labeling–Light Field Angular Consistency
LLP	Local Labels per Pixel
LP	Labels per Pixel
MANet	Multi-scale Aggregated Network
mIoU	mean Intersection over Union
MLA	MicroLens Array (a.k.a. lenslet array)
MRF	Markov Random Field
MSE	Mean Square Error
NeRF	Neural Radiance Field
NST	Neural Style Transfer
PVS	Pseudo Video Sequence
SAC	Segmentation Angular Consistency
SAI	Sub-Aperture Image
SGD	Stochastic Gradient Descent
SLFO	Static Light Field Over-segmentation
SLFS	Semi-supervised 4D Light Field Foreground-background Segmentation
SLIC	Simple Linear Iterative Clustering
SS	Self-Similarity
UE	Under-segmentation Error
VCLFS	View-Consistent Light Field Superpixel
VR	Virtual Reality

Chapter 1

Introduction

This chapter introduces the reader to the main context and motivation for this Thesis. Additionally, it presents the Thesis objectives and briefly explains its contributions. Finally, this chapter outlines the Thesis structure and clarifies the connection between the various chapters.

1.1 Context and motivation

During the COVID-19[1] lockdown, people had to stay isolated at home to avoid transmitting the virus which revealed the challenges of experiencing our life remotely and sometimes virtually. To cope with the challenging lockdown period, various solutions that exploited technology have developed, such as telemedicine platforms, virtual conferences, remote communication at work or school, and virtual tours. However, most of those solutions were Two-Dimensional (2D) in nature, yet many applications in our daily life still require beyond the traditional 2D content in terms of immersion and realistic experience. For example, a surgeon aiming to perform an operation remotely may need a realistic model of the patient's organs, and a therapist who seeks to help a person overcome anxiety or phobia virtually may need to simulate real world scenarios in a safe environment. Therefore, immersive imaging that simulates how we are used to observe the surrounding world in Three-Dimensional (3D) space is demanded. The word "immersive", which originates from the Latin "*immersus*", can be defined, in the context of imaging technologies, as "seeming to surround the audience, player, etc., so that they feel completely involved in something" [1].

In the past decades, with the advances in computational power, capturing/display hardware and available transmission bandwidth, imaging modalities that enhance the sense of presence and enable more realistic experiences for end users have appeared [2]. Moreover, immersive imaging technologies have evolved in research and industrial production to support those imaging modalities. Particularly in the era of extended reality, the production of matured

[1] COVID-19: stands for "Coronavirus Disease 2019", which was first detected in December 2019, in Wuhan, China, and it quickly spread worldwide, resulting in a global pandemic.

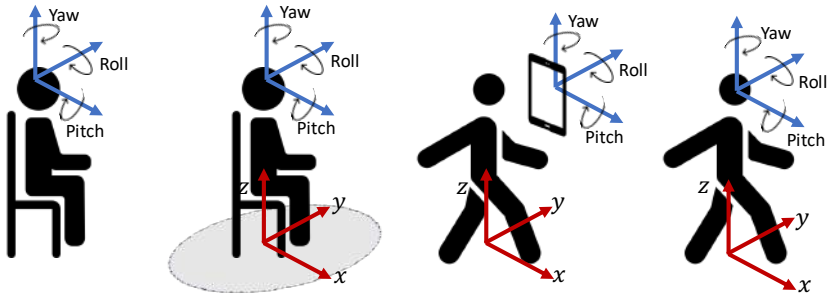


Figure 1.1: Examples of different DoF, where higher DoF offer a more realistic and immersive experience

hardware devices has increased, including Head-Mounted Displays (HMDs), such as Meta Quest 3 from Facebook [3], HoloLens from Microsoft [4], Vive Flow from HTC [5], and powerful displays, such as the recent Sony's spatial reality display [6].

Immersive imaging modalities provide the viewer with higher Degrees of Freedom (DoF) (i.e., the number of movements or orientations that a viewer can experience around the scene within a 3D space) than traditional 2D imaging. It is worth noting that the traditional 2D images provide the viewer with zero DoF since the viewer can only see one view of the scene and cannot move the head or body to change the viewing point. The higher DoF achievable in the different immersive imaging modalities is a result of capturing more information about the scene, such as capturing different viewing points, considering high spatial resolution, or including geometric information to obtain depth perception.

Before reviewing existing immersive imaging modalities and specifying the one this Thesis focuses on, it is helpful to understand and differentiate the levels of DoF available in immersive imaging modalities. The different DoF available in immersive imaging can be divided into four main levels [2], as shown in Figure 1.1 and listed below:

- **3DoF** – In this level, the viewer has three DoF and is limited to a rotation movement around three axes, typically changing yaw, roll, or pitch angles. Therefore, the viewer can change the viewpoint of the virtual environment without being able to freely walk within the virtual space.
- **3DoF+** – In this level, the viewer can change not only the viewpoint by changing yaw, roll, or pitch angles as in 3DoF but can also spatially move his/her head in a limited area along the x, y and z axes.
- **Windowed 6DoF** – In this level, the viewer can observe the scene by watching it

through a window in 6DoF with constrained rotation in pitch and yaw angles and constrained forward movements along the *y* axis.

- **6DoF** – In this level, the viewer can freely rotate around the pitch, yaw and roll axes, as well as freely move along the *x*, *y* and *z* axes. 6DoF means that the viewer can look around and move within the virtual environment in all directions. Basically, this is the one that mimics better how we experience a real 3D scene.

Immersive imaging modalities can be roughly understood as different ways of sampling light in 3D space. The interpretation of light and its nature has evolved over the centuries [7]. Back to the ancient Greeks, when Plato proposed that light consisted of rays emitted by the eyes that enabled the viewer to sense the color, size and shape of the surrounding objects. This theory prevailed for almost 1000 years until Ibn al-Haytham (Latinized as Alhazen and known as the father of optics) conclusively proved it to be wrong [7]. In the 11th century, Alhazen proposed that the light rays originate from the objects and travel to the eyes, and he also explained that the perception of an image occurs in the brain [7]. After hundreds of years, in the 20th century, Adelson and Bergen [8] described the set of light rays traveling in every position in 3D space (x, y, z), through every direction (θ, φ), over any wavelength range λ, and at any time t, and formulated this description by the fundamental "plenoptic function" – the root word of plenoptic came from the Latin *plenus* (full) + *opticus* (vision). The plenoptic function, which is a Seven-Dimensional (7D) function, is presented in (1.1) and illustrated in Figure 1.2:

$$P(x, y, z, \theta, \varphi, \lambda, t) \tag{1.1}$$

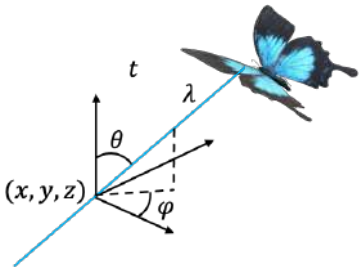


Figure 1.2: Illustration of the 7D plenoptic function that describes the light rays in 3D space

Sampling this plenoptic function may lead to a huge amount of data to be stored and processed. Therefore, in practice, it is not being directly sampled in computer vision and computer graphics applications, but rather a dimensionality reduction is first required.

In this section, **different immersive imaging modalities** will be briefly reviewed and ordered based on the DoF they provide. The following immersive imaging modalities

approximate the plenoptic function and vary in the DoF they provide and their capturing systems [2], [9]. Moreover, the entire or a subset of their content may be displayed using common displaying devices, and they can also be involved in similar practical applications. The immersive imaging modalities that will be reviewed in this section include Light Field (LF) imaging, omnidirectional imaging, holography and volumetric imaging. Among those modalities, this Thesis especially focuses on the light field imaging modality (as will be explained later in this Thesis) for which our contributions are made.

1.1.1 Light Field (LF) imaging

In the case of the LF imaging modality [10], the same scene is captured from different viewing points (as will be detailed later in this Thesis). LFs approximate the plenoptic function in (1.1) and record both the spatial and angular information of light rays in static LFs (a.k.a. still LF images), additionally, temporal information in the case of dynamic LFs (a.k.a. LF videos which can be a sequence of a still scene or a scene with motion). Thus, LF imaging modality provides end users with 3DoF to 6DoF depending on the optical design of the imaging system (i.e., the number of acquisition cameras/lenses and their arrangement). Due to the availability of multiple viewpoints of the scene, motion parallax (i.e., a visual perception effect in which the relative distances of objects to the observer can be determined based on their movement relative to the observer's viewpoint) can be provided when displaying LF content which enhances the immersive experience. Moreover, its rich recorded information can be exploited to advance several image processing and computer vision tasks, such as refocusing the scene and depth estimation [11]. a shows an example of LF refocusing where the flowers scene can be refocused after being captured using a Lytro LF camera [12]. Moreover, in b, the depth map of a scene is obtained from the captured LFs using a Raytrix LF camera [13]. A comprehensive review of LF imaging and its applications can be found in [10], [11], [14].

Figure 1.3: Examples of LF post-capture processing capabilities: a) Refocusing the scene after being captured [12]*; b) Depth map estimation* [13]

1.1.2 Omnidirectional (360°) imaging

In the case of omnidirectional imaging modality (a.k.a. 360°, panoramic and spherical), the surrounding environment of the capturing device (e.g., single or multiple conventional cameras, or 360º cameras) is captured in a 360° (horizontal) × 180° (vertical) Field of View (FoV) (i.e., the observable area of the captured scene) as illustrated in Figure 1.4a. Omnidirectional imaging approximates the plenoptic function in (1.1), where the whole scene is captured from all possible viewpoints over time [15], [16]. Omnidirectional imaging is often defined on the spherical domain, mapped to a 2D image or a multi-planar representation [15]. The variety of 360º cameras in the consumer market as shown in Figure 1.4, such as Samsung Gear 360 [17] and Insta 360 [18] increases the useability of the omnidirectional modality and advances its potential applications. The omnidirectional imaging provides end users with 3DoF (typically changing yaw, roll, or pitch angles) and enables the viewer to look around the scene from a single point of view and feel physically at the scene location. The omnidirectional modality is mainly used for creating realistic content for Virtual Reality (VR) and Augmented Reality (AR) applications to be used in different fields, such as entertainment and virtual tourism [19]. A recent tutorial that addresses omnidirectional foundations, challenges and applications can be found in [15]. Additionally, a comprehensive survey about 360º video streaming can be found in [19].

(a) (b) (c)

Figure 1.4: The omnidirectional modality captures the surrounding scene and enables 3DoF. Examples of 360º cameras: b) Samsung Gear 360 [17]*; c) Insta 360* [18]

1.1.3 Holography

In the case of the holography modality, the 3D hologram can be captured by splitting a coherent light beam (e.g., laser) into two beams as shown in Figure 1.5a [9]. The first one is redirected toward the object(s) and scattered to the recording medium, and the second one is redirected

toward the recording medium. Both beams interfere on the recording medium, resulting in a recorded interference pattern. The same coherent light beam is used to reconstruct the hologram, as shown in Figure 1.5b [9]. The holography word is derived from the Greek "*holo*" word which means "whole", and "*graphy*" which means writing. Thus, "holography" means writing the complete image. The theory of holography was developed by Dennis Gabor in 1947 [20], and he was given the Nobel Prize in Physics for this invention in 1971. Holography can provide the viewer with 6DoF and approximates the plenoptic function in (1.1) by recording not only the light intensity from an object but also the phase of light waves. Some available 3D displays in the consumer market are advertised as being holographic, but are actually not, such as the Hololens 2 [21] and the Leia transparent display [22]. Applications of holograms include but are not limited to, the medical field and education, e.g., to enable students to experience realistic content through 3D holograms. A review of holography in medical fields and future classrooms can be found in [23], [24].

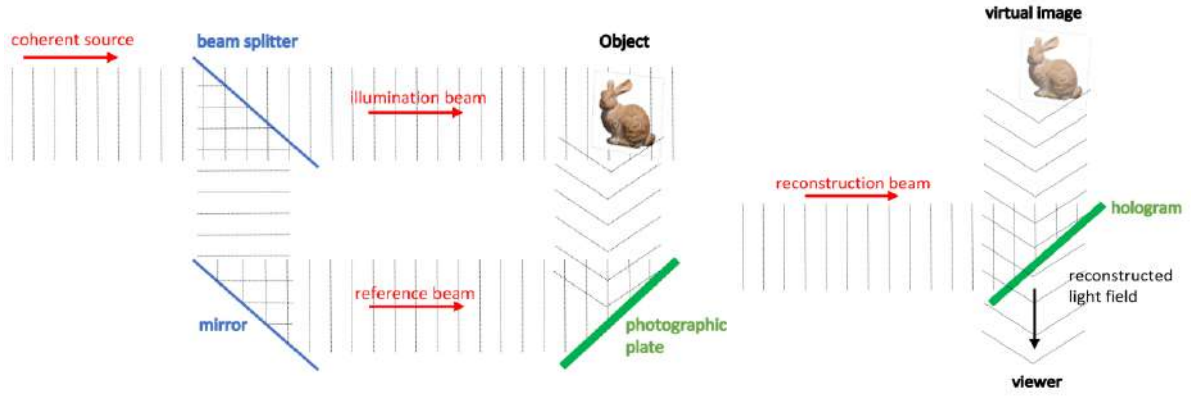


Figure 1.5: Illustration of holography capturing and reconstruction [9]: a) Capturing 3D holograms using coherent light source; b) Reconstructing captured holograms

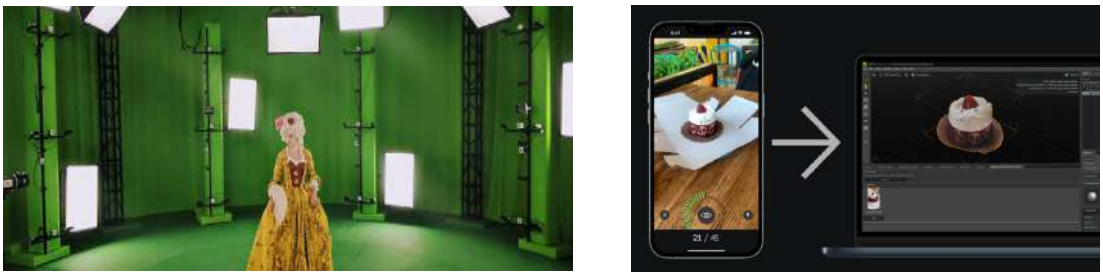
1.1.4 Volumetric imaging

In the case of the volumetric imaging modality, multiple accurately synchronized cameras can be placed around the scene/object to capture it from different angles. In volumetric imaging, object geometry (e.g., depth information) is also recorded using, for example, depth sensors, such as a Light Detection and Ranging (LiDAR) sensor. Volumetric data are typically a 3D set of samples that represent the value of some property at a 3D location (x, y, z). The volumetric data can include binary values (i.e., to represent the background and the object) or multivalued (i.e., to represent some measurable properties, such as color and density). Volumetric imaging

approximates the plenoptic function in (1.1) and provides the user with 6DoF since the captured scene/object can be seen from any position at any viewing angle and at any time (in volumetric videos) [2]. The volumetric content is usually represented by point clouds (i.e., a collection of points in 3D space for which 3D coordinates and additional attributes, e.g., light intensity, are defined). Additionally, the volumetric content can be represented by converting the point clouds into texture meshes (i.e., a collection of polygons or triangles that represent 3D objects).

To capture volumetric content, dedicated studios specifically designed for this task are usually needed, such as the Microsoft capturing studio [25] shown in Figure 1.6. However, with Artificial Intelligence (AI) technology advances, recent applications enable users to generate volumetric data even on smartphones (e.g., after capturing the object from the smartphone's camera from different angles). An example of this application is MagiScan [26], a mobile application powered by AI and supported by Nvidia Omniverse [27] (i.e., a platform for creating and operating metaverse applications), where the synthetic content can be generated on-premises or in the cloud. Additionally, the recent growing trend of using neural networks to synthesize new views from only some reference views captured from different viewpoints enables the rendering of volumetric content. An example of this trend is the Neural Radiance Field (NeRF), proposed by Mildenhall *et al.* [28], [29], [30]. The volumetric imaging modality can be exploited in different applications, such as medical imaging, training simulation, and marketing. In-depth volumetric content surveys can be found in [31], [16].

It is worth mentioning that some modalities can be computed from other modalities [32], [33], [34], e.g., the volumetric data can be computed from LFs with some limitations, e.g., in terms of noise (e.g., due to inaccurate estimation of depth values) and missing data (e.g., due to occlusions) [32], [33]. To achieve that, depth maps can be estimated from LFs, and by using the camera parameters, the attributes of each pixel, such as position and color, can be represented in point clouds.



(a) (b)

Figure 1.6: Examples of capturing volumetric content: a) Microsoft volumetric capturing studio [25]*; b) MagiScan AI-powered mobile application for creating 3D models* [26]

1.2 Light field overview

The apparent similarity between the above-presented immersive imaging modalities is the ability to capture richer scene/object information compared to traditional 2D content. In this context, LF imaging modality captures rich visual and geometric information including light ray intensities and directions; thus, it enables depth estimation and improves the immersive experience in various interesting applications (as will be detailed later in this Thesis). However, due to the high dimensionality of LFs, several challenges arise in terms of processing and editing. One major challenge is to achieve efficient LF processing/editing while maintaining consistency across all LF views. Consequently, this Thesis addresses this challenge and specifically considers achieving efficient and angularly consistent LF processing and editing. Before presenting the objectives of this Thesis and its original contributions, an extended description of LF imaging modality, including LF definition, acquisition, representation, displaying, potential applications, and its challenges and limitations will be presented in the following sub-sections. This should give the reader a good general background knowledge about LFs, which will make it easier to understand the following chapters.

1.2.1 History and description

Back in 1936, the LF concept was first introduced by Gershun [35] to describe the radiance distribution over space and directions. In 1991, Adelson and Bergen [8] further matured the work of Gershun and introduced the 7D plenoptic function, as explained earlier in this Thesis. In 1996, Levoy and Hanrahan [36], formulated a simplified mathematical representation of the 7D plenoptic function to represent LFs using only four dimensions. The following assumptions were considered to achieve the dimension reduction of the plenoptic function in (1.1) from a 7D, as shown in Figure 1.7, into a Four-Dimensional (4D) LF function, namely:

- **Fixed time** – By assuming the scene is static, then the t dimension is constant and the plenoptic function can be reduced into $P(x, y, z, \theta, \varphi, \lambda)$.
- **Free space** – By assuming the air to be transparent and that light rays are transmitted in free space, then the radiance along a ray through empty space remains constant. The plenoptic function can then be represented by its values on an arbitrarily selected surface surrounding the scene, e.g., a cube for its computational simplicity. Doing so, at every point in space outside of the surface, one can trace back a light ray to its surface

to obtain the corresponding ray intensity value [36]. Hence, considering one side of the cube surrounding the scene, the plenoptic function can be reduced into $P(x, y, \theta, \varphi, \lambda)$.

- **Trichromatic human vision system** – The human eye's retina includes three types of specialized cells (called cones) that are responsible for color vision. These three types of cones are sensitive to different wavelengths of light. More precisely, short-wavelength cones, medium-wavelength cones, and long-wavelength cones are more sensitive to blue light, green light and red light, respectively. Therefore, assuming the λ dimension is fixed, for each color component the plenoptic function can be reduced into $P(x, y, \theta, \varphi)$ [36].

These assumptions led to what is called the 4D LF function [36] that can be parametrized in two-plane parametrization (a.k.a. the lumigraph representation [37]), as shown in Figure 1.7b. The two-plane parametrization describes the coordinates of a light ray by knowing its intersection with the two parallel planes to specify its spatial position coordinates and the angle of the light ray in free space. Although any surface in the second assumption can be used, the flat faces of the cube offer a proper way of representing ray coordinates and it matches the geometry of the imaging system in practice [36]. Different symbols are used to describe the 4D LF function in the literature. However, in this Thesis, the following LF representation, shown in (1.2), is adopted:

$$LF(x, y, u, v), \tag{1.2}$$

where (x, y) are the spatial position coordinates and (u, v) coordinates indirectly specify the light propagation angle which will be called here as the angular location coordinates.

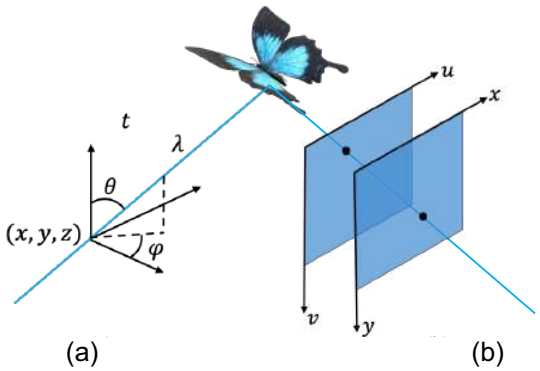


Figure 1.7: Different representations of light rays: a) The 7D plenoptic function; b) The two-plane 4D LF function (a.k.a. lumigraph representation) by assuming static scene, light ray transmission in free space and considering trichromatic human vision system

4D LF imaging captures the same scene from different perspectives. 4D LF can be viewed as a 2D array of 2D arrays as illustrated in Figure 1.8a. Several approaches to capture 4D LFs exist (as will be detailed below). As an example, considering an array of 2D cameras with parallel optical axis capturing the same scene from different viewpoints, different views with spatial shifts are generated (a.k.a. Sub-Aperture Images (SAIs)), as illustrated in Figure 1.8a. When stacking one row or one column of those views, an image with slanted lines which is known as the Epipolar Plane Image (EPI) (i.e., the unique 2D slice of the LF after fixing one spatial dimension and one angular dimension [38]) is created, as illustrated in Figure 1.8b. Notice that the slopes of the slanted lines in the EPIs are inversely proportional to the depth information, e.g., objects near the camera have a larger slope in the EPI and a smaller depth value and vice versa [38].

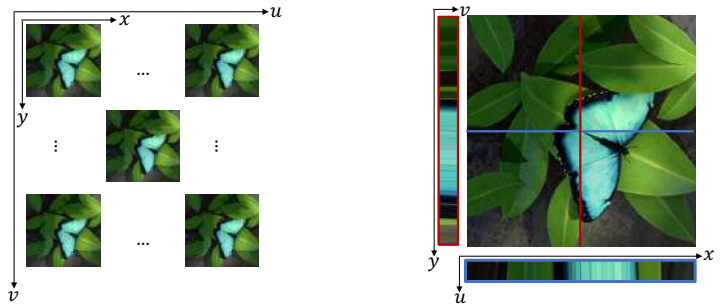


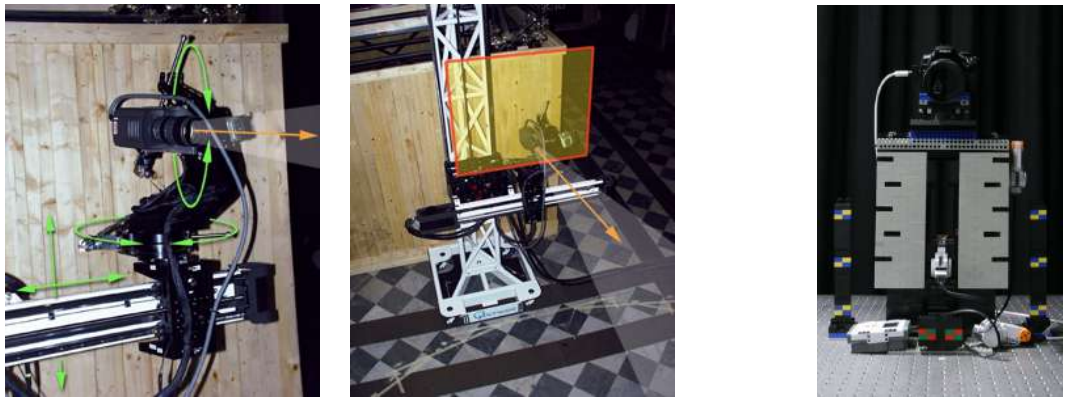
Figure 1.8: Example of 4D LF views and the central horizontal and vertical EPIs: a) Sub-aperture images; b) The central horizontal and vertical EPIs obtained from LF views

1.2.2 Light field acquisition

After explaining the principal concept of 4D LFs and its formulation, it is time to briefly review the different approaches to capture LFs and the advantages and drawbacks of each approach. It is essential to note that in 2D cameras, different light rays emitted from different angles are integrated into a single pixel in the sensor, resulting in a loss of angular information (i.e., the direction of light rays). On the other hand, both the spatial and angular information of light rays are recorded in 4D LF imaging. Several LF capturing systems are proposed in the literature and can be grouped, based on the used cameras, into two main categories: i) Conventional camera systems; and ii) Plenoptic LF camera systems. Notice that these two categories are for LF acquisition using real cameras but not computer-generated 4D LF (i.e., synthetic LF imaging). Both categories are presented below with their advantages and drawbacks.

1.2.2.1 LF capturing using conventional camera systems

In this category, LFs are captured using either a single or multiple traditional 2D cameras. In the first case, a single conventional camera is used to capture different viewpoint images of the same scene at different time instants. For example, the Stanford computer-controlled gantry [39] shown in Figure 1.9a, where the green arrows indicate the translation and rotation along and around the axes, respectively. Another example is the Stanford Lego gantry in Figure 1.9b, which simplified the LF capturing process by using Lego Mindstorm motors [39]. While the single camera approach is less expensive than using multiple cameras, it can only capture static LFs (since the gantry captures the viewpoint images at different instants of time).



(a) (b)

Figure 1.9: Examples of LF capturing using camera gantries: a) The Stanford computer-controlled gantry [39]*; b) The Stanford Lego gantry* [39]

In the second case, an array of cameras can be used to capture both static and dynamic LFs. The arrangements of the cameras can be regular or arbitrary. Designing a camera array can be challenging, and several technical issues may have to be dealt with, such as multiple camera synchronization and color calibrations. Examples of the existing available camera arrays in the literature are shown in Figure 1.10, namely the Stanford [40], Technicolor [41], Saarland University [42] camera arrays and Google capturing rig [43]. Recently, Google combined a mechanical gantry with an array of 16 GoPro Hero 4 cameras to capture LFs for VR applications [44].

Both single and multiple camera approaches are flexible regarding the used camera baseline, which can be adjusted according to the target application. Moreover, the spatial resolution (i.e., the number of pixels visible from each viewing point) and the angular resolution (i.e., the number of viewing points) of the captured LF depend on the sensor

resolution, the number of viewing points and camera parameters. More precisely, in the case of LFs captured with conventional camera systems, the spatial resolution is directly related to the resolution of the used camera(s) and the angular resolution is directly related to the number of cameras or the number of unique viewing points.

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

Figure 1.10: Examples of LF capturing using camera arrays: a) Stanford multi-camera array [40]*; b) Technicolor camera array* [41]*; c) Saarland camera array* [42]*; d) Google camera array* [43]*; e) Google 16 GoPro rotating array of cameras* [44]

Since camera arrays are typically bulky and expensive, they are unsuitable for many commercial uses. However, the potential of the multiple camera approach is evident in the latest smartphones with more affordable and portable camera array designs. Several camera array designs are small, thin, and cheap, providing small camera baselines and can be embedded in other devices and smartphones, as shown in Figure 1.11. For example, the high performance ultra-thin monolithic camera array proposed by Venkataraman *et al.* [45] captures static and dynamic LFs. Another example is the capturing system in recent smartphones, such as the Apple 15 Pro[2], Samsung S23 Ultra[3] and Huawei Mate 50 Pro[4].

(a) (b)

Figure 1.11: Examples of small camera arrays that can be embedded into portable devices: a) Pelican imaging camera array [45]*; b) Smartphones with multiple cameras*

[2] https://www.apple.com/pt/iphone-15-pro/
[3] https://www.samsung.com/pt/smartphones/galaxy-s23-ultra/
[4] https://consumer.huawei.com/en/phones/mate50-pro/

1.2.2.2 LF capturing using plenoptic LF camera systems

In the previous category, LFs are captured either by single or multiple cameras at different time instants. To capture LF simultaneously using one camera, a MicroLens Array (MLA) (a.k.a. lenslet array) can be placed between the main lens of a conventional 2D camera and its sensor. This makes it possible to capture both the scene spatial and angular information, as shown in Figure 1.12. The inspiration for the various approaches in this category started with the prototype proposed by Gabriel Lippmann in 1908 to capture LFs (his approach to capture LFs was called "integral photography") [46]. His prototype mimics the compound eye of insects by using multiple tiny lenses in front of photosensitive material to record the light intensity and direction [46]. However, his prototype remained experimental due to limitations in the quality of optics manufacturing.

After several decades of advances in optics manufacturing, LF capturing has become possible and plenoptic cameras emerged. Adelson and Wang were inspired by the lenses of the Lippmann prototype [47], and they proposed a novel optical system prototype called the "plenoptic camera". In their prototype, a pinhole array or MLA was used and placed between the main lens and the camera sensor [47]. Later, inspired by the Adelson and Wang plenoptic camera prototype, Ng *et al.* [48], improved their design and presented a hand-held camera to capture 4D LFs that uses a MLA. Moreover, they developed software to enable refocusing after scene capturing.

Figure 1.12: Examples of plenoptic LF cameras: a) Plenoptic 1.0 (unfocused) Lytro cameras [49]*; b) Plenoptic 2.0 (focused) Raytrix R11 camera* [50]

Later, LF plenoptic cameras started to be launched in the consumer market. In 2006, Ng started the Lytro company, which developed and sold two different LF plenoptic cameras for the consumer market, i.e., Lytro first generation (the blue one in Figure 1.12a) and Lytro Illum (the black one in Figure 1.12a). The Lytro plenoptic cameras were developed and supported until the Lytro closed its doors in 2018 [49]. Other commercial plenoptic LF cameras available in the market are produced by the Raytrix company, which started in 2010 with several designs and capabilities, such as Raytrix R11 [50], as shown in Figure 1.12b. The Raytrix cameras are more suited for industrial applications. The Lytro and Raytrix cameras correspond to two different types of setups used to capture LFs:

- **Plenoptic 1.0 (a.k.a. unfocused, standard or traditional) camera setup** – In the Plenoptic 1.0 camera setup [49], [51], the distance between the MLA and the sensor equals the focal length of the MLA, as shown in Figure 1.12a. Therefore, the main lens focuses the light onto the MLA, and the MLA directs the light rays into different regions of the sensor. Consequently, the microlenses are focused at infinity. Hence, each pixel of a micro-image (i.e., an image in the sensor created by a microlens) corresponds to a different viewing point.
- **Plenoptic 2.0 (a.k.a. focused) camera setup** – In the Plenoptic 2.0 camera setup [52], [53], the main lens focuses the light onto the image plane where the MLA is focused, and the MLA focuses the light rays on the sensor. In this camera setup, the focus of the main lens can be on a plane in front of the MLA, such as in Figure 1.12b or behind it. This corresponds to Keplerian and Galilean optical configurations which lead to real and inverted main lens images, respectively [54]. It is also worth mentioning that the MLA in Plenoptic 2.0 can include microlenses with different focal lengths (a.k.a. multi-focus plenoptic camera), which can provide deeper depth-of-field (i.e., the range of distance within a scene where the objects in an image appear in acceptably sharp focus) [55], [56].

Notice that according to the distance between two or more cameras/lenses/microlenses (a.k.a. baseline), dense or sparse 4D LFs are generated. In the case of dense 4D LFs, the camera baseline is relatively narrow; hence, most of the objects in the captured LFs mainly exist in all views. In contrast, in the case of sparse 4D LFs, the camera baseline is relatively wide; hence, the objects in the captured LFs may not exist in all LF views. In both designs of plenoptic LF cameras, the baseline between microlenses is very small, encouraging the development of exciting techniques, such as super-resolution (including spatial, angular or temporal

dimensions). It is worth mentioning that plenoptic cameras still have some limitations, such as color inconsistency across corresponding pixels and vignetting effect (i.e., reduction of the brightness of an image towards the border views compared to the center views).

Moreover, using one sensor to record spatio-angular information simultaneously leads to a spatio-angular resolution trade-off in the captured LFs. In the case of plenoptic LF camera systems, since only one sensor records light ray intensities that come from various microlenses, specifying the spatial and angular resolutions can be more complex and may depend on other parameters, such as microlens focal length.

While plenoptic LF cameras are specifically designed for LF capturing, a prototype from a German startup company was created to enable a standard camera to shoot LFs and capture depth information. The proposal was called K|Lense One [57], and it was considered the first LF lens. It included a mirror system inside the lens that worked as a light splitter to split the light rays into nine slightly different perspectives of the scene. Indeed, the idea would open new doors for LF applications; unfortunately, the K|Lens One project was canceled in early 2022 [58].

1.2.2.3 Computer-generated LFs

Besides the two main categories of capturing real world LFs, LFs can also be computer-generated (i.e., synthetic creations) by simulating realistic environments.

The first example of synthetic LF generation is by using simulated camera arrays in 3D software applications, such as Blender [59]. Additionally, several Blender add-ons are also available for LFs to facilitate creating camera array setups and generate disparity/depth maps and segmentation label images for all LF views, such as the LF add-on from HCI [60] (see Figure 1.13). The created synthetic datasets try to mimic real world scenes, and the applications used to create the synthetic datasets usually provide complete control over various parameters, such as lighting, camera baseline, focal length and camera resolution. Moreover, synthetic LF generation enables obtaining the Ground Truth (GT) depth maps or label images for segmentation. Synthetic LF datasets offer significant benefits in various applications, especially in deep learning based applications. Deep learning based applications may require a substantial number of labeled images or depth maps to train models. Interested readers can find a table with different LF datasets summarized by Sheng *et al.* [61], including real world and synthetic LFs for various LF applications.

The second example uses deep learning techniques and the recently proposed NeRF technology [28] for synthesizing new LF views from only a few reference real world or synthetic LF views [10]. The techniques in this approach rely on learning the features and patterns from a given dataset that has views captured from different angles. Afterwards, the trained model can infer the appearance and spatio-angular information of novel viewpoints. With the advances in NeRF technology, we can anticipate portable devices for capturing LFs to emerge. Currently, mobile phones and drones are already being used to capture LFs, without the need for complex camera rigs, the NeRF technology is exploited to reconstruct 3D objects from the captured LFs [62]–[64].

(a) (b)

Figure 1.13: Examples of synthetic LF generation: a) A screenshot of a 3D application by which LFs are generated [60]*; b) One view of the generated LF by synthetic cameras and its ground truth depth map* [60]

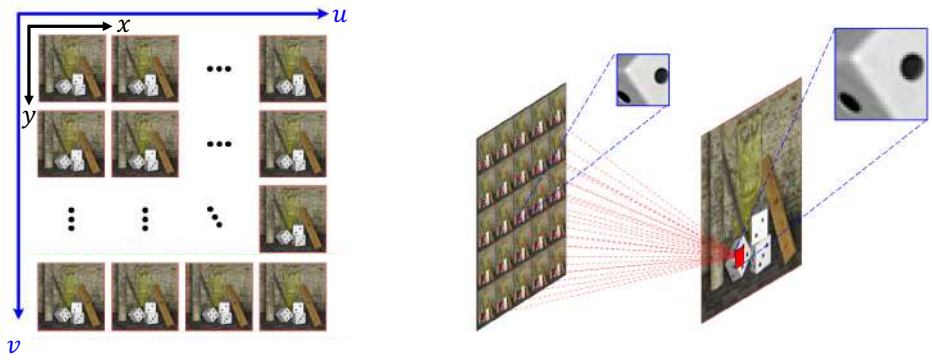
In summary, different approaches for LF acquisition are reviewed namely, by using conventional camera systems, plenoptic camera systems and by synthesizing LFs using synthetic camera systems. According to the acquisition approach, LFs can be represented in different raw representations which vary in their properties and applications as will be described in the following sub-section.

1.2.3 Light field representations

After acquiring LFs using any imaging system mentioned above, the acquired data can be accessed through one of the following raw representation formats, as illustrated in Figure 1.14:

- **Sub-aperture image representation (a.k.a. Multiview representation)** – When an LF is acquired by an array of cameras or camera gantry, the raw representation format is called SAI representation. In the SAI representation, the 4D LF is organized as a 2D array of 2D views where each view represents one unique viewing perspective of the scene, as shown in Figure 1.14a.

- **Lenslet representation** – When an LF is acquired using plenoptic lenslet cameras, the LF content is represented by an array of micro-images (a.k.a. microlens images or macro-pixel images). A micro-image comprises a low-resolution portion of the scene. Depending on the used plenoptic camera setup and the microlens properties, such as the size, shape, and spacing of the microlenses, the micro-image characteristics differ accordingly. Each micro-image includes information about the intensity and direction of light rays arriving at the camera from various angles. As can be noticed in Figure 1.14b, the lenslet representation contains a local mosaic pattern introduced by micro-images. The lenslet representation can be adopted to extract the spatio-angular information in some deep-learning based LF processing applications, as proposed in [65]. Generally, to better visualize LF content captured by plenoptic cameras, from different viewpoints, SAIs can be extracted from the lenslet representation. The process to do this, depends on the used plenoptic camera setup.



(a) (b)

Figure 1.14: Examples of 4D LF raw representation formats: a) SAI representation; b) Lenslet representation

To sum up, 4D LFs typically have two raw representations depending on the acquisition approach. Converting from one representation to another is possible (due to 4D indexing), as shown in Figure 1.14. In some cases, this conversion can be reversible or not, depending on the camera setup and the used algorithm for conversion. For example, in the focused and multi-focus plenoptic cameras, extracting SAIs from the raw lenslet representation is more challenging since it depends on the accuracy of the estimated disparity map and the patch size (i.e., number of pixels extracted from each micro-image) instead of a single pixel in the case of unfocused plenoptic cameras [66]. Depending on the target LF application, other intermediate and more adequate representations can be adopted, such as extracting the EPIs from LF views, organizing LF views in any scanning topology (e.g., raster or zig-zag) to form Pseudo Video Sequence (PVS), or applying LF over-segmentation to obtain a more compact

representation by grouping similar pixels across LF views into larger segments, as will be detailed later in Chapter 3 and Chapter 4.

1.2.4 Light field displays

As LF content inherently includes different viewpoints of the same scene, it is possible to display LF content across a diverse range of display technologies by generating display-compatible versions from the same LF content. LF content can be displayed using numerous available technologies, such as:

- **2D conventional displays** – When displaying LF content on 2D displays, only one LF view is displayed at a moment with the ability to change focus planes and viewing perspectives.
- **Stereoscopic displays** – When displaying LF content using stereoscopic displays, a pair of views is displayed (one for each eye), while wearable glasses are required to create a sense of depth perception. Notice that there are also two-view autostereoscopic displays that allow the viewing of 3D content with naked eyes.
- **Multiview autostereoscopic displays** – In this case, multiple LF views are displayed simultaneously without the viewer needing to wear any special glasses. Each eye will see a different view depending on its current position. As such, the multiview autostereoscopic displays provide the user with horizontal motion parallax (i.e., the user can move horizontally and see different stereo viewpoints of the scene). However, the number of views is relatively low. Some advanced displays in this category incorporate a head/eye tracking system to adjust the displayed views according to the viewer's position.

While the mentioned approaches make it possible to display LF content, displays that provide a more natural viewing experience and more accurate depth perception, simultaneously for one or multiple users, are required. Recently, several displays have become available in the consumer market with compelling capabilities and they deliver what has been expected in terms of depth perception, parallax, engagement, and immersion [67], [68]. Among the available display technologies that can be used for LF content, LF displays and HMDs will be described below in more detail with examples for each approach:

- **LF displays (a.k.a. super-multiview displays)** – LF displays not only provide the depth perception of the scene but also the natural full motion parallax. LF displays

enhance the immersive experience with naked eyes. The integral imaging approach to display LFs uses the reverse path of LFs captured by plenoptic LF cameras with MLAs since it also contains an MLA in its design [69], [70]. An example of this approach is the prototype of the FOVI3D display [71], [72], shown in Figure 1.15a. Additionally, the Spatial Reality display from Sony (shown in Figure 1.15b) contains a micro optical lens that is positioned precisely over the liquid crystal display and can separate the images that are seen by both eyes, allowing for stereoscopic viewing without the need for glasses. The Spatial Reality display responds to head movement with 6DoF in head motion [6], and it received the Best of Innovation award at the Consumer Electronics Show event in 2021. Other existing displays that provide full motion parallax without necessarily using an MLA in their design can be considered as LF displays. These displays often exploit a combination of optical devices, such as lenticular lenses and parallax barriers to allow different images to be seen from different viewing angles; thus achieving the 3D perception [73]. The exact technology used can vary depending on the specific implementation and manufacturer. For example, Looking Glass displays, presented in Figure 1.15c, have been a pioneer in LF displays for years and have different designs that allow one or a group of people to view 3D simultaneously [73].

(a) (b) (c)

Figure 1.15: Examples of LF displays: a) FOVI3D LF display [71], [72]*; b) Sony Spatial Reality display [6]; c) Looking Glass displays* [73]

- **Head-mounted displays** – With the growing interest in extended reality applications, various HMDs are available with diverse designs [3]–[5]. HMDs allow personal viewing of 3D content by visualizing the 3D content in front of the user's eyes. HMDs typically integrate small screens or lenses that display separate images to each eye, resulting in stereoscopic vision and providing a sense of depth. Moreover, most HMDs include sensors that track the user's head movement (e.g., rotation). Hence, they enable AR/VR applications to respond to the user's action in real time and provide a full parallax experience. Figure 1.16 shows examples of HMDs including: i) HoloLens from Microsoft [4]; ii) Vive flow from HTC [5]; iii) Meta Quest 3 from Facebook [3];

and iv) Vision Pro from Apple [74]. Moreover, unlike the previous HMD examples, Figure 1.16 shows thin, lightweight LF HMDs examples, such as the Creal LF HMD from CREAL that enables genuine focus at any distance of the scene [75] and Magic Leap 2 with built-in eye tracker [76]. Moreover, other prototypes were proposed that exploit the MLA principle, such as the Near-Eye LF display from NVIDIA [77], and Lenslet VR display [78].

As displaying devices keep advancing, we can expect a new era of displays in which LF content seamlessly integrates with the real world, creating captivating immersive experiences like never before.

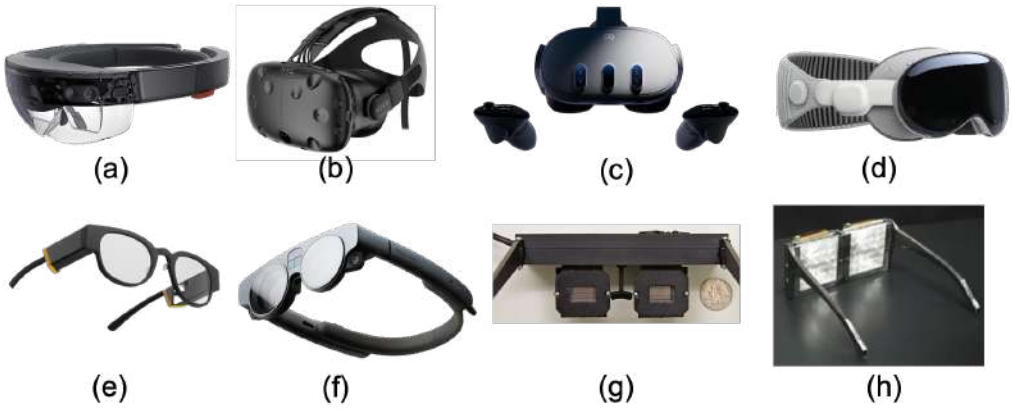


Figure 1.16: Examples of various designs for HMDs: a) HoloLens from Microsoft [4]; b) Vive flow from HTC [5]; c) Meta Quest 3 from Facebook [3]; d) Vision Pro from Apple [74]*; e) Creal LF HMD from CREAL* [75]*; f) Magic Leap 2 from Magic Leap* [76]*; g) Near-Eye LF display from NVIDIA* [77]*; h) Lenslet VR display* [78]

1.2.5 Light field applications

Thanks to the rich information that is included in LFs, LF imaging enables a range of powerful applications and, thus, has the potential to revolutionize various areas. LF usage can be beneficial for many practical applications in various areas, such as:

- **In computer vision** – The usage of LFs has shown a superior performance when compared to traditional 2D images in several computer vision tasks including:
  - *Depth/disparity estimation* – Thanks to the multiple viewpoints and the EPI structure of LFs, such rich information enables estimating per-pixel depth/disparity maps from LFs accurately. This can be achieved by exploiting the angular correlation and detecting occlusions across LF views. In the literature, several methods were proposed for LF depth/disparity estimation as in [10], [11], [14], [79].

- *Object segmentation* – 2D image segmentation is a challenging task, notably when color/texture information is not enough to separate overlapped objects that share the same color/texture but are located at different depth planes. Using LFs can help overcome this challenge and improve segmentation accuracy by exploiting the spatio-angular information and robustly detecting the occlusions across LF views [10], [11], [14]. More details about LF segmentation and available methods in the literature are presented in Chapter 5 and Chapter 6.
- *Scene understanding* – LFs can enhance scene understanding by providing the spatio-angular information of light rays. As an example, in smart cities where autonomous vehicles are operated, the ability to detect and recognize surrounding objects is crucial for overall safety and reliability. Using the rich information of LFs has shown improved performance compared to traditional 2D images in detecting cars, pedestrians, traffic signs and other objects in urban scenes [61]. Although available LF datasets for training deep learning models are relatively scarce, when compared to the 2D urban scenes datasets, the potential of using LFs to improve urban scenes understanding and autonomous driving is a growing work direction [61], [80]–[82].
- *Resolution enhancement* – Depending on the used LF acquisition device, a trade-off between spatial, angular or temporal (in the case of dynamic LFs) resolution may arise [14], [83]. To overcome this drawback, different methods in the literature proposed ways to enhance the recorded LF resolution by leveraging the additional information available in other LF views. Examples of such applications are LF angular super-resolution, a.k.a. LF view synthesis (i.e., the process of creating novel views from a given small set of reference views) and LF spatial/temporal super-resolution (i.e., the process of enhancing image details, sharpness, clarity and frame rate in the case of dynamic LFs) [83]–[85].

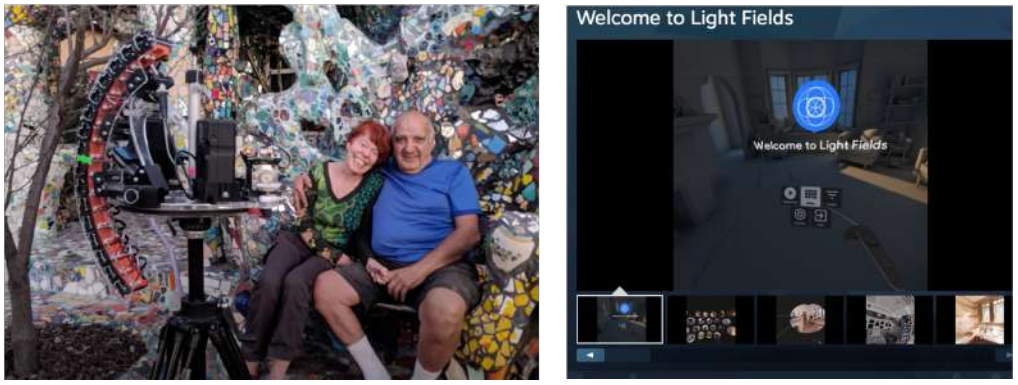
- **In movie production** – After capturing the same scene from different viewpoints, LFs enable post-capture refocusing (i.e., the ability to adjust the focus plane of the scene after it has been captured) [86], [42]. Such an advantage benefits movie production and allows for greater flexibility in post-production. Moreover, as described before, LFs enable estimating accurate depth information. Therefore, the objects in a scene can be segmented and extracted with the assistance of their depth information instead of using chroma-keying techniques (i.e., based on green screens in the background in the case of indoor scenarios). After extracting objects from the scene, one can edit the color,

change the background/lighting or perform other editing tasks. By exploiting LFs in movie production, more immersive content can be produced, and a sense of presence can be achieved which improves the user experience while watching a movie. Therefore, LF content and emerging displays together have the potential to revolutionize the cinema experience where viewers sitting in different locations could see a slightly different viewpoint similar to how we see objects in real life. An example of this application is movie production using the Lytro Cinema camera prototype [87] as shown in Figure 1.17. Although Lytro company closed its doors due to commercial issues in 2018, alternative devices that benefit from the power of LFs in movie production still deserve further investigation.

(a) (b)

Figure 1.17: Example of LF usage in movie production: a) Lytro Cinema camera prototype [87]*; b) Example of post-capture refocusing from Lytro Cinema* [87]

- **In video gaming and storytelling** – Extended reality applications have gained increasing attention in recent years and this increased the pressure on the production of mature HMDs. Immersive video gaming and storytelling are two entertainment applications, where HMDs offer a compelling and engaging experience and can fully or partially disconnect users from the physical world. In this context, capturing LFs of real world environments can give the end users a strong sense of realism. Thanks to the multiple viewpoints of LFs and the EPI structure, this improves rendering performance even for non-Lambertian (e.g., reflecting) regions [88], which is necessary for such applications. The realistic LF content can be especially beneficial in museum tours and storytelling. Recently, a prototype from Google [44] has been proposed to produce VR content that includes LFs captured from real world environments. The Google application “Welcome to light fields” is publicly available and enables end users to virtually explore numerous places [44], as presented in Figure 1.18.



(a) (b)

Figure 1.18: Example of LF usage in VR applications: a) Google's LF camera rig [44]*; and b) The "welcome to light fields" application for virtual reality content* [44]

- **In medicine and biology** – Compared to traditional microscopy, LF microscopy captures both the ray intensity and direction, thus enabling single-shot volumetric recording. LF microscopy enables depth estimation of the specimen and offers better 3D reconstruction. For example, Levoy *et al.* [89] proposed a prototype of LF microscopy by adding a microlens array to the traditional microscopy and achieved useful perspective views and focal stacks from the captured spatio-angular information. Moreover, Longo *et al.* [90] proposed the first demonstration of a flexible plenoptic microscope operating with hard X-rays. A recent review of LF microscopy can be found in [91], [92]. Although LF microscopy has unique advantages over the traditional one, it still has challenges, e.g., the trade-off between spatial and angular resolution. However, rapid technological advances, including hardware devices and software algorithms, can typically overcome many of these challenges. Another application in medicine could be to use LF content in medical applications e.g., using extended reality to help doctors and medical students in their simulations and diagnostics [93].
- **In security** – LF imaging can help in advancing the accuracy of security measures. Given the multiple perspectives and the ability to estimate the depth information from the recorded content, one can see through occlusions, which is crucial in some security scenarios. Moreover, LF content can be exploited in several biometric identifications, such as face scanning and recognition assisted by depth information, as shown in Figure 1.19 [13]. In [94], the results have demonstrated the importance of using additional information rendered by LF cameras to enhance the whole performance of biometric systems, including face and iris recognition. Additionally, object detection and tracking tasks can be improved by exploiting the depth information computed from LFs [95],

[96]. Detecting and tracking suspicious objects or individuals in high-security areas is essential in security applications. Furthermore, LFs enable post-capture refocusing, vital in surveillance scenarios where the region of interest can be refocused even after an image has been captured or the initial focus point was on another plane [97].

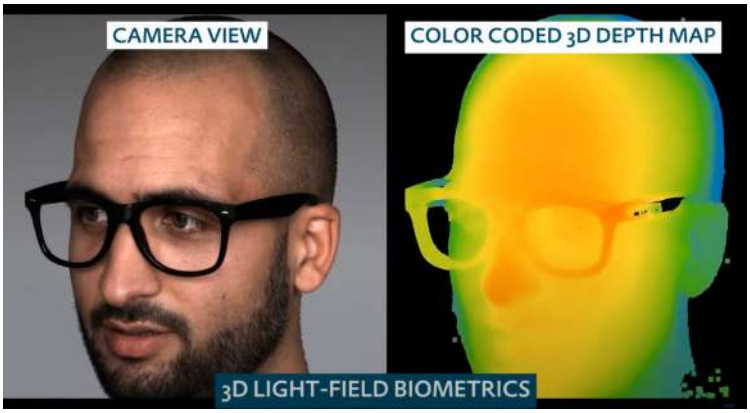


Figure 1.19: Example of Raytrix LF camera usage in face recognition using 3D LF biometrics [13]

- **In communications and metaverse** – LF content can enrich remote communications and the metaverse by exploiting the depth cues and its 3D representation. LF content and displays can also revolutionize remote conferencing in communication platforms by making the experience more engaging. Displaying captured LFs in personal communication scenarios enables the end users to better realize gestures and facial expressions, thus making remote communications more effective. Recently, Google revealed Project "Starline" and produced a glasses-free LF display, which can be considered as a "magic window" that makes the end users feel as if they are physically together and enriches real time remote video conferencing, as shown in Figure 1.20 [98]. As the metaverse has attracted increasing attention recently [99], LF content and its advantages to the metaverse are also being investigated [100]. Since LFs can provide higher DoF, the sense of presence within the metaverse can be improved making virtual environments feel more realistic [100]. Moreover, LF content can be used to generate more realistic avatars that better reflect the individual's appearance.

Finally, it is worth noting that LF applications are not limited to the presented examples, and LF technology can be beneficial for other areas, such as for optical inspection inside factories and research plants, as illustrated in [50]. The promise of immersive experience enabled by LF imaging modality comes with challenges and limitations that need to be further investigated.

Figure 1.20: The glasses-free LF display from Google's Starline project [98]

1.2.6 Examples of light field challenges and limitations

LF imaging is promising and can enhance how we are used to capture, visualize, and interact with visual content. However, LF imaging still presents numerous challenges and limitations that need further research and improvement. Examples of such challenges include:

- **Resolution trade-off** – LF capturing devices typically provide high resolution in one dimension at the expense of other dimensions (e.g., high angular resolution with low spatial resolution as in Plenoptic 1.0 cameras [49]). This limitation is due to the limited sensor sizes available. Therefore, efficient super-resolution methods are required to enhance the spatial/angular resolution depending on the target application.
- **Massive amounts of data** – The massive amounts of involved data need to be efficiently coded/compressed to facilitate LF storage and transmission according to each user's requirements. The interested reader is encouraged to read these recent reviews and references for efficient solutions for LF coding [101], [102], [103].
- **Processing and editing computational complexity** – Processing and editing LF content often require significant computational resources in terms of memory and execution time compared to 2D images. Therefore, efficient processing and editing methods that exploit correlations across LF views and reduce processing and editing complexity are needed.
- **Ensuring angular consistency (a.k.a. view-consistency) in processing and editing** – In 2D images, processing and editing accuracy is a major performance metric, for 4D

LFs preserving angular consistency is also essential. Ensuring processing and editing LF angular consistency means maintaining accuracy and coherence across LF views. More precisely, corresponding pixels across LF views must have the same features since they represent the same point in 3D space and changing the viewpoint should occur smoothly and naturally. Inconsistent LF processing and editing may lead to incorrect results with artifacts, which can have a negative subjective impact, especially when changing the viewpoints. Hence, it affects the end user's perceptual quality of the immersive experience. Therefore, proposing LF processing and editing methods that ensure consistency across LF views is crucial.

Notice that the first two examples are out of this Thesis scope, and this Thesis contributes to achieving efficient and angularly consistent 4D LF processing and editing. A detailed description of the specific challenges addressed in this Thesis and its major objectives will be presented in the following section.

1.3 Thesis objectives and original contributions

This Thesis seeks to address some of the abovementioned challenges and advance the state-of-the-art in terms of 4D LF processing and editing. More precisely, **this Thesis aims at achieving efficient 4D LF processing and editing by exploiting the spatio-angular similarities across all LF views, while ensuring angular consistency**. In this context, four major research objectives were defined to address specific tasks in LF processing and editing, namely:

1) **Proposal of an efficient disparity propagation method that enables computing angularly consistent disparity maps for all LF views –** One of the powerful advantages of LF imaging is the ability to estimate depth/disparity information from the rich recorded data. Existing LF disparity estimation methods in the literature can be classified according to the used approach as either classical methods, (e.g., relying on analyzing the EPI structure, matching corresponding pixels across views, or using defocus cues) or learning based methods (e.g., dependent on extracting deep features from LFs using various neural network layers [10], [11], [14], [79]). More details about LF depth/disparity estimation will be presented in Chapter 2.

   Depth/disparity estimation from LFs is an evolving work direction. However, most existing disparity/depth estimation methods either require dense LFs, estimate a disparity map only for the central view, or estimate a disparity map for any view in any angular location without adequately considering LF angular consistency constraints and

large occlusions [79], [104], [105]. Therefore, the first objective of this Thesis is to propose an efficient disparity propagation method that enables computing angularly consistent disparity maps for all LF views.

To achieve that, this Thesis proposes a method that exploits off-the-shelf deep learning based disparity estimation methods to estimate a disparity map for the central view. Afterwards, the method propagates the central view disparity values into all LF views progressively in an occlusion-aware manner to preserve the disparity maps angular consistency. This way, the computed per-pixel disparity can be used to guide LF processing/editing accurately and ensure angular consistency. In this Thesis, all the upcoming objectives rely on the availability of disparity maps for all LF views as a helpful feature for robust processing and editing. This Thesis objective has resulted in the following publication:

- M. Hamad, C. Conti, P. Nunes and L. D. Soares, "Efficient Propagation Method for Angularly Consistent 4D Light Field Disparity Maps," *IEEE Access*, vol. 11, pp. 63463-63474, Jun. 2023, doi: 10.1109/ACCESS.2023.3287920.

2) **Proposal of accurate and angularly consistent LF over-segmentation method for dense and sparse LFs –** In 2D images, one efficient approach before applying image processing or editing is to first apply image over-segmentation (i.e., segment an image into locally coherent regions that adhere more accurately to object boundaries by grouping similar pixels that share similar criteria) [106]–[108]. The rationale for this approach is to use the obtained image over-segmentation as an intermediate data representation to reduce the amount of data to be processed, thus simplifying subsequent tasks [107]. Similarly, this approach is adopted in this Thesis by applying the LF over-segmentation step for 4D LFs and using it as an intermediate representation for other processing and editing tasks. Applying 4D LF over-segmentation not only reduces the amount of data to be processed but also ensures angular consistency.

Existing 4D LF over-segmentation methods mostly rely on the *K*-means clustering algorithm to achieve the over-segmentation. Some limitations still exist in the existing 4D LF over-segmentation methods, such as using fixed clustering weights that may lead to non-optimal over-segmentation, assuming dense LFs without adequately considering sparse LFs, and not fully exploiting the spatio-angular information. Therefore, this Thesis proposes two different clustering based methods that address different limitations in the existing 4D LF over-segmentation methods. Both proposed methods are automatic, accurately adhere to object boundaries and ensure angular

consistency for all corresponding pixels across all LF views. Moreover, both methods exploit per-pixel disparity information during the over-segmentation as a valuable feature to achieve accurate and angularly consistent segmentation. Yet, those methods differ in their objective and the limitations they address.

The first method considers adaptively adjusting the clustering weights of the various features according to the LF content to achieve robust, compact (i.e., regular in size) and angular consistent 4D LF over-segmentation, as will be detailed in Chapter 3. This method assumes dense 4D LFs and is not suitable for sparse 4D LFs.

The second method proposes a flexible 4D LF over-segmentation for both dense and sparse 4D LFs. Different from the existing clustering based 4D LF over-segmentation methods that initialize clustering centroids only in the central view, this proposed method initializes the cluster centroids in different LF views (not only in the central view) to be able to consider all objects, not only those appearing in the central view. Moreover, existing methods typically apply *K*-means clustering in the central view and propagate the over-segmentation into all other LF views. However, this proposed method applies the *K*-means clustering in 4D space. To validate the proposed method on sparse 4D LFs, a new 4D LF dataset is also generated, the proposed dataset can also be used for different LF applications (e.g., in LF segmentation or depth/disparity estimation). Moreover, besides the frequently used metrics to evaluate 4D LF over-segmentation, this Thesis proposes a modified metric to evaluate over-segmentation angular consistency for both dense and sparse LFs. The work developed to reach this objective has led to the following journal publications:

- M. Hamad, C. Conti, P. Nunes and L. D. Soares, "ALFO: Adaptive Light Field Over-Segmentation," *IEEE Access*, vol. 9, pp. 131147-131165, Sept. 2021, doi: 10.1109/ACCESS.2021.3114324.
- M. Hamad, C. Conti, P. Nunes and L. D. Soares, "Hyperpixels: Flexible 4D Over-Segmentation for Dense and Sparse Light Fields," *IEEE Trans. Image Process.*, vol. 32, pp. 3790-3805, Jul. 2023, doi: 10.1109/TIP.2023.3290523.
  This paper was presented as well at the *IEEE Inter. Conf. Acoustics, Speech, Signal Process. (ICASSP)*, April 2024, Seoul, Korea, doi: 10.60864/fzyz-h513.

3) **Proposal of efficient and angularly consistent methods for LF segmentation –** While 2D image segmentation is challenging, 4D LF segmentation is even more demanding due to the additional angular consistency requirement it should consider.

Existing 4D LF segmentation methods can be categorized according to the level of the semantic meaning of the obtained segments into low-level segmentation (a.k.a. over-segmentation), mid-level segmentation (i.e., object level without semantic labels), and high-level segmentation (i.e., object level with semantic labels, such as a car, a person, etc.), as will be detailed in Chapter 5 and Chapter 6. Most existing 4D LF (mid/high-level) segmentation methods either: i) Rely on user scribbles or supervision; ii) Do not support sparse LFs; iii) Only apply segmentation to the central view; or iv) Do not adequately exploit LF view correlation or ensure angular consistency.

Therefore, this Thesis proposes two different methods for achieving 4D LF mid-level segmentation. The first method is semi-supervised (assisted by user selection) and aims at interactively segmenting the foreground from background objects. This proposed method relies on graph technique (i.e., graph cut optimization) as most existing 4D LF semi-supervised segmentation. Different from the existing 4D LF semi-supervised segmentation methods, this method greatly reduces the graph size and ensures segmentation angular consistency without the need for an explicit depth/disparity map estimation step. To achieve that, this method requires applying LF over-segmentation first. Then, it represents a 4D LF as a hypergraph based on LF over-segmentation and applies classical graph cut optimization to achieve foreground-background segmentation. This method, as the other existing semi-supervised methods, supports only dense LFs.

The second method is unsupervised (automatic) and aims at segmenting a 4D LF into multiple objects without depending on the user's scribbles or GT label images. The proposed method uses deep learning techniques, i.e., a Graph Neural Network (GNN), ensures angular consistency and does not require any GT labels to train the model. Moreover, it supports both dense and sparse LFs. To evaluate the segmentation angular consistency for both dense and sparse LFs, this Thesis proposes a set of complementary metrics for evaluating LF segmentation angular consistency. Up until now, it is the first 4D LF mid-level segmentation method that uses a GNN in an unsupervised manner and supports both dense and sparse LFs. This Thesis objective has resulted in the following publications:

- M. Hamad, C. Conti, A. M. de Almeida, P. Nunes and L. D. Soares, "SLFS: Semi-supervised Light-field Foreground-background Segmentation," *2021 Telecoms Conf. (ConfTELE)*, Leiria, Portugal, 2021, pp. 1-6, doi: 10.1109/ConfTELE50222.2021.9435461.

- M. Hamad, C. Conti, P. Nunes and L. D. Soares, "Unsupervised Angularly Consistent 4D Light Field Segmentation using Hyperpixels and a Graph Neural Network", *IEEE Open. J. signal process.,* accepted for publication, doi: 10.1109/OJSP.2025.3545356.

4) **Proposal of an efficient method for angularly consistent LF editing –** 4D LF editing is generally more challenging than traditional 2D image editing. The reason is that any edit applied in one view must be consistent and accurate in all other LF views. In this Thesis, "Neural Style Transfer" was selected as the target LF editing application. Neural style transfer consists of using neural networks to generate a new image from two reference images, namely one for the content of the new image and another one for the style and colors (usually an art image). Neural style transfer is a trendy editing application in computer vision. However, applying existing 2D methods for each LF view independently can lead to unnatural artifacts and inconsistent results.

   Therefore, only a few methods addressed neural style transfer for 4D LF to ensure editing angular consistency. However, they either: i) Rely on dense LFs; ii) Need to train a model for each different style; or iii) Require optimizing angular consistency for each LF which can be very time-consuming. This Thesis proposes an angularly consistent 4D LF neural style transfer method that is flexible to any style image and supports different disparity ranges. This method highlights the advantage of exploiting LF over-segmentation since it only applies neural style transfer for reference views and then relies on the obtained 4D LF over-segmentation to apply edit propagation to all LF views robustly and consistently. This Thesis objective has resulted in the following international conference publication:

   - M. Hamad, C. Conti, P. Nunes and L. D. Soares, "View-consistent 4D Light Field Style Transfer using Neural Networks and Over-segmentation," *IEEE 14th Imag., Video, Multidimensional Signal Process. (IVMSP)*, Nafplio, Greece, 2022, pp. 1-5, doi: 10.1109/IVMSP54334.2022.9816312.

1.4 Thesis structure

This Thesis adopts an article-based structure (a.k.a. composite Thesis) in which a collection of research articles is concatenated and accompanied by an extended introduction and conclusion that tie them together. More precisely, after this introductory chapter, the proposed publications

are concatenated and organized in a sequence of chapters, as will be explained in this section. Finally, this Thesis concludes with some final remarks about the Thesis achievements and some suggested research directions for future work.

Since this Thesis is article-based, outlining its structure helps the reader to understand the connection between the various chapters and the rationale for the Thesis work. Therefore, the structure of this Thesis is presented in Figure 1.21, where the arrows between two blocks indicate that the result of the source block is used/required by the target block. As presented in Figure 1.21, this Thesis includes:

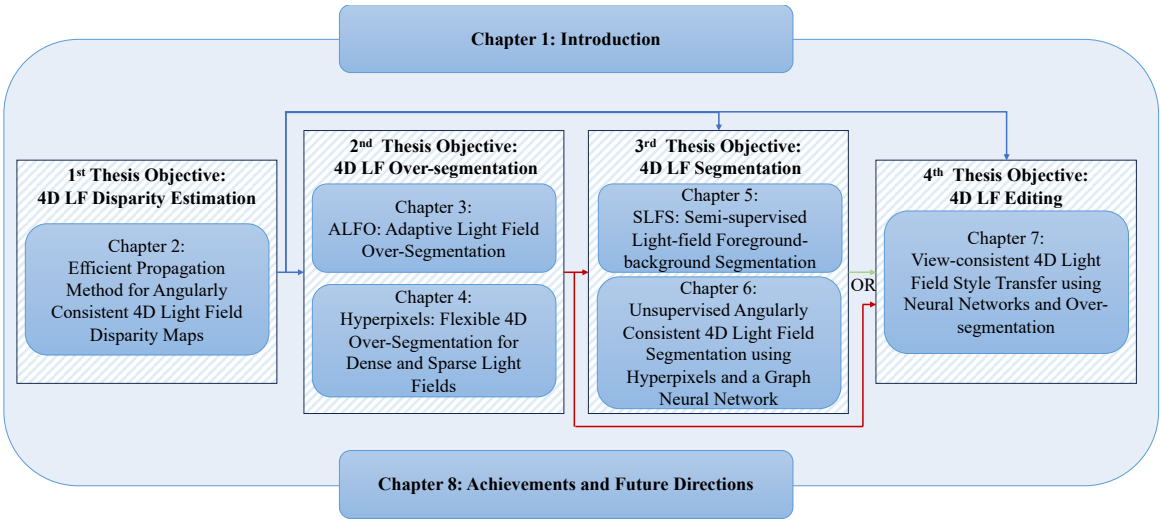


Figure 1.21: Thesis structure and the connection between the various chapters, where the arrows between two blocks indicate that the result of the source block is used/required by the target block

- **Chapter 1** – This corresponds to the current chapter, and it is an extended introduction to the Thesis. It provides some context for the Thesis and describes some fundamental concepts behind LF imaging. This is followed by the motivation for the developed work, as well as the defined objectives and the original contributions. At the end, it presents an overview of the structure followed in the Thesis.
- **Chapter 2** – This chapter tackles the first Thesis objective and proposes an efficient method to compute disparity maps for all LF views while ensuring the angular consistency of those maps. As mentioned above, LF depth/disparity information can guide other processing tasks, such as segmentation, and lead to more robust and realistic results by improving processing accuracy and angular consistency. Thus, the obtained disparity maps will be used/required in the following chapters.

- **Chapter 3** – This is the first of two chapters that tackle the second Thesis objective and proposes a 4D LF over-segmentation method that adaptively weights several features to generate segments that adhere to object boundaries and maintain angular consistency. The proposed method in this chapter is suitable for densely sampled 4D LFs since it assumes that each segment in the central view exists in all other LF views, which is a similar assumption as in the existing methods.
- **Chapter 4** – This chapter further extends the work related to the second Thesis objective and introduces the "hyperpixel" concept in 4D space. It also proposes a flexible 4D LF over-segmentation method that can be used for both dense and sparse LFs. The obtained 4D LF over-segmentation of Chapter 3 and Chapter 4 will be exploited in the following chapters to facilitate LF segmentation and edit propagation tasks.
- **Chapter 5** – This is the first of two chapters that tackle the third Thesis objective and proposes a semi-supervised 4D LF foreground-background segmentation method. This method is interactive and relies on the user's scribbles to select any object in the scene and apply segmentation to all LF views. While semi-supervised segmentation is required in some applications, fully unsupervised segmentation is also encouraged, especially with the advances in deep learning techniques. The following chapter overcomes the user's scribbles constraint in 4D LF mid-level segmentation.
- **Chapter 6** – This chapter continues the work related to the third Thesis objective and proposes an unsupervised and angularly consistent 4D LF segmentation method using a GNN. This method represents the 4D LF as a hypergraph based on 4D LF over-segmentation and uses a GNN model to perform unsupervised segmentation on 4D space. The segmented objects obtained from the proposed methods in Chapter 5 and Chapter 6 can be used to apply edits only for specific objects in 4D space or integrated with any extended reality applications.
- **Chapter 7** – This chapter tackles the fourth Thesis objective and proposes a method that achieves neural style transfer for 4D LFs. The proposed method uses a deep learning architecture that applies neural style transfer only to a subset of LF views and then propagates the edits into all other LF views consistently. To achieve that, this chapter requires the disparity maps for all LF views (Chapter 2) and over-segmented 4D LFs obtained using Chapter 3 or Chapter 4.

- **Chapter 8** – This is the final chapter, which concludes this Thesis by discussing the achievements and outlining possible directions for future work.

Chapter 2

Efficient Propagation Method for Angularly Consistent 4D Light Field Disparity Maps

Journal: IEEE Access

Date of Publication: June 2023

Volume: 11

Pages: 63463-63474

DOI: 10.1109/ACCESS.2023.3287920

IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received 31 May 2023, accepted 13 June 2023, date of publication 20 June 2023, date of current version 28 June 2023.

Digital Object Identifier 10.1109/ACCESS.2023.3287920

RESEARCH ARTICLE

Efficient Propagation Method for Angularly Consistent 4D Light Field Disparity Maps

MARYAM HAMAD, (Graduate Student Member, IEEE), CAROLINE CONTI, (Member, IEEE), PAULO NUNES, (Member, IEEE), AND LUÍS DUCLA SOARES, (Senior Member, IEEE)

Instituto de Telecomunicações, Instituto Universitário de Lisboa (ISCTE-IUL), Lisbon 1649-026, 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 Light Field (LF) imaging, since it conveys both spatial and angular scene information, can facilitate computer vision tasks such as depth/disparity estimation. Although disparity maps can be estimated for all LF views, most existing methods merely estimate depth/disparity for the central view and do not adequately deal with other LF views. However, having depth/disparity maps for all LF views can be useful for enhancing immersive multimedia applications, such as 3D reconstruction and LF editing. To overcome this limitation, in this paper, an efficient and occlusion-aware disparity propagation method is proposed. The proposed method generates disparity maps for all LF views given a single disparity map for one reference view (e.g., the central view). The disparity map for the reference view is propagated first into the four corner views to ensure angular consistency. Afterwards, an off-the-shelf existing disparity estimation model is used to fill any remaining holes in the corner views. Finally, disparity maps for the remaining views are recursively generated through a fast propagation step, which is followed by a final refinement step to regularize the generated disparity maps. The proposed method not only generates disparity maps for all LF views but also handles occlusions and ensures angular consistency. Experimental results on synthetic and real LF datasets with different disparity ranges, using several accuracy and angular consistency metrics, show outperforming or competitive results compared to the benchmark methods with a significant complexity reduction.

INDEX TERMS Light field disparity estimation, angular consistency, fast disparity propagation, deep learning.

I. INTRODUCTION

Light Field (LF) imaging has attracted increasing attention from researchers due to its ability to capture not only light intensity but also ray directions [1], [2], [3]. 4D LFs can be represented as an array of views (a.k.a. sub-aperture images) $I(x, y, u, v)$, where (x, y) are the spatial coordinates and (u, v) are the angular coordinates of each view. By fixing one angular and one spatial coordinate, an Epipolar Plane Image (EPI) (i.e., the unique 2D spatio-angular LF slice typically containing a regular structure with several oriented lines [1], [4]) can be obtained, as illustrated in Fig. 1. By exploiting the rich information captured by LFs and the possible LF representations, new capabilities are enabled, such as post-capture refocusing and depth

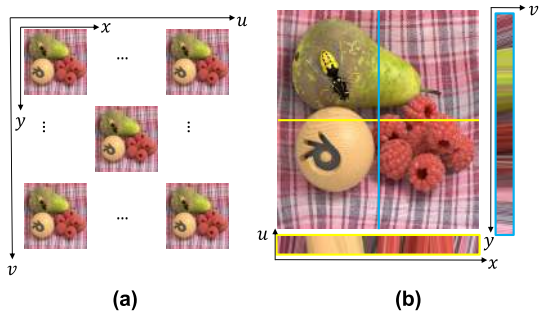


FIGURE 1. Example of LF representations. a) 4D LF represented as an array of views; b) Horizontal and vertical EPIs.

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

estimation. Additionally, disparity maps can be estimated from LFs to represent the displacement of corresponding pixels in several LF views, which is inversely proportional to the depth [4]. As humans, depth/disparity information estimated by our brains is exploited to efficiently process the surrounding world. Similarly, machines can benefit from depth/disparity information to enhance the processing of captured images. Therefore, many interesting applications for 2D content rely heavily on the use of depth maps (e.g., obtained from sensors) as an additional feature, besides the visual appearance, to apply efficient processing and editing.

In the case of 4D LF applications, besides achieving accurate editing, ensuring angular consistency is also essential. This is especially important when navigating between LF views using virtual reality headsets or LF displays. Therefore, generating angularly consistent disparity maps for all LF views has become a task of growing interest to guide several computer vision applications, such as LF segmentation, view synthesis, 3D scene reconstruction, and augmented/mixed reality [3], [5], [6], [7].

Several LF disparity estimation methods have already been proposed in the literature, e.g., [8], [9], [10], [11], [12], [13], [14], [15], [16], and [17], as briefly reviewed in Section II. Most existing LF disparity estimation methods estimate disparity maps only for the central view. However, having disparity maps for all LF views can be useful for enhancing several applications, such as 3D reconstruction and LF editing. The few methods that consider estimating disparity for all LF views, e.g., [10], [16], and [17], either are not adequately considering consistency across LF views, are computationally complex, or are only suitable for densely sampled LFs.

In this context, the main contribution of this paper is an efficient disparity propagation method that generates angularly consistent disparity maps for all LF views, which works for both densely and sparsely sampled LFs. The proposed method exploits the correlations across LF views and propagates a given disparity map from only one reference view into all LF views in an occlusion-aware manner, while also ensuring angular consistency. The proposed propagation method starts by propagating the reference view disparity map into the corner views and assigning disparity values to all their pixels. Afterwards, disparity values are recursively propagated to the remaining LF views with the guidance of the reference and corner views in both horizontal and vertical angular directions. Finally, a last refinement step is included to smooth the disparity maps. Experiments using different accuracy, consistency and complexity metrics show outperforming or competitive results when compared to the existing methods, while reducing the computational complexity.

The remainder of this paper is organized as follows. Section II briefly reviews the related work on 4D LF disparity estimation, Section III describes the proposed method in detail and Section IV evaluates its performance through a series of experimental results. Finally, Section V concludes the paper with some final remarks and directions for future work.

II. RELATED WORK

In recent decades, several 4D LF disparity estimation methods have been proposed. Existing methods can be classified as either classical or deep learning-based 4D LF methods depending on the used disparity estimation approach:

A. CLASSICAL 4D LF DISPARITY ESTIMATION METHODS

Classical 4D LF disparity estimation methods exploit different LF representations and analyze the geometry to estimate disparity information using manually designed features. This type of methods can be further classified into three categories, according to the used LF representation:

- **EPI-based methods:** the methods in this category rely heavily on the EPI regular structure. In the EPI space, a 3D point is represented by a line whose slope is inversely proportional to its disparity value [1], [4]. Wanner and Goldluecke [8] analyzed EPIs using structure tensors to locally estimate disparity values. Zhang et al. [9] proposed a spinning parallelogram operator for depth estimation on EPI space. Khan et al. [10] proposed a disparity estimation method to compute disparity maps for all LF views by detecting EPI edges and diffusing them spatially within the central view and then propagating the central view into all LF views. EPI-based methods typically achieve high estimation accuracy, but only for densely sampled LFs.
- **Sub-aperture image-based methods:** the methods in this category rely on matching corresponding pixels between LF views, i.e., stereo matching, using a robust patch-based block-matching approach. A cost volume is usually constructed to measure the similarity and angular consistency between LF views. Jeon et al. [11] proposed a disparity estimation method by applying the phase shift theorem. Huang et al. [12] proposed an empirical Bayesian framework for computing LF disparity for both dense and sparse LFs. While matching corresponding pixels to estimate disparity maps is widely used, in dense LFs with a quite narrow baseline, sub-aperture image matching can lead to poor accuracy and occlusions can cause impossible correspondences [2].
- **Focal stack-based methods:** the methods in this category produce a focal stack from LFs and rely on defocus cues to estimate the disparity. They assume that in-focus points are projected at the same spatial location in the different views [13], [14], [15]. Lee and Park [15] proposed a unified model for depth estimation by combining focus, defocus and matching corresponding pixels. The methods that rely on LF focal stack are robust to occlusions and noise. However, they may suffer from ambiguities due to the used patch and

aperture sizes, making the approach not as accurate as most methods in the previous categories.

B. DEEP LEARNING-BASED 4D LF DISPARITY ESTIMATION METHODS

Deep learning-based 4D LF disparity estimation methods have been recently proposed to improve the performance of existing classical methods while greatly reducing the disparity estimation time. These methods rely on deep learning techniques and most of them are supervised by ground truth disparity maps to estimate disparity information. Heber and Pock [18] proposed the first convolutional neural network model to learn an end-to-end mapping between a 4D LF and its corresponding depths. Afterwards, Heber et al. proposed a U-Net architecture with 3D convolutions to estimate LF disparity maps for the central EPIs [19]. The EPI-based fully-convolutional neural Network (EPINet) and Multi-scale Aggregated Network (MANet) proposed in [20] and [21] significantly improved the disparity estimation accuracy for the central view and heavily rely on the EPI structure in densely sampled LFs. Shi et al. [16] overcame this limitation by proposing a framework that can be used for both dense and sparse LFs. While this method can estimate a disparity map for any LF angular location, angular consistency across views is not ensured. Jiang et al. [17] proposed a disparity estimation method starting from the four corner views. After that, the disparity is propagated into all other LF views and a 3D reconstruction method is used to fill the holes (i.e., remaining regions after propagation without disparity values). Although it can estimate disparity maps for both dense and sparse LFs, relying on the initial estimation of corner views can significantly affect the disparity estimation performance for wide baseline LFs. Wang et al. [22] proposed a generic mechanism for LF processing including disparity estimation using domain-specific convolutions. Recently, Chao et al. [23] proposed a disparity estimation method called SubFocal that learns the disparity distribution of dense LFs and estimates a smooth disparity map for the central view by using cost volumes at the sub-pixel level. Supervised deep learning-based methods achieve state-of-the-art results. However, they require a large number of training LFs with ground truth disparity maps, which are challenging to obtain in the real world. Moreover, training deep learning models using only synthetic LF datasets may not adequately handle the domain shift between the real world and synthetic datasets. Therefore, several unsupervised methods are proposed to handle this challenge, although the performance is slightly reduced [24].

III. PROPOSED DISPARITY PROPAGATION METHOD

The proposed method comprises three main steps as illustrated in Fig. 2. To start, two inputs are required, namely a 4D LF and an estimated disparity map of one reference view with respect to its adjacent right view (estimated by any available method). In this paper, the central view is selected as a reference view since it is equidistant from all corner views. Hence, represents a good compromise in terms of the remaining holes after propagating its disparity map into all corner views. Therefore, from hereinafter in this paper, the central view will be considered the reference view. Notice, however, that the proposed method can use any disparity estimation method and any angular location for the reference view, though the results may be affected accordingly, as explained in Section IV.

To apply the propagation into all LF views, in the first step, the input reference view disparity map is propagated into the four corner views in an occlusion-aware manner. The remaining holes in the corner views after propagation are filled by estimating their disparity values. Any disparity estimation method that can estimate disparity for any angular location for dense and sparse LFs can be used to fill the holes. In the second step, the disparity maps for the remaining LF views (i.e., all LF views except the reference and corner views) are generated via a recursive propagation in both horizontal and vertical directions separately. Afterwards, disparity maps from both horizontal and vertical propagation are fused for each view using their arithmetic mean.

In the last step, the disparity values of any remaining holes are computed, and a final edge-preserving refinement is applied to further regularize the output. The following subsections describe these steps in more detail.

A. DISPARITY PROPAGATION FOR CORNER VIEWS

To ensure angular consistency across LF views, the reference view disparity map, d^{ref}, is initially propagated into the four corner views (since they typically include most of the scene information) – in this paper, the central view located in the angular location (u_c, v_c), is used as reference view; thus $ref=(u_c, v_c)$. The propagation is achieved by assigning the same disparity value of each pixel in d^{ref} to the corresponding pixel in each corner view, which are computed using d^{ref} itself, as shown in (1):

$$\begin{cases} x^{(u,v)} = x^{ref} + d_{hor}^{ref\rightarrow(u,v)}, \\ y^{(u,v)} = y^{ref} + d_{ver}^{ref\rightarrow(u,v)}, \end{cases} \tag{1}$$

where x^{ref}, y^{ref} are the spatial coordinates from which the propagation is applied; $x^{(u,v)}$, $y^{(u,v)}$ are the corresponding spatial coordinates of x^{ref}, y^{ref} in view (u, v); $d_{hor}^{ref\rightarrow(u,v)}$, $d_{ver}^{ref\rightarrow(u,v)}$ are the horizontal and vertical disparity values located in the spatial position $\left(x^{ref}, y^{ref}\right)$ from the reference view to view (u, v). To ensure integer positioning, rounding is applied to the projected coordinates. Assuming a regular arrangement of cameras with a parallel optical axis and uniform camera baseline and focal length, as assumed in [25], and [26], the horizontal and vertical disparities from the reference view into any other LF view (u, v) is computed using (2). Equations (1) and (2) hold under the above assumption. Otherwise, camera parameters must be

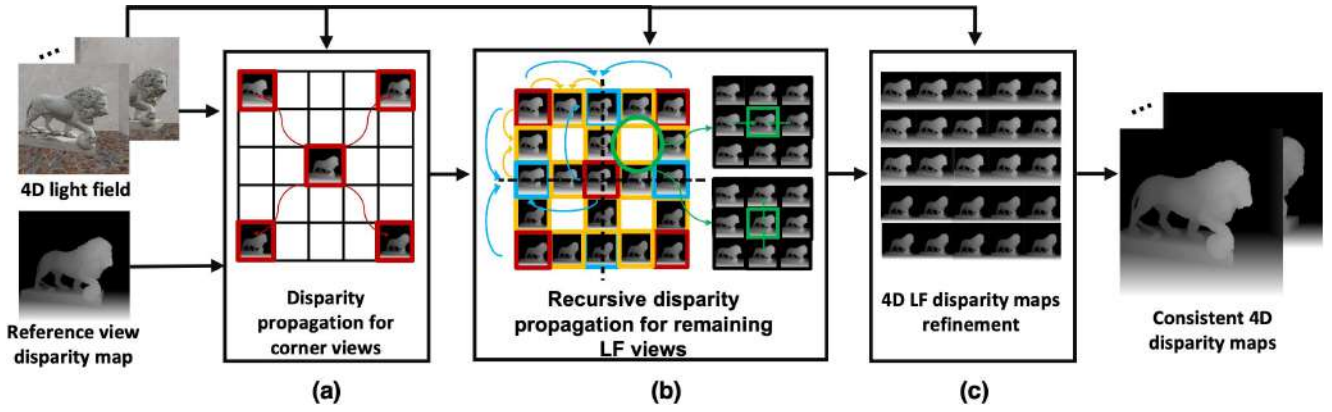


FIGURE 2. Illustration of the proposed disparity propagation method: a) Disparity propagation for corner views; b) Recursive disparity propagation for all remaining LF views to ensure angular consistency; c) Disparity map refinement for all LF views.

considered.

$$\begin{cases} d_{hor}^{ref\to(u,v)} = d^{ref} \times (u - u_c), \\ d_{ver}^{ref\to(u,v)} = d^{ref} \times (v - v_c)\,. \end{cases} \tag{2}$$

To detect occlusions, a binary occlusion mask is initialized for each LF view for all pixels, where each pixel is initially labeled as occluded. After estimating the disparity map of the reference view and propagating it into other LF views, all spatial locations that have disparity values are labeled as non-occluded and the remaining ones keep the initial occluded label.

To ensure occlusion-aware propagation, the input 4D LF and corresponding texture variation maps are used. Therefore, the input 4D LF is converted to the CIELAB color space. Then, a per-pixel texture variation map is generated from the L channel by computing the local standard deviation of a (3×3) neighborhood for all the pixels in each view. The texture variation maps are used to guide the propagation when different objects share either the same color or the same disparity values. Disparity propagation is applied only if the color and texture difference, D, between pixels in *ref* view and corresponding pixels in another view, as in (3), is less than or equal to a pre-defined threshold, τ, (i.e., $D \leq \tau$). In this paper, τ is set to 0.01 after extensive experiments to allow for a reasonable difference due to rounding and lighting differences in each view.

$$D = \sqrt{(l_i - l_j)^2 + (a_i - a_j)^2 + (b_i - b_j)^2 + (t_i - t_j)^2}, \tag{3}$$

where l, a, b are normalized color channel values (using min-max normalization [27]) in CIELAB color space; i, j represent the original pixel in *ref* view and corresponding pixels in (u, v) view, respectively; and t is the normalized texture value (using min-max normalization) for each pixel. The above thresholding operation is beneficial in preventing inaccurate projections into other views if the reference disparity maps have inaccurate values. Different values of τ are tested to study their effect in Section IV.

During propagation, the occlusion mask is checked for each spatial location, and when it has already a non-occluded label (i.e., another candidate has been propagated into the same location due to rounding, or inaccurate estimation), the maximum disparity value between the previous and current candidates is kept only if $D \leq \tau$, otherwise the disparity value will not be changed. The rationale for keeping the maximum disparity value comes from the observation that foreground objects, which are typically not occluded, have larger disparity values.

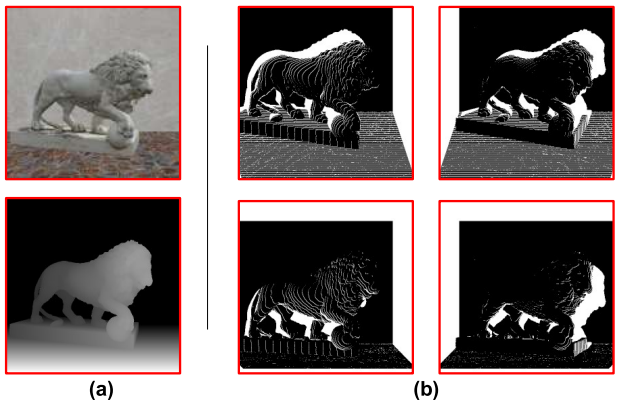


FIGURE 3. Example of occlusion masks after projecting the disparity map of the central view into other LF views: a) The central view and its disparity map; b) Binary occlusion masks of the four corner views after propagating the central disparity map into each one. White pixels indicate pixels without disparity values, i.e., occluded/invisible pixels (relatively to the central view).

The remaining holes in the corner views (white regions in Fig. 3b), i.e., regions without disparity values, need to be filled next. Instead of applying a blind filling/inpainting to those holes, the actual disparity values are truly estimated from the input 4D LF. Therefore, any existing disparity estimation method that can compute disparity maps for any angular location, and not only for the central view, for both dense and sparse LFs, can be used to fill the remaining holes. Different disparity estimation methods are used and evaluated in Section IV to study their effect on the estimated disparity maps. After assigning disparity values for all pixels in corner views, the corner views are used to guide the propagation for the remaining LF views, as explained in the next step.

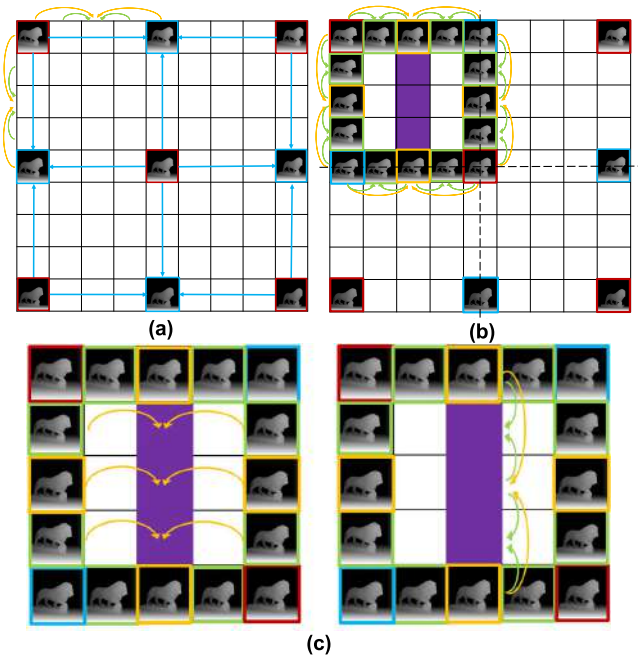


FIGURE 4. Disparity map estimation via propagation: a) Middle views between any two corner views (blue squares); b) Border and central crosshair views (yellow and green squares) using a recursive propagation; c) Internal views, such as purple shaded squares, have two different propagations from horizontal and vertical directions independently, then both computed disparity maps are fused to create one disparity map for each internal view.

B. RECURSIVE DISPARITY PROPAGATION FOR REMAINING LF VIEWS

In this step, disparity maps of all other LF views are obtained in three stages as presented in Fig. 4.

First, a disparity map is assigned to each middle view located halfway between any two corner views (blue squares in Fig. 4a). To achieve that, the reference view and the nearest two corner views are propagated into each middle view as shown in Fig. 4a (considering occlusions as in the previous step). Those three propagated disparity maps are then fused by considering their arithmetic mean.

Second, a disparity map is assigned to each remaining border and crosshair view, i.e., views in the central horizontal and vertical angular coordinates (yellow and green squares in Fig. 4b). To achieve that, a recursive propagation from two reference views located in the same horizontal or vertical angular dimension into the middle view located halfway between them is applied, as shown in Fig. 4b, until no more middle views without disparity values remain. At this point, the LF is divided into four quadrants and no disparity maps are still assigned for the internal views of each quadrant.

Third, a disparity map is assigned to the internal views in each quadrant by applying a recursive horizontal and vertical propagation independently, as shown in Fig. 4c (the same way as it has been done in the second stage for each row or column of internal views). Both disparity maps generated from horizontal and vertical recursive propagation are then fused for each internal LF view by considering their arithmetic mean.

TABLE 1. Test datasets used in the experiments.

4D LF dataset for testing	Disparity range
EPFL real world [29]: Fruits, Swans	$[-1, 1]$
Stanford real world [30]: Lego, Bunny	$[-3, 3]$
HCI synthetic [31]: Buddha, Papillon, Horses, StillLife	$[-4, 4]$
HCI* synthetic [28]: Table, Dishes (* for distinguishing the two different HCI datasets)	$[-4, 4]$
Inria synthetic sparse LFs [16]: Lion, Electro devices	$[-20, 20]$

C. 4D LF DISPARITY MAPS REFINEMENT

In this step, all LF views already have a disparity map. However, remaining hole locations, caused by occluded regions that do not exist in either the reference view or the corner views, or just caused by rounding the coordinates to integer indexing, need to be filled. Therefore, the four nearest left, right, top and bottom spatial neighbors that have disparity values for each pixel are considered in each view. The disparity value corresponding to the minimum difference $D_k < \tau, k = 1, \ldots, 4$ as in (3), is assigned as the disparity value of that location. If more than one neighbor has an equal D value, the minimum disparity value is considered. The reason for considering the minimum disparity is that the remaining holes, typically belonging to occluded regions, cannot be seen in the reference or corner views, are deeper than frontal objects (i.e., occluding objects), and hence have lower disparity values. After filling the remaining isolated pixels and holes, a simple and fast 2D edge-preserving median filtering using a (5×5) kernel size is applied spatially for all LF views to refine the estimated disparity maps.

IV. RESULTS AND EVALUATION

In this section, the proposed method is compared to several benchmark methods, namely: i) Shi et al. method [16], which is applied for each LF view independently since it only estimates one disparity map for any angular location; ii) Jiang et al. method [17]; and iii) Khan et al. method [10]. Both [10] and [17] create disparity maps for all LF views.

Moreover, several LF datasets with different disparity ranges are used (see Table 1). Notice that HCI and HCI∗ datasets are both synthetic dense LF datasets, however, they are different in the disparity ranges and in the spatial resolution. The entire (9×9) views for all datasets are considered. Only synthetic LFs with Ground Truth (GT) disparities for all LFs are used for the quantitative evaluation. The EPFL and Stanford LF datasets do not have GT disparity maps and, hence, the quantitative evaluation is not applied to them.

To quantitatively evaluate the proposed method, three different metrics are used, namely: i) Mean Square Error

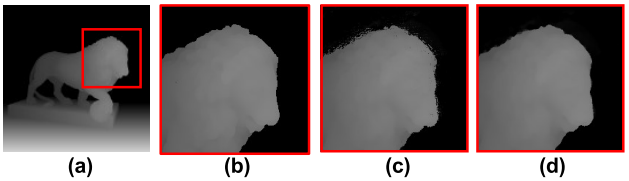


FIGURE 5. A visual example of the refinement step using edge-preserving filtering. a) GT disparity of view (3, 6); b) GT disparity enlargement; c) Proposed without refinement; d) Proposed with final refinement.

(MSE) [28]; ii) Percentage of Bad Pixels (BP) (i.e., the percentage of pixels with a disparity error above a certain threshold; the typically used Bad Pixels error thresholds of 0.01, 0.03 and 0.07 are used) [28]; and iii) View Consistency Error (i.e., where disparity maps of all LF views are projected into each angular location; then the variance of all projected disparity maps is computed for each angular location (81 values) as defined, formulated and implemented in [10]).

As explained in Section III, in the proposed method. a reference view disparity map is required as input. Additionally, the holes in corner views after propagating the reference view disparities need to be filled by estimating their disparities. Therefore, any existing method that can estimate disparity maps for any angular location in dense and sparse LFs can be used (more accurate is favored). In this paper, to achieve both accuracy and angular consistency for the generated disparity maps, Chao et al. SubFocal method [23] is adopted to estimate the input disparity map. The reason for choosing the SubFocal method is that it ranks first place, as reported by the authors, among other 99 submitted methods on the HCI 4D LF benchmark [28] considering different metrics. However, the SubFocal method can estimate a disparity map only for the central view and has been trained for dense LFs with a disparity range of $[-4, 4]$. Therefore, the remaining holes after propagation to the corner views are filled by using the pre-trained model of Shi et al. [16] that fine-tuned the optical flow estimation network (a.k.a. FlowNet 2.0) [32] for LF disparity estimation in any angular location. Moreover, to consider sparse LFs, the model in [23] is retrained in our experiments by using LFs with a wider disparity range (i.e., $[-20, 20]$). To retrain the SubFocal method for sparse LFs, the Inria synthetic sparse LF dataset in Table 1 was used for training (the same number of LFs was used for training, i.e., 16 LFs, as in [23]). The hyperparameters were kept the same as in [23], except for the disparity sampling step size which was set to 2.5 instead of 0.5 to reduce the cost volume complexity. In this paper, the retrained model is tested for sparse LFs using Lion, and Electro devices test LFs. In the experiments, different methods are also used to study the effect of the selected disparity estimation method on the proposed propagation method (including the use of the method [16] for both the reference and corner views).

Notice that some results are not available (indicated n/a in Fig. 7, Fig. 8 and Fig. 10) since the EPI-based method in [10] does not support LFs with large disparity ranges. Additionally, the SubFocal method [23] is used to estimate the input reference view disparity map for most datasets. However, disparity maps of the HCI dataset [31] are estimated using the Shi et al. method [16] since the SubFocal method [23] generates inaccurate disparity estimations for this dataset due to the different resolutions and the domain shift of training and testing, as reported by the authors and shown later in Fig. 9.

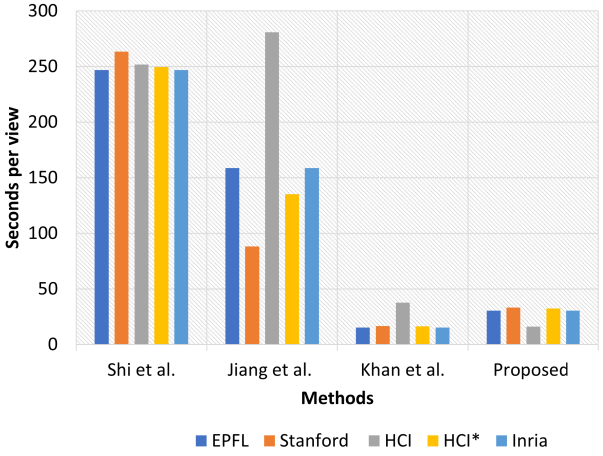


FIGURE 6. Average CPU time in seconds per view.

The proposed method is implemented using MATLAB and all results ran on a desktop computer with a 64-bit Ubuntu operating system, AMD® Epyc 7282 16-core CPU, NVIDIA GeForce RTX 3090 and 256 GB RAM.

A. QUANTITATIVE AND QUALITATIVE RESULTS

In this section, the proposed method results are presented and compared to the benchmark methods using several datasets with various disparity ranges. Notice that only the datasets that have ground truth disparity maps are used in the quantitative results namely, HCI [31], HCI* [28] and Inria [16] LF datasets.

Initially, the effect of the parameter τ is studied by using different values and finding the evaluation metrics accordingly. Therefore, 4 different experiments are conducted where different values of τ are used, i.e., 0.001, 0.01, 0.1 and ∞, where ∞ refers to the case where the visual consistency is discarded during the propagation. As can be seen from Table 2 using different τ values can slightly affect the accuracy and the CPU time. As illustrated in Section III, the occlusion-aware propagation step considers the disparity values of foreground and background regions. However, to avoid wrong projection for objects that have similar disparity values but are different in color, the τ value is set to 0.01. This value prevents inaccurate projection from occurring due to discrete sampling, rounding errors or inaccurate estimated values. The chosen value allows for reasonable visual differentiation across LF views, accounting for varying lighting conditions. Simultaneously, it strikes a reasonable balance between accuracy, efficiency, and prevention of inaccurate projections.

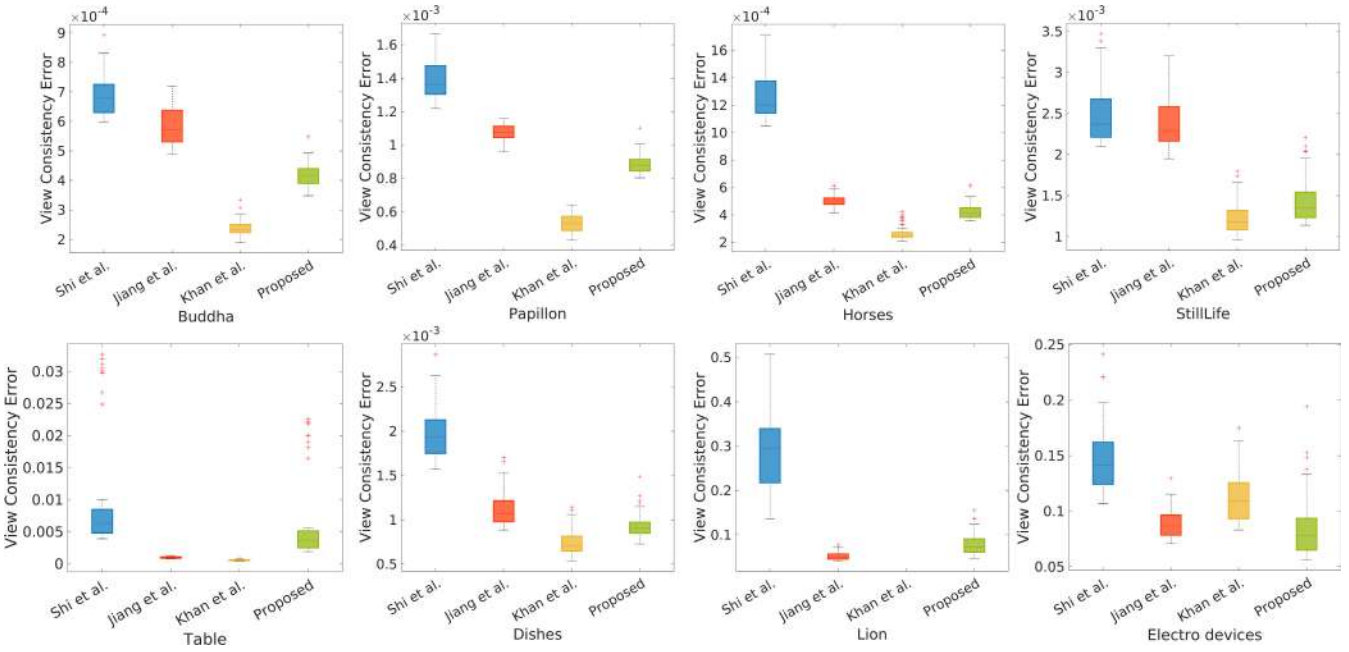


FIGURE 7. Summary statistics of view consistency error across all LF views for each test LF.

TABLE 2. Average quantitative results using the proposed method with different τ values on various LF datasets for all LF views.

Metric	LF \ τ	0.001	0.01	0.1	∞
MSE × 100 (↓)	HCI [31]	0.68	**0.66**	0.73	0.76
	HCI* [28]	1.63	1.49	0.83	**0.83**
	Inria [16]	**39.16**	41.52	44.43	45.53
BP 0.01 (↓)	HCI [31]	62.16	**62.10**	62.11	62.12
	HCI* [28]	22.32	19.74	19.60	**19.58**
	Inria [16]	**87.43**	87.87	88.01	88.03
BP 0.03 (↓)	HCI [31]	25.78	**25.75**	25.82	25.85
	HCI* [28]	10.04	9.07	8.73	**8.70**
	Inria [16]	**64.58**	65.44	65.83	65.88
BP 0.07 (↓)	HCI [31]	**16.76**	17.27	17.46	17.52
	HCI* [28]	5.35	4.94	4.69	**4.67**
	Inria [16]	**31.63**	32.09	32.64	32.73
Average CPU time in seconds per view	HCI [31]	17.11	16.01	15.97	**15.92**
	HCI* [28]	32.93	32.46	32.43	**32.41**
	Inria [16]	33.63	33.18	33.14	**33.12**

TABLE 3. Average quantitative results on various LF datasets using different 4D LF disparity estimation methods for all LF views.

Metric	LF \ Method	Shi et al. [16]	Jiang et al. [17]	Khan et al. [10]	Proposed without refine.	Proposed with refine.
MSE × 100 (↓)	HCI [31]	0.70	0.98	1.20	0.70	**0.66**
	HCI* [28]	2.14	1.66	3.95	**1.46**	1.49
	Inria [16]	67.62	2104.39	118.40	44.50	**41.52**
BP 0.01 (↓)	HCI [31]	63.11	**58.25**	75.08	62.26	62.10
	HCI* [28]	50.93	48.53	75.38	**19.40**	19.94
	Inria [16]	**82.90**	96.30	94.41	88.38	87.87
BP 0.03 (↓)	HCI [31]	27.01	**24.66**	39.68	26.05	25.75
	HCI* [28]	25.41	25.30	45.98	**8.90**	9.07
	Inria [16]	**59.09**	89.26	83.58	66.72	65.44
BP 0.07 (↓)	HCI [31]	7.50	8.28	15.87	7.04	**6.82**
	HCI* [28]	14.09	16.34	25.86	4.98	**4.94**
	Inria [16]	37.82	80.37	64.79	34.01	**32.09**

To study the impact of the final refinement step, Table 3 shows the performance of the proposed method with and without applying the edge-preserving filter to refine the generated disparity maps in the refinement step (i.e., step C in Section III). As presented in Table 3, the results are slightly improved when a simple median filter is applied to all LF views to regularize the estimated disparity maps in most datasets. The used filter is simple, fast, preserves scene edges and can reduce inaccurate disparity propagations, especially for a few pixels with wrong disparity values surrounded by pixels with accurate ones (see Fig. 5).

Moreover, the proposed method is evaluated and compared to the benchmark methods using various datasets, as presented in Table 3, Table 5, Fig. 6, Fig. 7 and Fig. 8.

To compare the computational complexity between the various methods, all methods were run using the CPU, and CPU times are reported in Fig. 6. The reported time for the proposed method includes the disparity estimation time for the reference view and for occlusions in corner views and all the steps in Section III. The CPU time spent by all the benchmark methods for all LF views is then divided by the number of views to obtain an average CPU time per view. The breakdown of the average CPU time for each step is

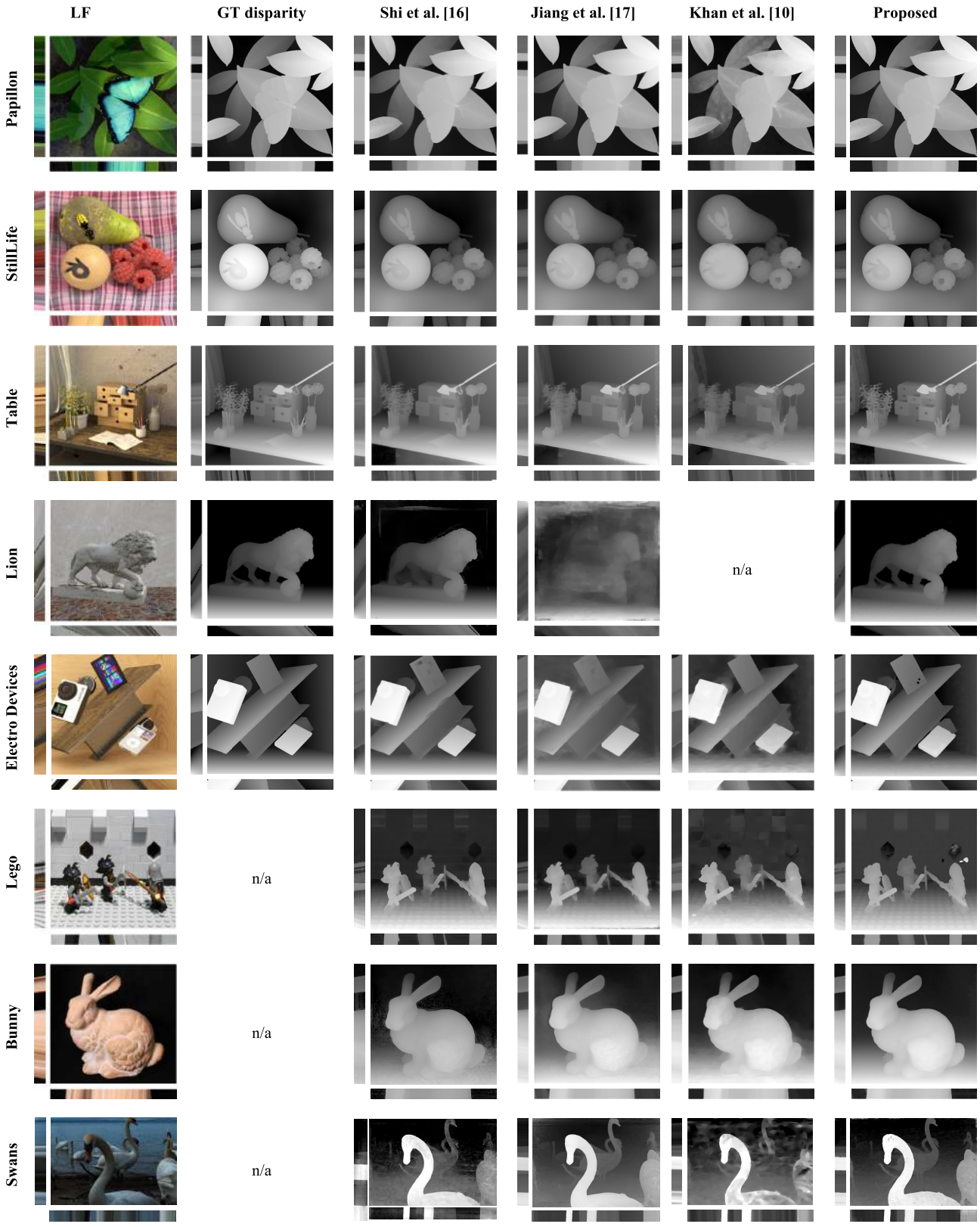


FIGURE 8. Visual comparison using the proposed disparity propagation method and the benchmark methods for dense and sparse LFs. The central view and central horizontal and vertical EPIs are shown for all LFs. Not available (n/a) results for the Khan et al. method since it does not support very sparse LFs.

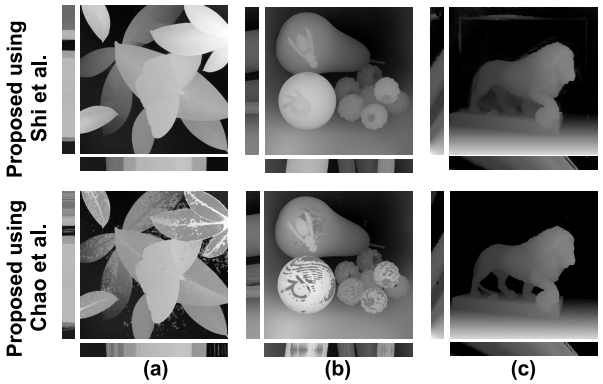


FIGURE 9. Examples to show the effect of the selected central view disparity estimation method on the proposed disparity propagation method. First row, using the Shi et al. method [16]; Second row, using Chao et al. SubFocal method [23].

TABLE 4. Breakdown of the average CPU time for the proposed method (in seconds).

4D LFs dataset for testing	(A) Disparity estimation for the reference view	(B) Disparity estimation for the 4 corner views	(C) Disparity propagation into all 81 views	Average CPU time per view $\frac{(A)+(B)+(C)}{81}$
EPFL [29]	1456.38	987.36	19.19	30.41
Stanford [30]	1613.42	1053.61	22.15	33.20
HCI [31]	251.72	1006.90	38.04	16.01
HCI* [28]	1612.37	998.05	18.70	32.46
Inria [16]	1612.22	1050.91	24.27	33.18

reported in Table 4. As in Table 3 and Fig. 6, the results of the proposed method generate competitive accuracy results while reducing the complexity when compared to the benchmark methods, especially for challenging sparse LFs.

Besides the improvements of the proposed method in terms of accuracy metrics in some datasets, a significant reduction in time is shown in Fig. 6 when compared to Shi et al. [16] and Jiang et al. [17]. Compared to Khan et al. [10], the CPU time results are still competitive. However, when the Khan et al. method is used for the reference and corner views, the proposed method requires less CPU time than [10], as shown later in this section. This reduction in time is achieved by exploiting the correlation between LF views and applying angularly consistent propagation. The CPU time for the proposed method depends on the used methods to estimate the reference view disparity map and the occluded regions as described below in this section.

Regarding the angular consistency of obtained disparity maps, Fig. 7 shows the angular consistency using boxplots where the central mark indicates the median and the bottom and top edges of the box indicate the 25th and 75th percentiles, respectively. Notice that, in this experiment, two different disparity estimation methods are used during the propagation steps in the proposed method (e.g., the SubFocal

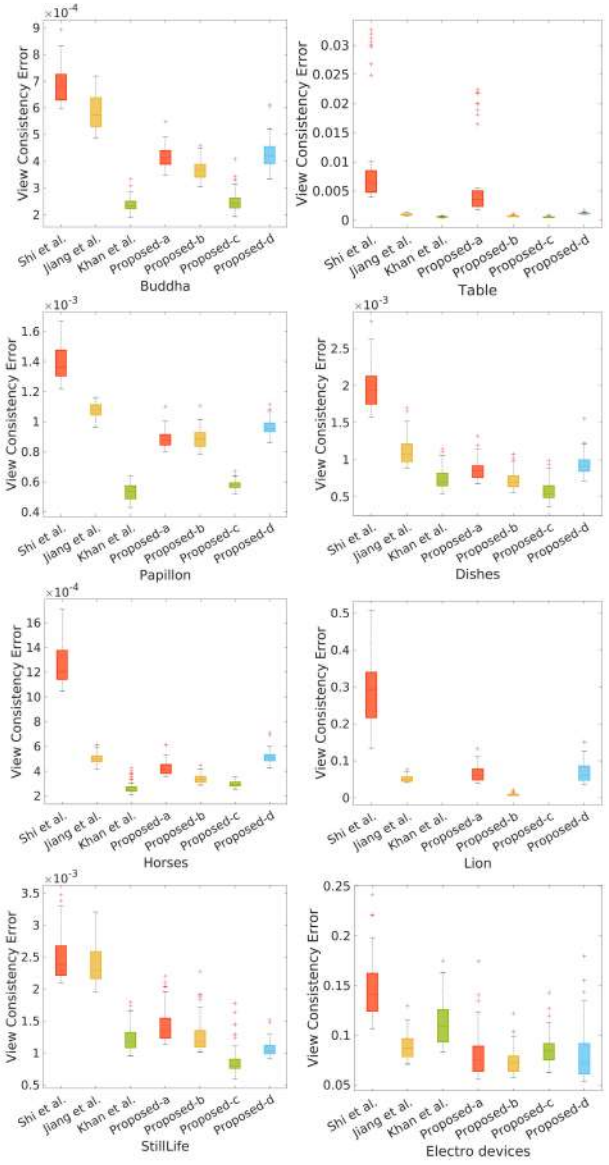


FIGURE 10. Summary statistics of view consistency error across all LF views for each test LF. In this figure, the results of the proposed method are compared using the benchmark method to estimate the disparity map for the reference view and occluded regions in the corner view namely, using Shi et al. method (Proposed-a); using Jiang et al. method (Proposed-b); using Khan et al. method (Proposed-c); and using the ground truth disparity (Proposed-d). The proposed disparity propagation method leads to better view consistency compared to the original benchmark methods.

method [23] is used for the reference view and the pre-trained model of Shi et al. [16] is used for occlusions in the corner views). Despite that, the proposed method outperforms Shi et al. [16] by ensuring angular consistency during the propagation as can be noticed in Fig. 7. Additionally, the proposed method outperforms Jiang et al. method [17] in 6 LF scenes, while their method outperforms the proposed method in the remaining 2 LFs (i.e., Table, and Lion). Khan et al. method [10] outperforms the proposed method for dense LFs and ensures angular consistency across views. However, the method in [10] does not adequately support large occlusions

TABLE 5. Average quantitative results on various LF datasets using different 4D LF disparity estimation methods for all LF views. The propagation results are tested using different disparity estimation methods for the reference view and hole filling in the coroner views, including the ground truth disparity. The best results among the proposed method results using different estimation techniques are highlighted in bold.

Metric	LF \ Method	Shi et al.[16]	Proposed using Shi et al. [16]	Jiang et al. [17]	Proposed using Jiang et al. [17]	Khan et al. [10]	Proposed using Khan et al. [10]	Proposed using ground truth
MSE× 100 (↓)	HCI [31]	0.70	**0.66**	0.98	0.96	1.20	1.11	0.28
	HCI* [28]	2.14	2.14	1.66	**1.70**	3.95	3.89	0.51
	Inria [16]	67.62	**68.44**	2104.39	2129.15	118.40	111.78	8.96
BP 0.01 (↓)	HCI [31]	63.11	62.10	58.25	**58.26**	75.08	74.94	10.37
	HCI* [28]	50.93	**46.38**	48.53	48.87	75.38	75.44	5.89
	Inria [16]	82.90	**80.48**	96.30	96.28	94.41	94.12	19.20
BP 0.03 (↓)	HCI [31]	27.01	25.75	24.66	**24.69**	39.68	39.40	1.72
	HCI* [28]	25.41	**23.17**	25.30	25.54	45.98	45.76	2.62
	Inria [16]	59.09	**55.15**	89.26	89.14	83.58	82.65	7.26
BP 0.07 (↓)	HCI [31]	7.50	**6.82**	8.28	8.20	15.87	15.54	0.65
	HCI* [28]	14.09	**13.15**	16.34	16.31	25.86	25.43	1.78
	Inria [16]	37.82	**33.33**	80.37	80.07	64.79	63.50	2.11

across LF views, which is typical for sparse LFs. While the view consistency of disparity maps is essential, accuracy is also important. As can be seen from Fig. 7 and Fig. 8, for the Lion LF, for instance, achieving the best performance in terms of view consistency does not necessarily lead to better visual accuracy and in terms of other metrics. Besides the visual results shown in Fig. 8, readers are encouraged to see also the dynamic results in our GitHub repository.[1]

To study the effect of the selected disparity estimation method on the proposed propagation method, different disparity estimation methods are used for estimating the reference view and filling the occlusions in the corner views. To achieve that, the benchmark methods [10], [16], [17] and the ground truth disparity are compared to the proposed method results generated by using each benchmark method for estimating the reference view and filling the holes in the corner views. After that, the proposed propagation method is applied to compute disparity maps for all LF views.

As can be seen in Table 5, by only using one reference disparity map and by exploiting the correlations across LF views, the proposed propagation method outperforms the original benchmark methods in most test LFs and can generate competitive results in others. Moreover, the proposed propagation method ensures better view consistency than the original benchmark methods in most LFs as presented in Fig. 10. Notice that, for some LFs, using the ground truth disparity in the proposed method has lower performance, in terms of the view consistency metric, compared to the estimated ones, as shown in Fig. 10. The reason for this is that the ground truth disparity is more distinct and sharper around objects boundaries when compared to the smooth estimated ones. Hence, it generates larger and sharper holes

[1]Dynamic results for all LF views can be found at: https://github.com/MaryamHamad/LFDisparityPropagation

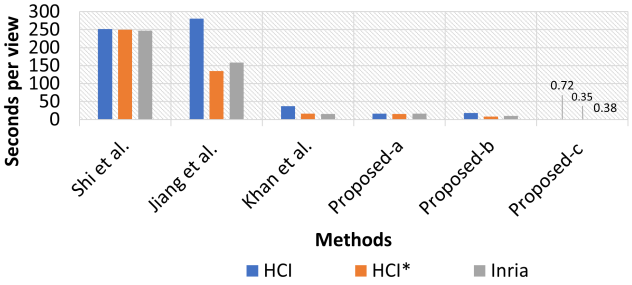


FIGURE 11. Average CPU time in seconds per view. In this figure, the results of the proposed method are generated by using the benchmark method to estimate the disparity map for the reference view and occluded regions in the corner views namely, using Shi et al. (Proposed-a); using Jiang et al. (Proposed-b); and using Khan et al. (Proposed-c). The proposed disparity propagation method significantly reduces the required time compared to the original benchmark methods.

in occluded regions that need to be filled after propagation. Small differences in filling those regions across LF views can heavily affect the consistency metric results. Finally, the proposed method can drastically reduce the average CPU time per view when compared to the benchmark methods, as shown in Fig. 11.

To sum up, the proposed disparity propagation method enables computing an accurate disparity map for each LF view only from one reference view disparity map and hole filling in the corner views. The proposed method leads to improved accuracy and view consistency for most of the LF datasets and reduces the computational complexity compared to the benchmark methods. Some limitations remain such as if the input reference view has inaccurate estimation, there is no correction step to check if the values are accurate or not, and the inaccurate values will be propagated into all other LF views as shown in Fig. 9. This limitation can be avoided by using an accurate disparity estimation method to estimate the disparity map for the reference view.

V. FINAL REMARKS

In this paper, an efficient disparity propagation method is proposed to generate angularly consistent disparity maps for all LF views. Given only one estimated disparity map of a reference view, the proposed method exploits the correlation across LF views and propagates the reference disparity map to the corner views at first. The remaining holes in the corner views are not interpolated but truly estimated by adopting an off-the-shelf disparity estimation method. Afterwards, disparity maps of the reference and corner views are propagated recursively in horizontal and vertical angular directions in an occlusion-aware manner into all remaining LF views. Finally, a refinement step is included to regularize the final disparity maps and fill any remaining holes. Since most of the existing methods estimate disparity information for the central view only, the proposed method can be used as plug and play with them to enable the generation of angularly consistent disparity maps for all LF views. Experimental results for several LF datasets with different disparity ranges show competitive results in terms of angular consistency and estimation accuracy compared to the existing methods with a significant complexity reduction.

For future work, the question of how to adaptively select the location of the reference view and the possibility of adding more reference views will be investigated to effectively consider occlusions based on the LF disparity range. Moreover, the current implementation is not optimized yet and the computational complexity of the proposed disparity propagation method can be further reduced.

REFERENCES

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

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

[3] S. Zhou, T. Zhu, K. Shi, Y. Li, W. Zheng, and J. Yong, "Review of light field technologies," *Vis. Comput. Ind., Biomed., Art*, vol. 4, no. 1, pp. 1–13, Dec. 2021.

[4] 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. 7–55, 1987.

[5] M. Hamad, C. Conti, P. Nunes, and L. D. Soares, "ALFO: Adaptive light field over-segmentation," *IEEE Access*, vol. 9, pp. 131147–131165, 2021.

[6] D. Egan, M. Alain, and A. Smolic, "Light field style transfer with local angular consistency," in *Proc. IEEE Int. Conf. Acoust., Speech Signal Process. (ICASSP)*, Toronto, ON, Canada, Jun. 2021, pp. 2300–2304.

[7] M. Hamad, C. Conti, A. M. de Almeida, P. Nunes, and L. D. Soares, "SLFS: Semi-supervised light-field foreground-background segmentation," in *Proc. Telecoms Conf. (ConfTELE)*, Leiria, Portugal, Feb. 2021, pp. 1–6.

[8] S. Wanner and B. Goldluecke, "Variational light field analysis for disparity estimation and super-resolution," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 36, no. 3, pp. 606–619, Mar. 2014.

[9] S. Zhang, H. Sheng, C. Li, J. Zhang, and Z. Xiong, "Robust depth estimation for light field via spinning parallelogram operator," *Comput. Vis. Image Understand.*, vol. 145, pp. 148–159, Apr. 2016.

[10] N. Khan, M. H. Kim, and J. Tompkin, "View-consistent 4D light field depth estimation," in *Proc. 31st Brit. Mach. Vis. Conf. (BMVC)*, 2020.

[11] H. Jeon, J. Park, G. Choe, J. Park, Y. Bok, Y. Tai, and I. S. Kweon, "Accurate depth map estimation from a lenslet light field camera," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Boston, MA, USA, Jun. 2015, pp. 1547–1555.

[12] C. Huang, "Empirical Bayesian light-field stereo matching by robust pseudo random field modeling," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 41, no. 3, pp. 552–565, Mar. 2019.

[13] M. W. Tao, S. Hadap, J. Malik, and R. Ramamoorthi, "Depth from combining defocus and correspondence using light-field cameras," in *Proc. IEEE Int. Conf. Comput. Vis.*, Sydney, NSW, Australia, Dec. 2013, pp. 673–680.

[14] H. Lin, C. Chen, S. B. Kang, and J. Yu, "Depth recovery from light field using focal stack symmetry," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, Santiago, Chile, Dec. 2015, pp. 3451–3459.

[15] J. Y. Lee and R. Park, "Complex-valued disparity: Unified depth model of depth from stereo, depth from focus, and depth from defocus based on the light field gradient," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 43, no. 3, pp. 830–841, Mar. 2021.

[16] 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.

[17] X. Jiang, J. Shi, and C. Guillemot, "A learning based depth estimation framework for 4D densely and sparsely sampled light fields," in *Proc. IEEE Int. Conf. Acoust., Speech Signal Process. (ICASSP)*, Brighton, U.K., May 2019, pp. 2257–2261.

[18] S. Heber and T. Pock, "Convolutional networks for shape from light field," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Las Vegas, NV, USA, Jun. 2016, pp. 3746–3754.

[19] S. Heber, W. Yu, and T. Pock, "Neural EPI-volume networks for shape from light field," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, Venice, Italy, Oct. 2017, pp. 2271–2279.

[20] C. Shin, H. Jeon, Y. Yoon, I. S. Kweon, and S. J. Kim, "EPINET: A fully-convolutional neural network using epipolar geometry for depth from light field images," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Salt Lake City, UT, USA, 2018, pp. 4748–4757.

[21] Y. Li, L. Zhang, Q. Wang, and G. Lafruit, "MaNet: Multi-scale aggregated network for light field depth estimation," in *Proc. IEEE Int. Conf. Acoust., Speech Signal Process. (ICASSP)*, Barcelona, Spain, May 2020, pp. 1998–2002.

[22] Y. Wang, L. Wang, G. Wu, J. Yang, W. An, J. Yu, and Y. Guo, "Disentangling light fields for super-resolution and disparity estimation," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 45, no. 1, pp. 425–443, Jan. 2023.

[23] W. Chao, X. Wang, Y. Wang, L. Chang, and F. Duan, "Learning sub-pixel disparity distribution for light field depth estimation," 2022, *arXiv:2208.09688*.

[24] J. Jin and J. Hou, "Occlusion-aware unsupervised learning of depth from 4-D light fields," *IEEE Trans. Image Process.*, vol. 31, pp. 2216–2228, 2022.

[25] 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.

[26] 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.

[27] 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.

[28] K. Honauer, O. Johannsen, D. Kondermann, and B. Goldluecke, "A dataset and evaluation methodology for depth estimation on 4D light fields," in *Proc. 13th Asian Conf. Comput. Vis. (ACCV)*, Taipei, Taiwan, 2016, pp. 19–34.

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

[30] V. Vaish and A. Adams. (2008). *The (New) Stanford Light Field Archive*. Accessed: Jun. 24, 2020. [Online]. Available: http://lightfield.stanford.edu/acq.html

[31] 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.

[32] E. Ilg, N. Mayer, T. Saikia, M. Keuper, A. Dosovitskiy, and T. Brox, "FlowNet 2.0: Evolution of optical flow estimation with deep networks," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Honolulu, HI, USA, Jul. 2017, pp. 1647–1655.

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 Ph.D. degree with Instituto Universitário de Lisboa (ISCTE-IUL), Portugal. During her degree, she spent one semester as an Exchange Student with the ERASMUS+ Program, Middle East Technical University (METU), Turkey. She completed her professional internship in information science and technology with the IAESTE Program, Multimedia Signal Processing Group, Instituto de Telecomunicações, Portugal, as a Researcher. Currently, she is a Researcher with the Multimedia Signal Processing Group and a member of the IEEE Women in Engineering Society, IEEE Signal Processing Society, and IEEE Young Professionals Group. Her current research interests include immersive visual technologies, such as light field imaging, digital image processing, and computer vision. She acts as a reviewer of the IEEE Access.

CAROLINE CONTI (Member, IEEE) received the B.Sc. degree in electrical engineering from Universidade de São Paulo (USP), Brazil, in 2010, and the Ph.D. degree in information science and technology from Instituto Universitário de Lisboa (ISCTE-IUL), Portugal, in 2017. Currently, she is a Senior Researcher with the Multimedia Signal Processing Group, Instituto de Telecomunicações, and an Assistant Professor with the Information Science and Technology Department, ISCTE-IUL. Her research interests include immersive visual technologies and image and video processing, including light field processing and coding. She has contributed over 25 papers to international journals and conferences in these areas. She also serves as an Associate Editor for IEEE Transactions on Image Processing, has been a Guest Editor for *Signal Processing: Image Communication* (Elsevier) journal, and actively participates as a reviewer for various IEEE and EURASIP journals and conferences.

PAULO NUNES (Member, IEEE) received the Graduate degree in electrical and computers engineering from 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. Currently, he is a Senior Researcher with the Multimedia Signal Processing Group, Instituto de Telecomunicações, Portugal. In addition, he is also an Associate Professor with the Information Science and Technology Department, Instituto Universitário de Lisboa (ISCTE-IUL), Portugal. His current research interests include 2D/3D image and video processing and coding, namely light field image and video processing and coding. He has coordinated and participated in various national and international (EU) funded projects and has acted as a project evaluator for the European Commission. He acts often as a reviewer of various ACM, EURASIP/Elsevier, IEEE, IET, MDPI, SPIE, and Springer conferences and journals and a member of the program and organizing committees of various international conferences. He has contributed more than 70 papers to international journals and conferences in these areas.

LUÍS DUCLA SOARES (Senior Member, IEEE) received the Licenciatura and Ph.D. degrees in electrical and computer engineering from Instituto Superior Técnico (IST), Universidade Técnica de Lisboa, Portugal, in 1996 and 2004, respectively. Currently, he is a Senior Researcher with the Multimedia Signal Processing Group, Instituto de Telecomunicações, Portugal. In addition, 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 and biometric recognition. He has contributed more than 70 papers to international journals and conferences in these areas. In addition, he has participated in the development of the MPEG-4 visual standard and in several national and international projects. He is a member of the Editorial Board of the *EURASIP Journal on Advances in Signal Processing* (Elsevier). In parallel, he acts as a reviewer of several IEEE, IET, and EURASIP journals and conferences.

Chapter 3

ALFO: Adaptive Light Field Over-Segmentation

Journal: IEEE Access
Date of Publication: September 2021
Volume: 9
Pages: 131147-131165
DOI: 10.1109/ACCESS.2021.3114324

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.

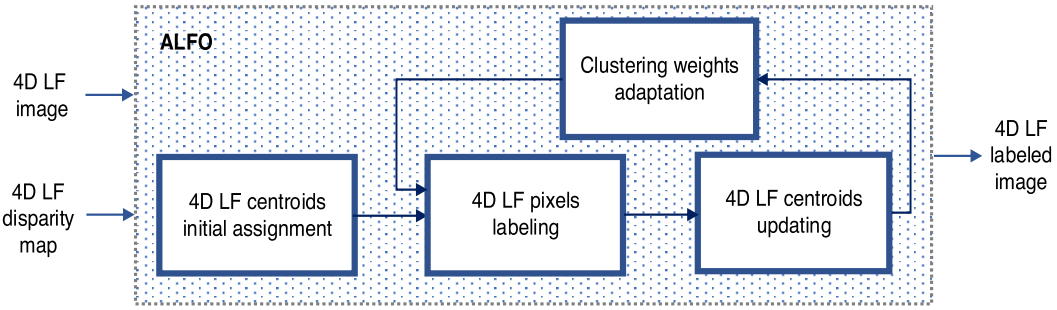


FIGURE 1. Overview of the proposed ALFO method. Given a 4D LF image and the corresponding disparity maps for all views, initial centroids, characterized by distinct features, are assigned in a reference view. Next, the 4D LF superpixel segmentation is achieved by iteratively applying K-means clustering, 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 $c \in \{1,\dots,K\}$
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,

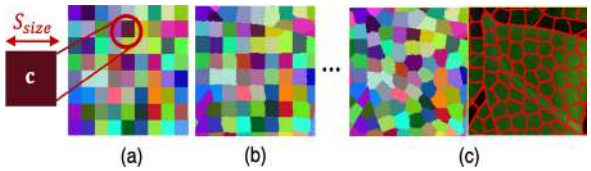


FIGURE 2. Visual representation of the clustering iterations: a) initial square grid in the central view only. Each square represents a superpixel and the center point of each square represents its centroid. In (a), for illustration, all pixels are labeled, however, initially, only centroids have labels;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

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

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 $(d_{hor,\mathbf{c}}^{ref\to(u,v)}, d_{ver,\mathbf{c}}^{ref\to(u,v)})$ are the horizontal and vertical disparities from the reference view $ref = (u_{ref}, v_{ref})$ 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]:

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

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):

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

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

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

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

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

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

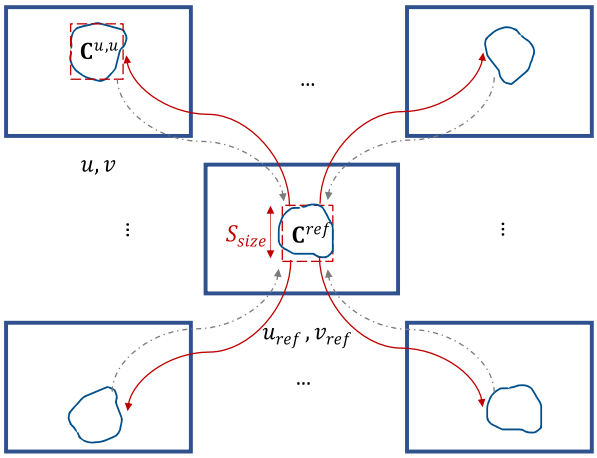


FIGURE 3. Assuming all centroids in the reference view exist in all other views, the projection of a centroid from the reference view into other views is illustrated by the solid red arrows. Similarly, back-projection of all pixels that belong to a superpixel from all other views into the reference view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

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

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):

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

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):

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

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):

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

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):

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

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):

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

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:

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

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 $L(\boldsymbol{p}) = 0$ for each pixel;
Initialize pixel distance $D(\boldsymbol{p}) = \infty$ 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, $\Omega_{\mathbf{c}}$, around the projected $\mathbf{c}$;
      for *each pixel* $p \in \Omega_{\mathbf{c}}$ **do**
        Compute features distance, $D_w(\mathbf{p}, \mathbf{c})$, using (9);
        if $D_w(\boldsymbol{p}, \boldsymbol{c}) < \mathbf{D}(\boldsymbol{p})$ **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.

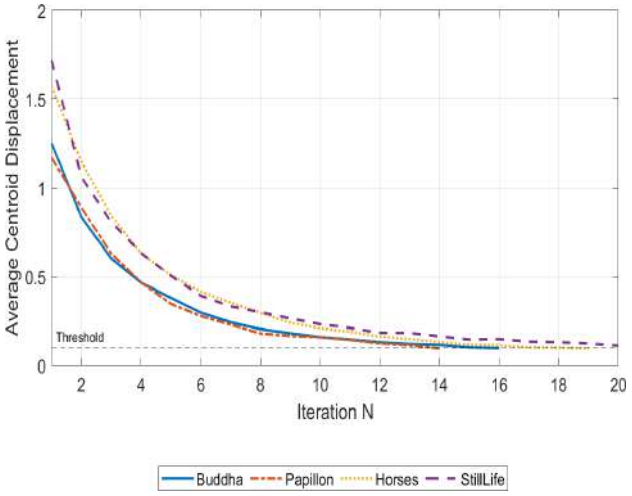


FIGURE 4. Average displacement of centroid spatial coordinates, D_{avg}, in pixels, along the number of iterations. $S_{size} = 20$.

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]:

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

  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.

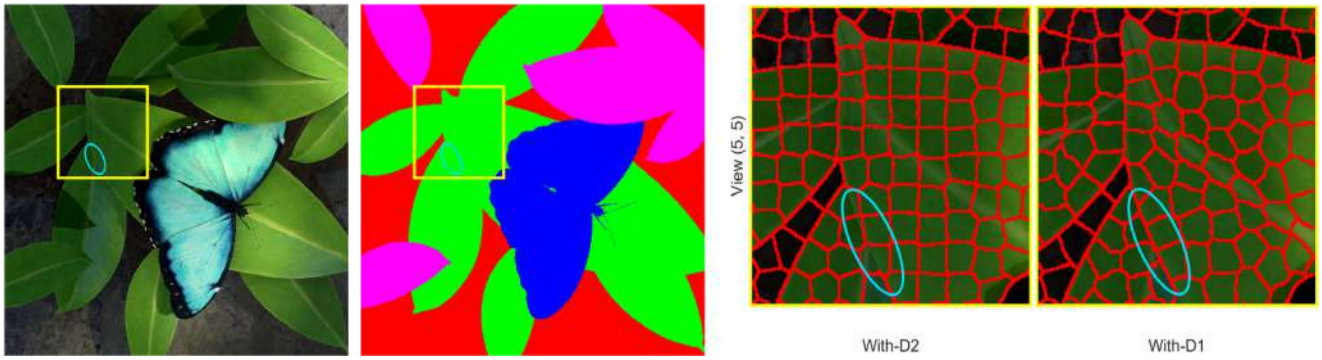


FIGURE 5. Visual results for Papillon test image, with and without using the squared disparity distance in the clustering weighted distance, (With-D2), (With-D1), respectively. Portions of the central view (5, 5) are selected and highlighted on both the test images and the corresponding ground truth 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:

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

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.

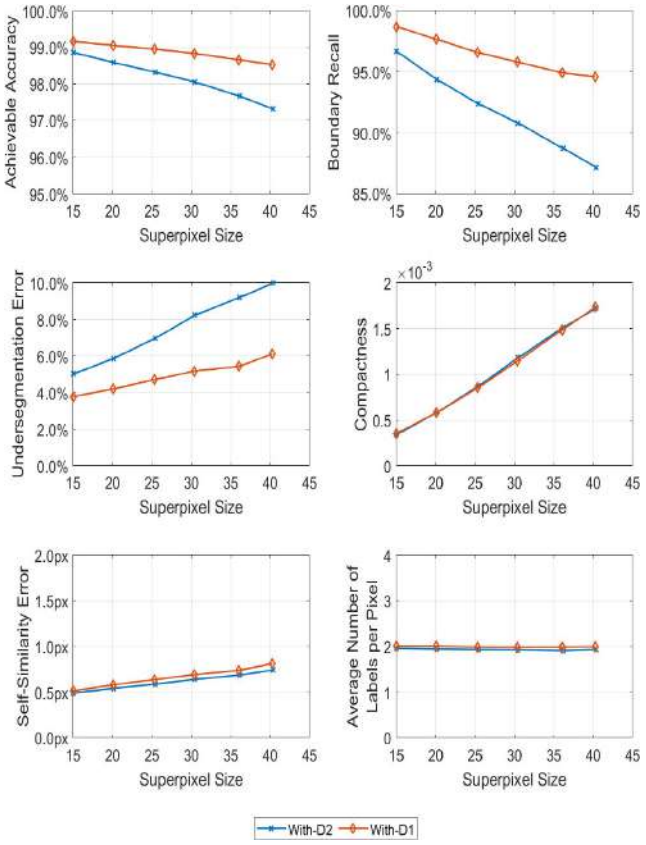


FIGURE 6. Average quantitative evaluation on all LF images of the HCI 4D LF dataset, (With-D2), (With-D1) indicate with and without using the squared disparity distance in the clustering weighted distance, respectively.

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

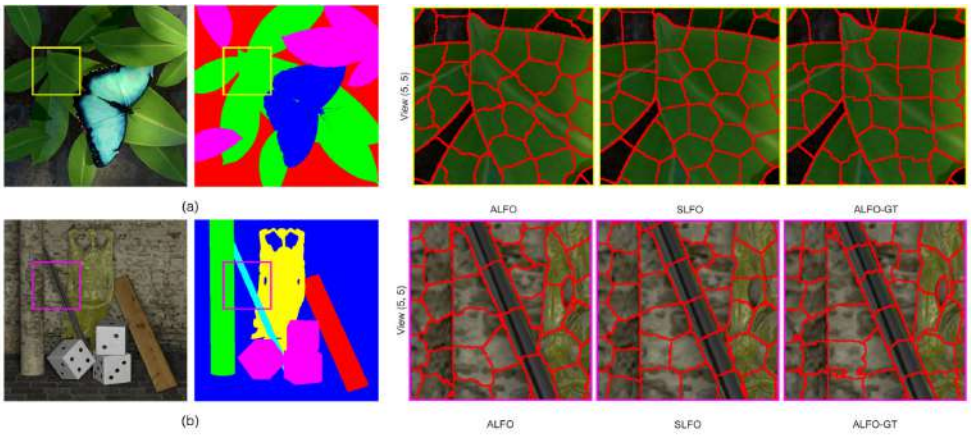


FIGURE 7. Visual results for two test images of the HCI 4D LF dataset for different test conditions of the proposed ALFO method, namely SLFO and ALFO-GT. Portions of the central view (5, 5) are selected and highlighted on both the test images and the corresponding ground truth label images. Adaptive 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

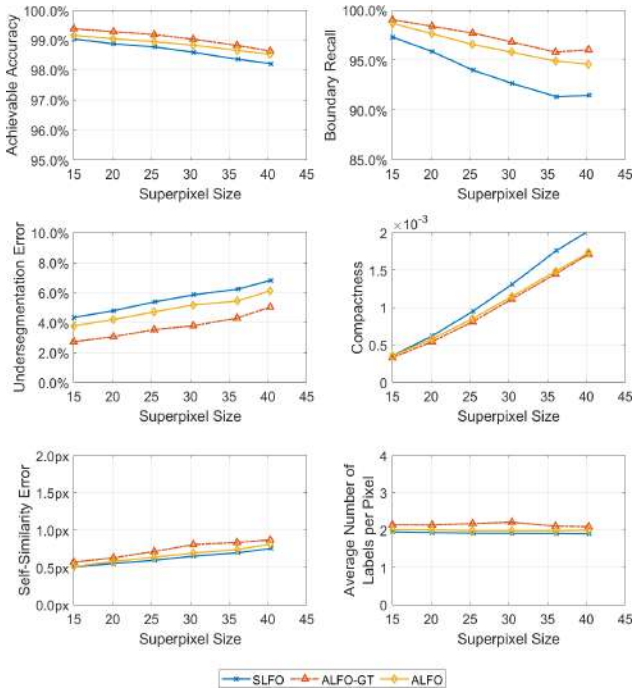


FIGURE 8. Average quantitative evaluation on all LF images of the HCI 4D LF dataset for different test conditions of the proposed method, namely SLFO, ALFO-GT and the proposed ALFO method. Adaptive clustering weights and good disparity maps can improve the segmentation performance.

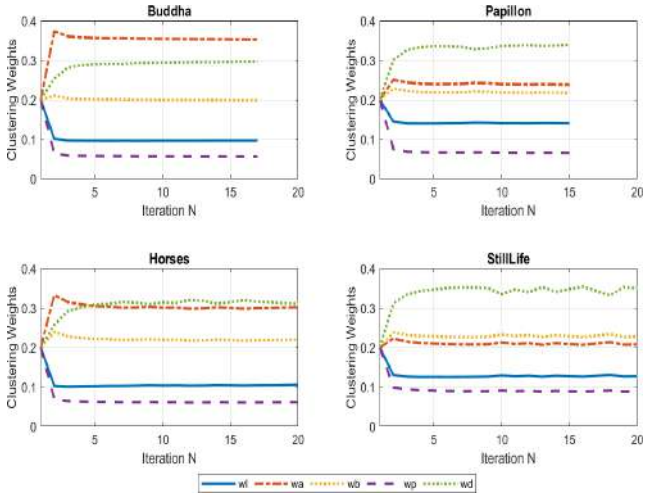


FIGURE 9. Clustering weights adaptation along the number of iterations for different test images. The included weights are w_l, w_a, w_b for color channels, w_p for relative position and w_d for the disparity. $S_{size} = 20$.

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

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.

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.

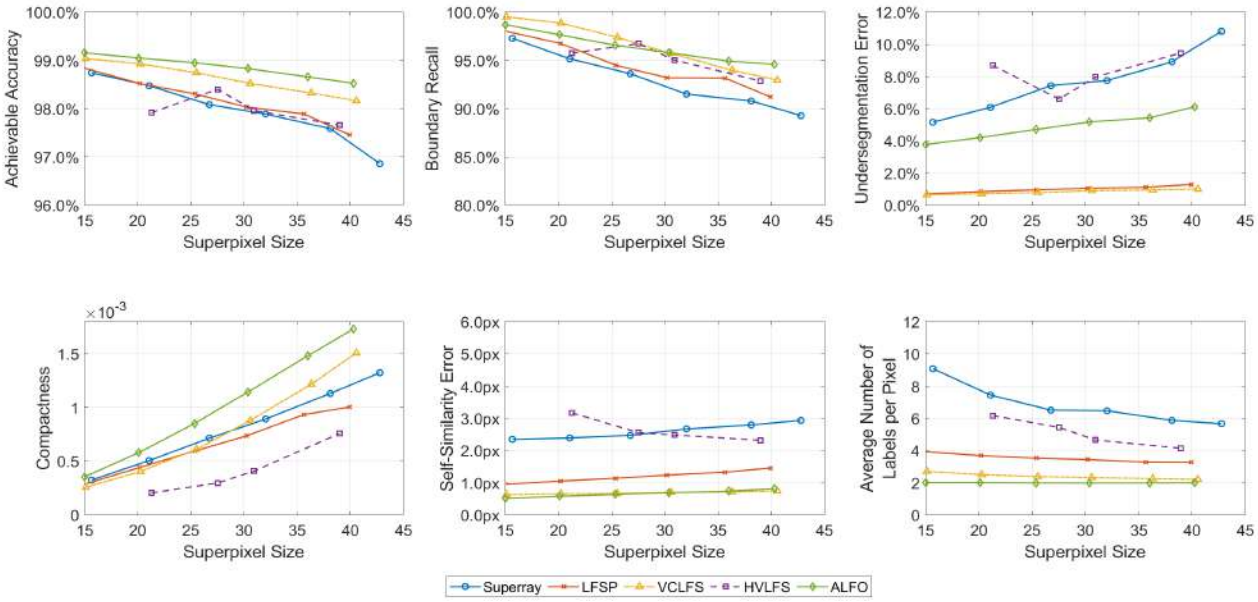


FIGURE 10. Average quantitative evaluation on all LF images of the HCI 4D LF dataset for different 4D LF superpixel segmentation methods.

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

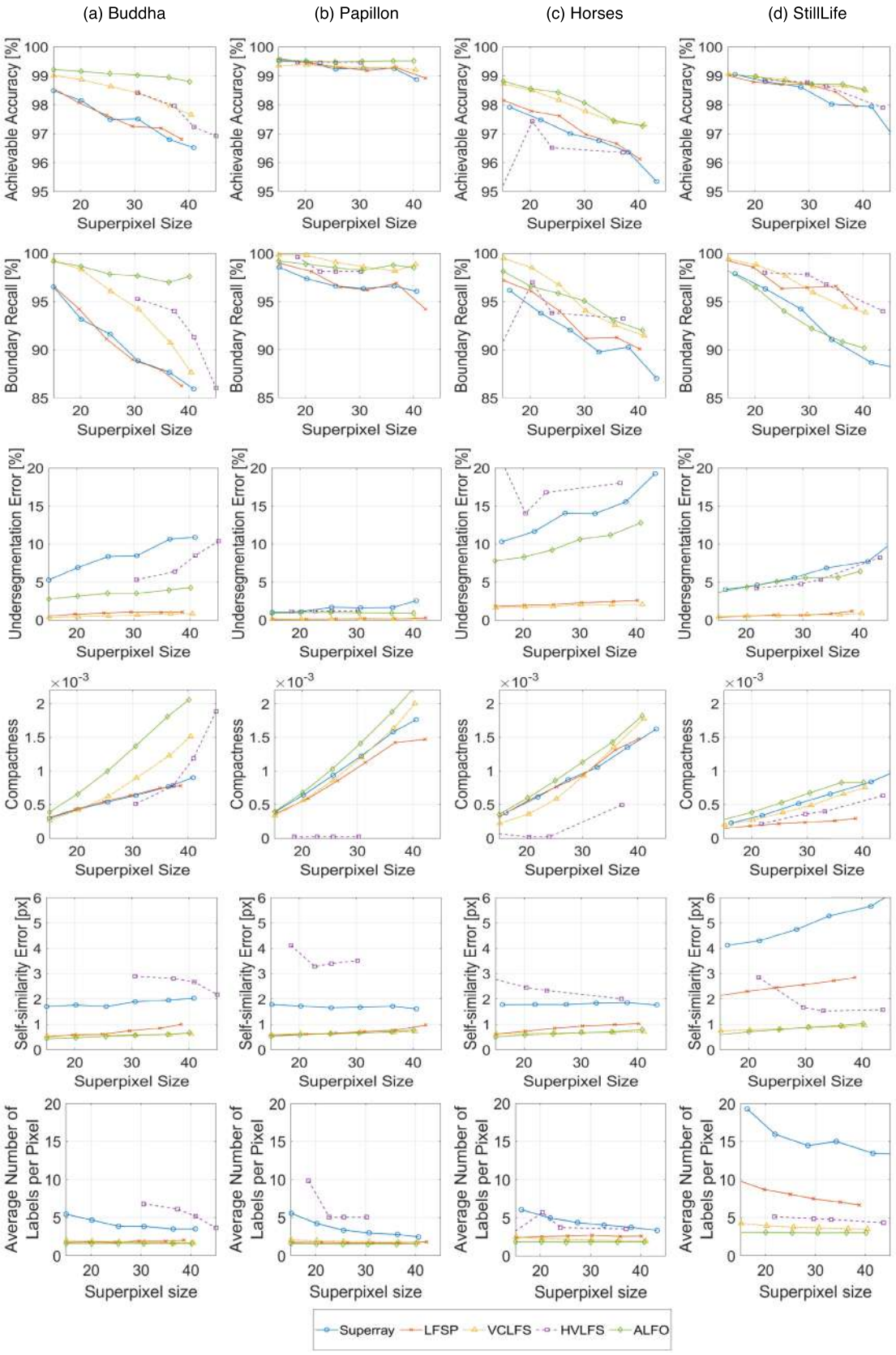


FIGURE 11. Per-image quantitative evaluation on the HCI 4D LF dataset for different 4D LF superpixel segmentation methods.

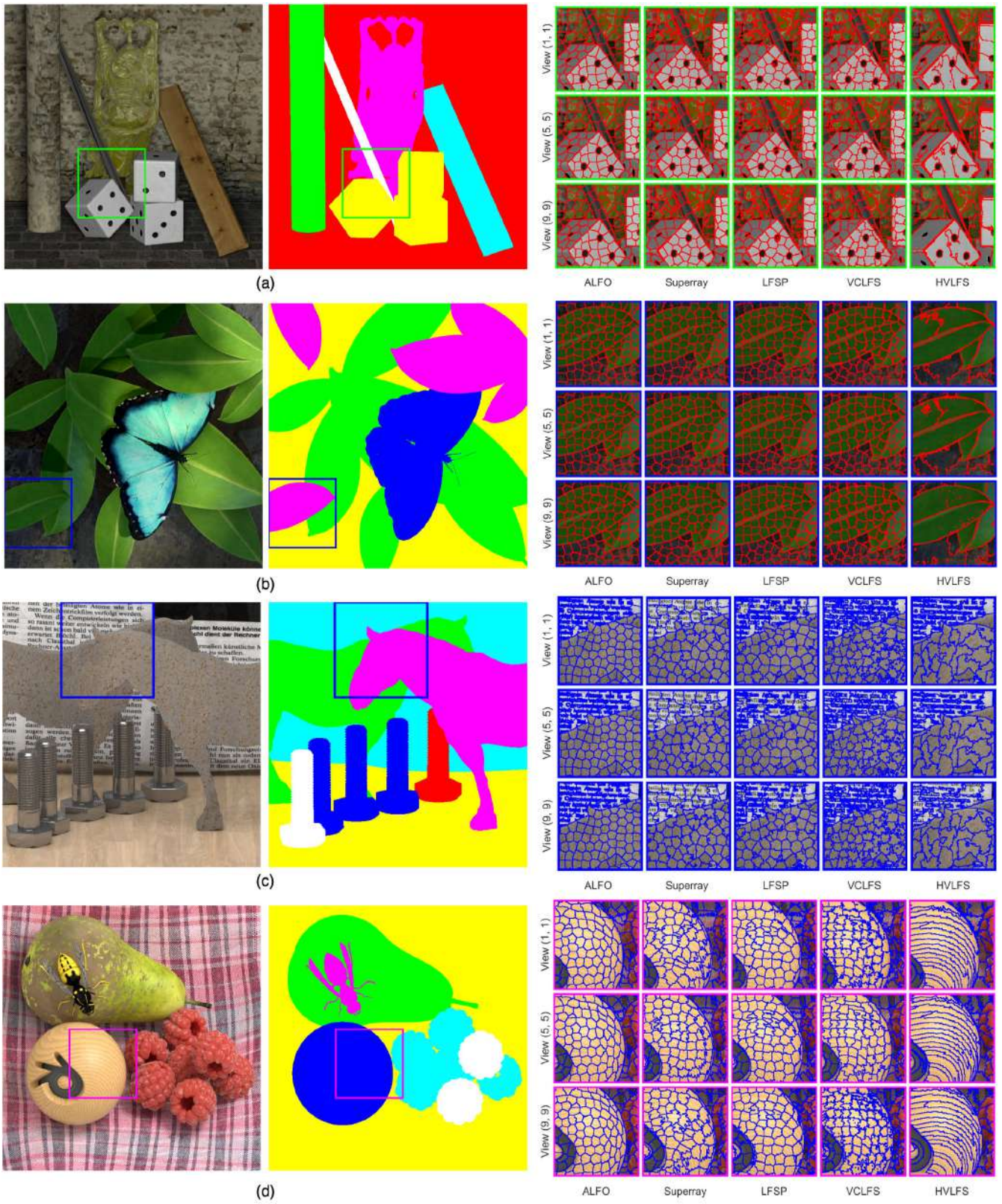


FIGURE 12. Visual results to evaluate accuracy, compactness and consistency across views for the proposed ALFO, Superray, LFSP, VCLFS and HVLFS methods on the HCI 4D LF dataset. Challenging regions (highlighted on both the test images and the corresponding ground truth label images) are selected to show 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. $S_{size} = 20$.

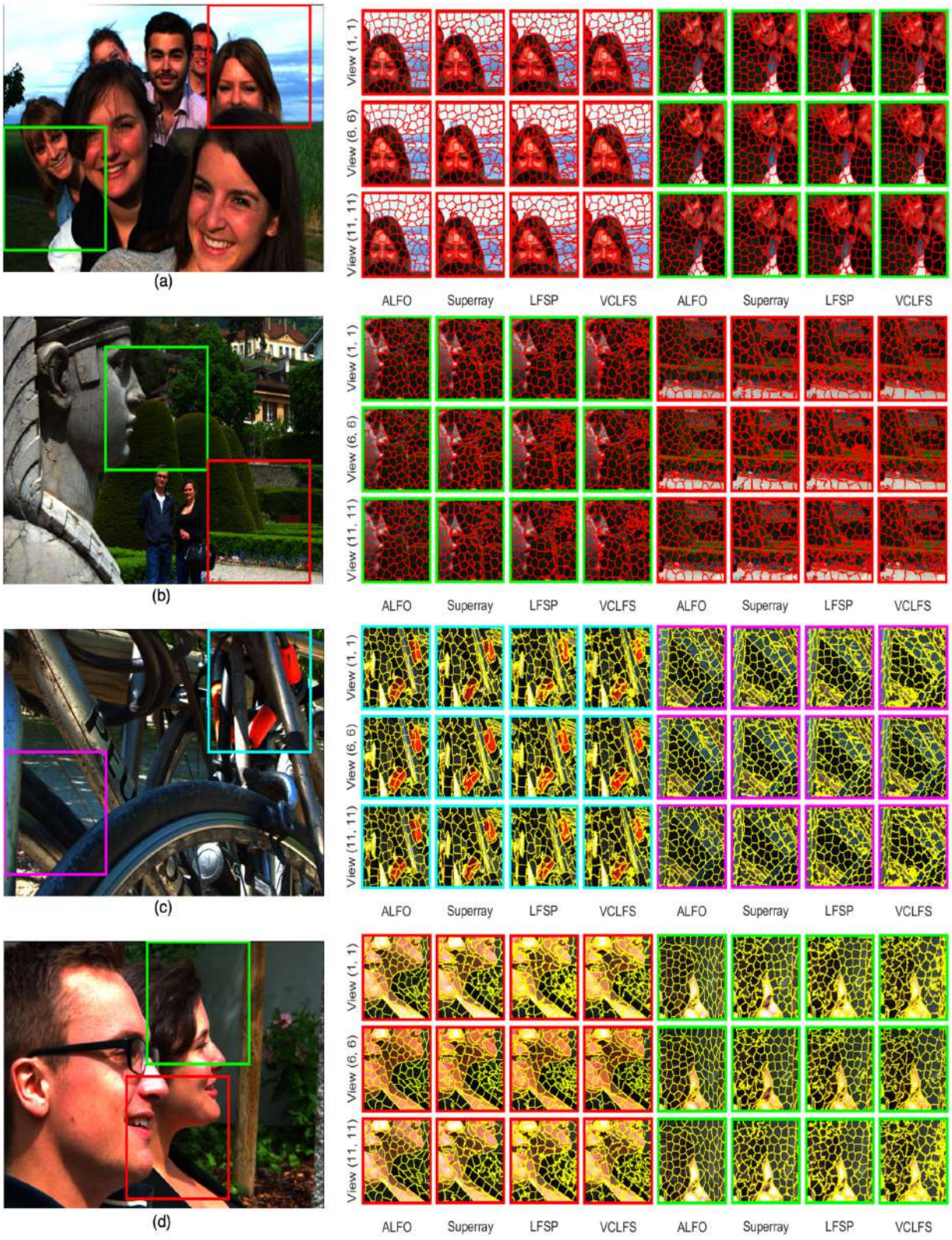


FIGURE 13. Visual results to evaluate accuracy, compactness and consistency across views for the proposed ALFO, Superray, LFSP and VCLFS methods on the MMSPG LF dataset. In real 4D LF images, the noise existence and non-even lighting generally can affect the segmentation accuracy or generate flickering 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.

Chapter 4

Hyperpixels: Flexible 4D Over-Segmentation for Dense and Sparse Light Fields

Journal: IEEE Transactions on Image Processing
Date of Publication: July 2023
Volume: 32
Pages: 3790-3805
DOI: 10.1109/TIP.2023.3290523

***This paper is presented as well at IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP), April 2024, Seoul, Korea, DOI: 10.60864/fzyz-h513**

Hyperpixels: Flexible 4D Over-Segmentation for Dense and Sparse Light Fields

Maryam Hamad, *Graduate Student Member, IEEE*, Caroline Conti, *Member, IEEE*, Paulo Nunes, *Member, IEEE*, and Luís Ducla Soares, *Senior Member, IEEE*

***Abstract*— 4D Light Field (LF) imaging, since it conveys both spatial and angular scene information, can facilitate computer vision tasks and generate immersive experiences for end-users. A key challenge in 4D LF imaging is to flexibly and adaptively represent the included spatio-angular information to facilitate subsequent computer vision applications. Recently, image over-segmentation into homogenous regions with perceptually meaningful information has been exploited to represent 4D LFs. However, existing methods assume densely sampled LFs and do not adequately deal with sparse LFs with large occlusions. Furthermore, the spatio-angular LF cues are not fully exploited in the existing methods. In this paper, the concept of hyperpixels is defined and a flexible, automatic, and adaptive representation for both dense and sparse 4D LFs is proposed. Initially, disparity maps are estimated for all views to enhance over-segmentation accuracy and consistency. Afterwards, a modified weighted *K*-means clustering using robust spatio-angular features is performed in 4D Euclidean space. Experimental results on several dense and sparse 4D LF datasets show competitive and outperforming performance in terms of over-segmentation accuracy, shape regularity and view consistency against state-of-the-art methods.**

***Index Terms*— Light field over-segmentation, 4D *K*-means clustering, light field representation, superpixel, supervoxel.**

I. Introduction

THE required resolution (e.g., spatial, angular and temporal) and degrees of freedom in multimedia applications are growing rapidly. Consequently, the associated computational complexity for processing the data is also increasing significantly. 4D Light Fields (LFs) that capture the same scene from different perspectives are a clear example of what this trend is leading to [1]. To efficiently process the huge amount of data, one possible approach is to reduce the number of data units that need to be processed. This can be achieved by grouping the locally homogenous data units according to some criteria into larger ones. This approach is known as "image over-segmentation". A recent trend in computer vision is to process 2D images and 3D volumes at a higher-level representation instead of at the pixel-level representation [2]. As an example, image over-segmentation can be used as a pre-processing step in image compression [3], [4], object tracking [5], object segmentation [6], [7], 3D semantic segmentation [8] and saliency detection [9]. Considering that image over-segmentation can be applied to 2D images and 3D volumes to facilitate subsequent applications, applying a similar approach to 4D LFs would also make sense.

4D LFs indirectly describe the distribution of light rays in free space by capturing the same scene from several points of view [1], [10]. Depending on the LF capturing approach, dense or sparse 4D LFs can be generated [1]. In dense LFs, most of the objects exist in all LF views and, therefore, LF processing or editing can be done on only a single LF view, or a small subset of LF views, and then propagated into all other LF views using, for example, LF view warping. In sparse LFs, however, such possibility is limited by largely occluded regions or regions that only appear in some LF views due to the viewing angle. To handle these specific issues of sparse LFs, all objects that appear in any LF view must be considered, and an adequate propagation method must be used to ensure accurate and angularly consistent LF processing or editing. In both cases, due to the existing similarities within LF views, LF over-segmentation can be exploited to group data units within and across LF views. Therefore, a significant reduction in the number of data units to be processed can be achieved to facilitate subsequent tasks [7], [11], [12], [13]. 4D LF over-segmentation should aim at not only spatial accuracy (i.e., adhering well to object boundaries and separating regions correctly), but also angular consistency (i.e., segmented regions not changing abruptly when the viewpoint changes). Currently, only a few methods for 4D LF over-segmentation are available in the literature. These methods can be classified as being either clustering-based methods [11], [14], [15], [16], [17] or graph-based methods [18], depending on the used approach. The clustering-based approach is adopted in this paper, since it is widely used due to the superior results in terms of accuracy and also due to the reduced computational complexity and memory usage, when compared to graph-based ones [2], [19]. Although the available methods that tackled 4D LF over-segmentation challenges have significantly improved over-segmentation angular consistency (compared to simply applying a 2D method to each view

Manuscript received 14 February 2022; revised 31 March 2023; accepted 13 June 2023. Date of publication 5 July 2023; date of current version 11 July 2023. This work was supported by the Fundação para a Ciência e a Tecnologia (FCT)/ Ministério da Ciência, Tecnologia e Ensino Superior (MCTES) through national funds under Project UIDB/50008/2020 and Project PTDC/EEI-COM/7096/2020. The associate editor coordinating the review of this manuscript and approving it for publication was Prof. Yulan Guo. *(Corresponding author: Maryam Hamad.)*

The authors are with Instituto de Telecomunicações, Instituto Universitário de Lisboa (ISCTE-IUL), 1649-026 Lisbon, Portugal (e-mail: maryam.hamad@lx.it.pt; caroline.conti@lx.it.pt; paulo.nunes@lx.it.pt; lds@lx.it.pt).

This article has supplementary downloadable material available at https://doi.org/10.1109/TIP.2023.3290523, provided by the authors.

Digital Object Identifier 10.1109/TIP.2023.3290523

independently), remaining limitations still need to be further investigated.

Firstly, existing methods consider dense LFs (i.e., captured with narrow baselines between views) and do not adequately deal with sparse LFs with large occlusions (i.e., captured with large baselines between views). For example, one reference view (e.g., the central view) or the structure of the central Epipolar Plane Image (EPI) (i.e., the unique 2D spatio-angular slice of the LF typically containing a regular structure with several oriented lines [20]) is used to perform 2D over-segmentation. After that, the obtained segments are propagated to other LF views. For this, it is assumed that each 2D segment in the central view should have a corresponding one in all other LF views (i.e., "full-sliced" property). This assumption, however, may not always hold, notably for sparse LFs. In the sparse LF case, some objects may not exist in all LF views, either because they are occluded in some LF views by foreground objects or because they fall outside the viewing angle of those views.

Secondly, the spatio-angular LF cues, including depth or disparity information (i.e., the displacement of a point between different views, which is inversely proportional to the depth), and 4D spatio-angular coordinates are not fully exploited in most existing methods. The used disparity information in some existing methods is either estimated for some pixels only (e.g., the clustering centroids) or for all pixels in one reference view only (e.g., the central view) [11], [15]. Moreover, disparity information in some methods is used to enforce a view consistent projection for the clustering centroids, but not as a discriminative feature to guide the over-segmentation (for instance, when color information is insufficient to separate different regions [17]). Additionally, all available clustering-based methods are still not 4D in nature, meaning that the clustering is applied using 2D Euclidean space without considering the angular dimensions, and the centroids are fixed in one angular location. Lastly, none of the previous methods (except in [17]) support adaptive clustering.

In this paper, a novel clustering-based 4D LF over-segmentation method that tackles these limitations is proposed. The contribution of this paper is four-fold:

- **The definition of 4D hyperpixels for dense and sparse LFs**– The "hyperpixels" definition is provided to have an entity that adequately reflects the high dimensional nature of the basic element of 4D LF over-segmentation, supporting flexible clustering/grouping criteria for both dense and sparse LFs. The provided definition extends the existing definitions in [11] and [15] as detailed in Section III.
- **Flexible, adaptive and consistent 4D over-segmentation method for dense and sparse LFs**– In this paper, LF over-segmentation is applied using a modified K-means clustering in the 4D hypercubic domain that is adapted to LF content and fully exploits the spatio-angular cues. As such, it is the only over-segmentation method for LFs that is truly 4D in nature. The differences between the proposed hyperpixel over-segmentation method and other methods is detailed in Section IV. Experimental results, including dynamic results in the supplemental materials, show superior performance when compared to existing methods.
- **A 4D LF dataset of sparse LFs with a large absolute disparity range**– To validate our proposed method for sparse 4D LFs quantitively, a dataset of 4D LFs including non-Lambertian objects and complex texture regions that mimic real images is generated. This is the first sparse 4D LFs dataset that includes ground truth segmentation label images, disparity, and depth maps for all LF views. It is publicly available and can be used to qualitatively and quantitively evaluate 4D LFs for several LF applications.
- **Labeling–LF Angular Consistency (LLFAC) metric**– Existing LF view consistency metrics discard the large occlusions in off-central views when projected into the central view and, hence, may not fairly evaluate the view consistency in sparse LFs. In this paper, we highlight the importance of having metrics for sparse LFs that can consider local angular consistency. Therefore, we adapted the recently proposed metric that is applied for LF style transfer applications [21] to evaluate labeling LF angular consistency for dense and sparse LFs.

The remainder of the paper is organized as follows. Section II briefly reviews the related work on LF over-segmentation. Section III introduces the concept of hyperpixels in 4D space and explains the differences with respect to previous definitions. Section IV describes the proposed method in detail, while in Section V its performance is evaluated through a series of experiments. Finally, Section VI concludes the paper with final remarks and proposes directions for future work.

II. Related Work

Image over-segmentation aims at providing a more meaningful representation of an image and can reduce the number of processing data units. Ren and Malik [22] first defined a group of locally coherent pixels that share the same visual properties as "superpixels". Subsequently, this concept has inspired many researchers to propose various 2D over-segmentation methods, of which a comprehensive review can be found in [2]. More recently, deep learning was exploited in 2D image over-segmentation, leading to a further improvement in accuracy [23], [24]. However, applying 2D over-segmentation methods to each LF view independently will not ensure LF angular consistency, which is crucial for many applications. The superpixel concept has also been extended to consider 3D volumes [25], videos [26] and higher-dimensional visual data, such as 4D LFs, where over-segmentation angular consistency is particularly important.

In this section, the few available 4D LF over-segmentation methods are briefly reviewed. Current 4D LF over-segmentation methods can be classified as clustering-based or graph-based, depending on the approach used to divide 4D LFs into homogeneous regions.

A. Clustering-Based 4D LF Over-Segmentation

In this class, 4D LFs are divided into a certain number of homogeneous clusters of pixels with similar sizes using the

K-means clustering technique. Currently, all available methods in this category start the clustering process by initializing the centroids only in the central view of the LF.

Hog et al. [11] proposed a fast method that groups light rays of similar color in an LF into what they defined as "superrays" using 2D K-means clustering. The angular consistency is enforced by projecting the superrays in the central view into all other views and vice versa, using the disparity values of their centroids. Notice that the disparity values are estimated only for the centroids in the central view in the initial position of the centroids to apply the projection step and are not included as a clustering feature. Therefore, a cleaning step is needed to correct wrongly labeled or unlabeled pixels due to inaccurate projection or clustering, especially in largely occluded regions. Later, the authors extended their work to handle LF videos by also considering the temporal dimension [27].

Zhu et al. [15] defined the concept of 4D LF SuperPixel (LFSP) and a metric for evaluating LFSP angular consistency (i.e., the self-similarity metric). The method proposed in [15] to generate LFSPs relies on segmenting the central view firstly with a 2D K-means clustering algorithm, assisted by the disparity feature only for the central view. After that, superpixels are projected to other views using the centroids disparity values. Finally, an optimization stage is needed to ensure the EPI space regularity. In this work, the "full-sliced" property is assumed, which can represent a significant limitation for sparse LFs.

Khan et al. [16] proposed a novel View-Consistent Light Field Superpixel (VCLFS) segmentation. Initially, the over-segmentation is applied in the EPI space for the central horizontal and central vertical EPIs independently, by considering that each pair of lines defines a 2D segment. After that, a 2D K-means clustering is applied after combining the horizontal and vertical EPIs into the central view. Labels are then propagated to all off-central LF views using per-pixel disparity. Although the disparity for all views is used during the clustering, relying on EPI regularity can limit the VCLFS method performance for sparse LFs (e.g., due to their irregular EPI structure).

Recently, Hamad et al. [17] proposed an adaptive LF Over-segmentation (ALFO) method based on modified 2D K-means clustering. In the ALFO method, the weights applied to the different features for clustering are adjusted adaptively based on the image content. Hence, the balance between regularity, compactness, and angular consistency is improved. In this method, per-pixel disparity is required as input and exploited during the clustering. Although ALFO has shown outperforming performance, it still does not fully exploit the spatio-angular cues, this fact will be further discussed in Section IV-F. Moreover, as in the previous methods, only the central view is used to initialize the centroids, which is not adequate for sparse LFs and largely occluded regions.

B. Graph-Based 4D LF Over-Segmentation

In this class, LF over-segmentation is considered as a graph-partitioning task. More precisely, an undirected graph is created from a 4D LF by considering every single pixel in a 4D LF as a graph node. Afterwards, according to the edge weights between adjacent nodes, the graph is cut into sub-graphs with each sub-graph representing a 4D segment. Generally, applying graph optimization on a huge number of pixels, may require a long execution time and extensive consumption of resources.

Li et al. [18] proposed a Hierarchical and View-invariant LF Segmentation (HVLFS) method. By creating a weighted undirected 4D graph from a 4D LF, the over-segmentation is achieved by maximizing the graph entropy in the 4D LF domain. In this method, different features are used to guide the over-segmentation, such as color, depth and texture. The entropy rate for the over-segmentation is measured in the EPI space to ensure angular consistency. This method generates subgraphs with different sizes according to the user input. However, some limitations remain regarding the need for proper normalization of the used weights for optimization to robustly fit different LF datasets. Moreover, since angular consistency is handled by tracking the EPI structure, the method has been shown to fail when applied to sparse 4D LFs [18].

III. Hyperpixels Definition

A pixel (short for "picture element") is the fundamental unit in 2D images. Similarly, the fundamental unit of 3D volumes is called a voxel (short for "volume element"). Given the fact that these low-level representations do not necessarily have a perceptual meaning [22], a more compact and natural representation is desired. Therefore, locally coherent pixels/voxels in 2D/3D space can be grouped into superpixels /supervoxels [25], respectively, according to some criteria. The main objective is to provide a more meaningful representation and to reduce the number of processing data units. Recently, a froxel was defined to describe an element of a frustum-aligned voxel grid, by using depth and camera-setup-dependent discretization of the view frustum [28].

For 4D LFs, the concepts of superray [11] and LFSP [15] were proposed. These concepts, however, still have some limitations that prevent them from being truly analogous to the superpixel and supervoxel ideas but extended for 4D LFs.

In this paper, we try to overcome such limitations by introducing the concept of "hyperpixel", simply defined as "a group of similar pixels in the discrete 4D LF space". The criteria used to define what are similar pixels will depend on the specifics of the over-segmentation method adopted. The differences with respect to superrays and LFSP are described as follows.

The authors in [11] defined superrays as "*groups of rays of similar color coming from the same scene area*". This definition implies a representation in the continuous 3D scene space, although the authors used it interchangeably to refer to its corresponding projection in the discrete 4D LF space (x, y, u, v). Moreover, in this definition, the authors impose the following constraint on the grouping of rays: the rays in each superray must have a similar color. The goal of our proposal is to have an entity defined purely in the discrete 4D LF space without imposing any constraint on the similarity criteria used for grouping. With the proposed definition of hyperpixels,

Fig. 1. Examples of regions only visible in some views. The fire extinguisher is occluded by the blue car in view (5, 9). The blue car is not visible in view (5, 1) because it is outside the viewing angle of this view. This scene is one of the generated sparse 4D LFs in our dataset.

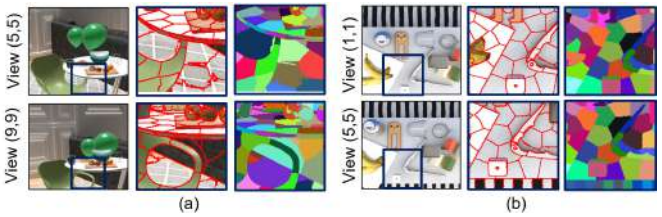


Fig. 2. Visualization of non-existent or occluded regions in the central view, i.e., view (5, 5), that are visible in other LF views and vice versa. a) The part of the sofa that can be seen through the hole of the chair armrest in view (9, 9) is occluded in view (5, 5) ; b) The bottom part of the black and white carpet appears in view (5, 5) but is not visible in view (1, 1) because it falls outside the viewing angle of this view. These scenes are from our sparse 4D LF dataset.

pixel grouping can be performed using a variety of pixel features (e.g., texture, depth, 4D spatial-angular coordinates, etc.). Obviously, the pixel grouping can still be performed using only the color feature, as is the case of superrays. The choice of grouping criteria to be used depends on the specifics of the over-segmentation method adopted.

According to [15], "*LFSP is a light ray set which contains all rays emitted from a proximate, similar and continuous surface in 3D space*". This definition also implies a representation in the continuous 3D scene space, although the authors of [15] also used it interchangeably to refer to its corresponding projection in the discrete 4D LF space (x, y, u, v). Moreover, in this definition, the authors impose the following constraint on LFSPs: "*there are 2D slices of LFSP in all views of light field in free space (i.e., without occlusion)*". On the other hand, hyperpixels are not required to have 2D slices in all LF views, even for objects in free space (i.e., without occlusion). This is particularly important when considering sparse LFs, where it is possible that some objects in free space are only visible in some views and large occlusions can exist (see Fig. 1). Obviously, our definition would also support the case in which a given object in free space is visible in all LF views; in that situation, a 2D slice would exist in all views, as in LFSP. In Fig. 2, an example is shown of how hyperpixels can have slices in some views and not be present in other views if no corresponding pixels exist in those views.

To sum up, we consider that the hyperpixel concept reflects adequately the high dimensional nature of the basic element of 4D LF over-segmentation and it is sufficiently generic and flexible to comprise the 4D projections of both existing superrays and LFSPs definitions.

IV. Proposed 4D Light Field Over-Segmentation

This paper proposes a flexible, adaptive, and view-consistent 4D over-segmentation method for dense and sparse static LFs. According to the hyperpixel definition, our proposed LF over-segmentation method aims at grouping similar pixels in 4D space into hyperpixels. For grouping, several features are considered (i.e., 4D position, color and disparity values). To achieve that, K-means clustering is applied in 4D space. In summary, given a 4D LF scene, disparity maps for all LF views and the hyperpixel size, the proposed method undergoes four main steps (see Fig. 3), where each step is detailed in the following subsections:

1. Initial clustering centroids (i.e., the hyperpixel center of mass in 4D space) are first selected by considering the central view and largely occluded regions from other views. Each centroid is characterized by several features.
2. K-means clustering is applied in 4D LF space and all pixels are labeled iteratively to minimize the within-hyperpixel variance.
3. Centroids color, 4D position and disparity features are adjusted at each iteration during the clustering.
4. Clustering weights are adapted after each iteration.

Steps 2, 3 and 4 are repeated until convergence is reached.

In this paper, we assume a regular arrangement of cameras with a parallel optical axis and uniform camera baseline and focal length. However, the proposed method can also be extended and applied to other camera arrangements by adjusting the used equations accordingly. The main notations used in this paper are listed in TABLE I.

TABLE I
Main Notations Used in This Paper

Symbol	Definition
$I(x, y, u, v)$	A static 4D light field with x, y spatial coordinates and u, v angular coordinates
K	Number of hyperpixels
H_{size}	Grid step size (a.k.a., hyperpixel size)
Ω_i	Searching window centered at centroid c, where $i \in \{1, \dots, K\}$
$\lvert A \rvert$	Cardinality of set A
$\mathbf{p}$	A pixel in 4D space with $(x_\mathbf{p}, y_\mathbf{p}, u_\mathbf{p}, v_\mathbf{p})$ coordinates
$\mathbf{c}$	A centroid in 4D space with $(x_\mathbf{c}, y_\mathbf{c}, u_\mathbf{c}, v_\mathbf{c})$ coordinates before being projected into other views
$\mathbf{c}'$	A projected centroid in 4D space into (u', v') 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 $\mathbf{p}$ from view (u, v) to view (u', v')
H_i	A hyperpixel represented by a centroid with index i
$D_f(\mathbf{p}, \mathbf{c})$	Distance between pixel $\mathbf{p}$ and centroid $\mathbf{c}$ according to feature f, $f \in \{p, l, a, b, d\}$
WV_f	Within-cluster variance of feature f
w_f	Clustering feature weight of feature f
D_{GT}	Ground truth disparity maps
L_{GT}	Ground truth segmentation label images

A. Occlusion-Aware Centroids Initialization

The first step in the proposed hyperpixels over-segmentation method is to select initial centroids to guide the 4D clustering process. Different from other available clustering-based LF over-segmentation methods, where the centroids are initialized in a pre-defined reference view (e.g., the central view), the proposed method enables occlusion-aware centroids initialization. Initializing centroids only in the central view

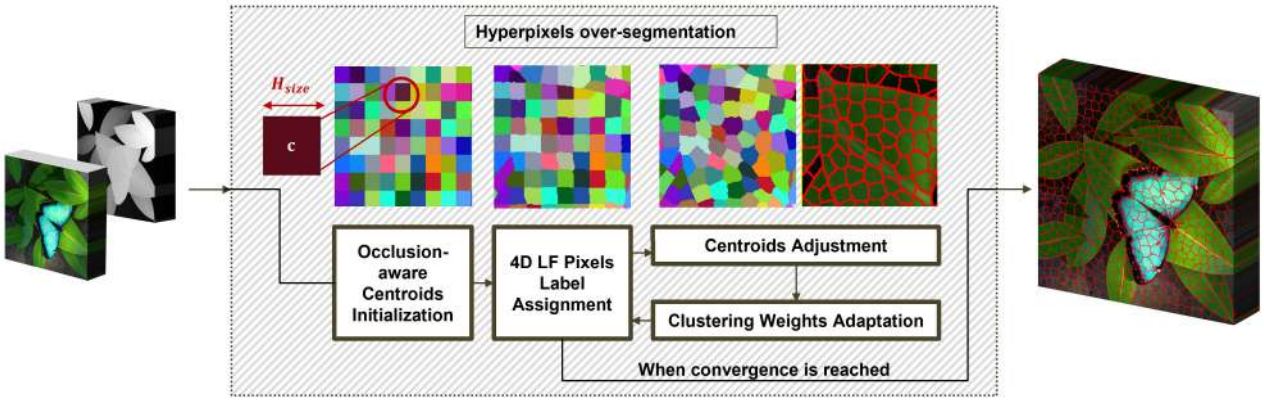


Fig. 3. The main steps of the proposed 4D LF over-segmentation method. Given a 4D LF and the corresponding disparity maps for all views, initial centroids characterized by distinct features are assigned in the reference view/views. Next, hyperpixels are generated by iteratively applying 4D K-means clustering, including pixel labeling, centroids adjustment and clustering weights adaptation, until convergence is reached.

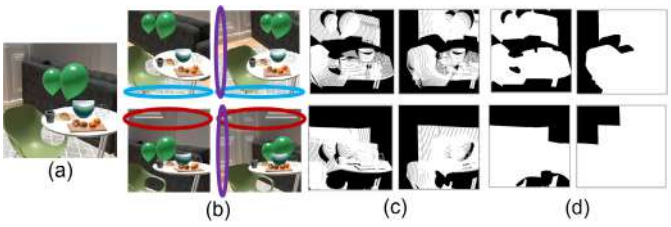


Fig. 4. Example of occluded regions in sparsely sampled LFs. a) The central view; b) 4 reference corner views; c) Occluded regions (black regions) in each view; d) Visibility masks for corner views after redundancy removal. Only the central view and the black regions in the corner views as in (d) will be used to initialize unique centroids to consider the largely occluded regions.

may generate inaccurate over-segmentation for occluded or non-existent regions in the reference view due to different view perspectives; this can be critical due to largely occluded regions in sparse LFs. Therefore, to handle this problem, the four corner views are considered along with the central view for centroid initialization. These extreme corner views are selected since they typically contain all LF information.

To detect the small color differences, before initializing the centroids, the input LF views are converted to the CIELAB color space, which is widely used for image segmentation since it mimics human visual perception. To avoid biased clustering, the LF views and the disparity maps are normalized according to the min-max normalization method as in [17]. Given the normalized inputs, the centroids are initially distributed in the central view over a uniform 2D square grid with step size, H_{size} (a.k.a. hyperpixel size). Afterwards, to detect the occluded or non-existent regions in the central view that are visible in any corner view, the central view is warped to the corner views by using its disparity map. All the occluded regions in each corner view are represented by a binary visibility mask where the occluded regions are assigned the value 0 (black pixels in Fig. 4c).

To avoid redundancy, when initializing new centroids in the corner views, the regions that represent the same occluded 3D points in more than one corner view are kept only in one corner view and discarded from others (see, for example, the ovals with similar color in Fig. 4b). To achieve that, each corner view is iteratively warped into other corner views using its disparity map. Afterwards, pixels in the current corner view that overlap with the projected pixels from other corner views are kept only in the visibility mask of the current corner view and discarded from the visibility masks of other corner views. Moreover, the connected pixels (with 8-direction connectivity) in the occluded regions that are smaller than H_{size}, are also discarded. Finally, new centroids are initialized uniformly only in the remaining regions in the corner views that do not have corresponding centroids in the central view as applied earlier to the central view. After initializing the centroids in the central and corner views, that represent the hyperpixels, each pixel in 4D space will be clustered to the appropriate hyperpixel as explained in the next step.

B. 4D LF Pixels Label Assignment

In this step, each pixel in the 4D LF is labeled and assigned to the corresponding hyperpixel based on the similarity in the used clustering features. To exploit LF cues during the clustering, each pixel is characterized by a feature vector $[x, y, u, v, l, a, b, d]$ according to its position in the 4D space, where (x, y) are the spatial coordinates, (u, v) are the angular coordinates, (l, a, b) are the color components in the CIELAB color space, and d is the disparity value. To assign labels for all pixels in 4D LF, a modified version of the K-means clustering algorithm is used by considering an adaptive weighted clustering in 4D space.

In 4D LFs, considering cameras with a parallel optical axis, the scene is captured from different angular perspectives hence, views with spatial shifts are generated. These shifts lead to the appearance of slanted lines in the EPI space, as can be seen in Fig. 5 where the EPI slices with yellow and red borders are generated by first stacking the central horizontal and vertical LF views, respectively. Different from voxels in 3D space, the corresponding pixels that represent the same 3D point in 4D space have a spatial shift across views, horizontally and vertically, according to the disparity of each object in the scene.

Therefore, to support truly 4D clustering, the centroids are projected into each LF view to enforce the cross-view

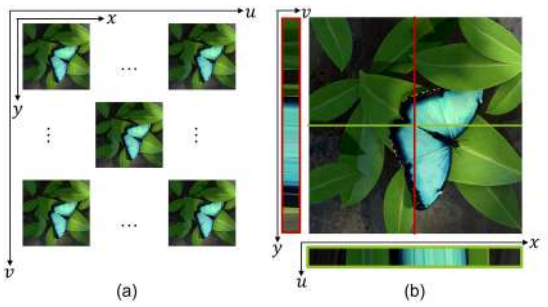


Fig. 5. In 4D LFs, each LF view (i.e., a slice of 4D LF in a particular angular plane (u, v)) captures the scene from a different view perspective, resulting in shifted light rays across views as can be seen in the yellow and red bordered EPIs shown below and to the left of the central view.

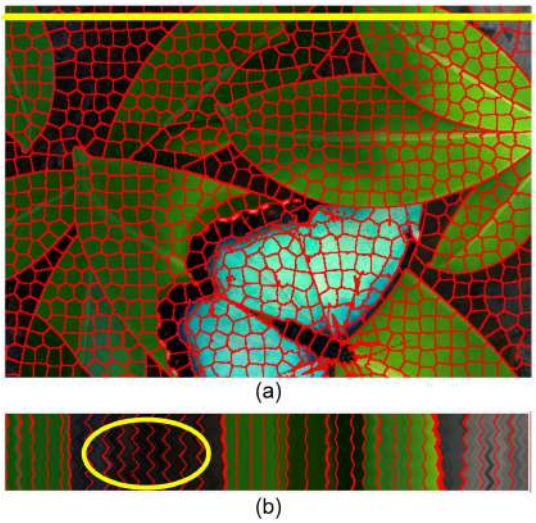


Fig. 6. To ensure consistency with respect to the EPI slanted nature, centroids are projected spatially during the 4D clustering. a) 2D view overlayed with hyperpixel borders; b) A stack of horizontal EPIs when projecting the centroids into each view.

consistency according to the slanted nature of the EPIs as in Fig. 6. Notice that the EPIs in Fig. 6 are generated by stacking the 4D LF views in serpentine order (to maintain connectivity in the EPI lines for better visualization), resulting in 2D horizontal EPI slices. Due to the differences in sampling the angular and spatial dimensions (especially for sparse LFs), a sampling compensation is needed. This can be achieved here by shifting the LF views using their disparity maps during the clustering to make the corresponding pixels aligned as described below.

More precisely, the 4D K-means clustering is applied in each view by spatially projecting the centroids, using their disparities, from their current angular position into each view without changing their angular dimensions, as in (1):

$$\begin{aligned} x_{\mathbf{c}'} &= x_{\mathbf{c}}^{(u',v')} = x_{\mathbf{c}}^{(u,v)} + d_{hor,\mathbf{c}}^{(u,v)\rightarrow(u',v')}, \\ y_{\mathbf{c}'} &= y_{\mathbf{c}}^{(u',v')} = y_{\mathbf{c}}^{(u,v)} + d_{ver,\mathbf{c}}^{(u,v)\rightarrow(u',v')}, \end{aligned} \tag{1}$$

where $(x_{\mathbf{c}'}, y_{\mathbf{c}'})$ are the spatial coordinates of the projected centroid, $\mathbf{c}'$, using the disparity of the centroid located in (u, v) view, and $d_{hor,\mathbf{c}}^{(u,v)\rightarrow(u',v')}$ and $d_{ver,\mathbf{c}}^{(u,v)\rightarrow(u',v')}$ are, respectively, the horizontal and vertical disparities from (u, v) view to (u', v') view. Considering that the used disparity estimation methods for densely and sparsely sampled 4D LFs generate per-pixel disparities from each view to its right horizontal adjacent view [11], [15], [17], the disparity value is here computed as in (2):

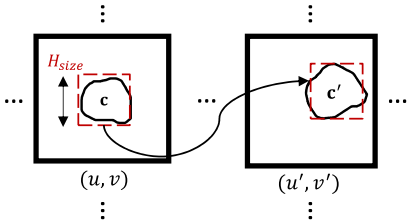


Fig. 7. Example of spatial projection of a hyperpixel centroid from view (u, v) into view (u', v') using the horizontal and vertical disparity values.

$$\begin{aligned} d_{hor,\mathbf{c}}^{(u,v)\rightarrow(u',v')} &= d_{\mathbf{c}} \times \left(u' - u\right), \\ d_{ver,\mathbf{c}}^{(u,v)\rightarrow(u',v')} &= d_{\mathbf{c}} \times \left(v' - v\right), \end{aligned} \tag{2}$$

where $d_{\mathbf{c}}$ is the disparity of the centroid, $\mathbf{c}$, from each view to its right horizontal adjacent view and (u, v) are the angular coordinates where the centroid is located. Notice that in (2) a uniformly sampled camera setup is considered. However, if the camera baselines are different for horizontal and vertical directions, then a consideration of camera parameters is needed [15]. When centroids are projected into other views, their spatial position $(x_{c'}, y_{c'})$ may belong to $\mathbb{R}^2$, however, color and disparity values in the used datasets are only available for integer positions. Therefore, the coordinates of the projected centroids are rounded to ensure integer indexing belonging to $\mathbb{Z}^2$. More precisely, for projection, unnormalized position and disparity values are used. However, during 4D clustering and weights adaptation steps, the normalized unrounded values are used.

Due to the high dimensionality of 4D LFs and since most hyperpixels usually have a local slice in each view, the searching of the nearest centroid is applied, as proposed for 2D images [29], in a small searching window, Ω_i, around each centroid in each view as defined in (3):

$$\Omega_i = (4 \times H_{size})^2, \tag{3}$$

where $i \in \{1, \ldots, K\}$, H_{size} is the hyperpixel size as in Fig. 7.

Let $H = \{H_1, \ldots, H_K\}$ represent the set of all hyperpixels where K is the number of hyperpixels. This way, the over-segmentation can be considered as an energy minimization problem in (4):

$$E = \arg\min_{H} \sum_{i=1}^{K} \sum_{\mathbf{p} \in H_i} D_w\left(\mathbf{p}, \mathbf{c}_i\right), \tag{4}$$

where $\mathbf{p}$ is a pixel in 4D space that belongs to hyperpixel H_i, D_w is the weighted distance, and $\mathbf{c}_i$ is the centroid of H_i in 4D space. In this step, each pixel in Ω_i is assigned to the "nearest" hyperpixel based on, D_w, as in (5)-(10):

$$\begin{aligned} D_w\left(\mathbf{p}, \mathbf{c}\right) = w_p \times D_p^2 + w_l \times D_l^2 \\ + w_a \times D_a^2 + w_b \times D_b^2 + w_d \times D_d, \end{aligned} \tag{5}$$

where w_p is the position clustering weight, w_l, w_a, w_b are the color clustering weights, w_d is the disparity clustering weight and D_p, D_l, D_a, D_b, D_d are the position, color and disparity distances between each pixel $\mathbf{p}$ and a target centroid $\mathbf{c}$, respectively, D_d here is not squared to impose a larger penalty on the disparity feature as in [17]. The distances in this paper are computed as follows:

$$D_p(\mathbf{p},\mathbf{c}) = \sqrt{\frac{(x_\mathbf{p}-x_{\mathbf{c}'})^2+(y_\mathbf{p}-y_{\mathbf{c}'})^2+(u_\mathbf{p}-u_\mathbf{c})^2+(v_\mathbf{p}-v_\mathbf{c})^2}{8\times H_{size}^2+(N_u-1)^2+(N_v-1)^2}}, \tag{6}$$

$$D_l(\mathbf{p},\mathbf{c}) = \sqrt{(l_\mathbf{p}-l_\mathbf{c})^2}, \tag{7}$$

$$D_a(\mathbf{p},\mathbf{c}) = \sqrt{(a_\mathbf{p}-a_\mathbf{c})^2}, \tag{8}$$

$$D_b(\mathbf{p},\mathbf{c}) = \sqrt{(b_\mathbf{p}-b_\mathbf{c})^2}, \tag{9}$$

$$D_d(\mathbf{p},\mathbf{c}) = \sqrt{(d_\mathbf{p}-d_\mathbf{c})^2}, \tag{10}$$

where $\mathbf{p}$ represents each pixel in 4D space that belongs to the searching window centered on centroid $\mathbf{c}$. Furthermore, $x_{\mathbf{c}'}$, $y_{\mathbf{c}'}$ are the spatial coordinates of centroid $\mathbf{c}$ when projected into the view of $\mathbf{p}$ with angular coordinates $(u_\mathbf{p}, v_\mathbf{p})$. Additionally, $(u_\mathbf{c}, v_\mathbf{c})$ is the original angular coordinate of centroid $\mathbf{c}$ without projection and N_u, N_v are the horizontal and vertical angular dimensions, respectively. The projected spatial position is used here to enforce cross-view consistency by considering the disparity between views and to compensate for the difference in sampling spatial and angular dimensions. To normalize the position feature, D_p is divided by $\left(8\times H_{size}^2+(N_u-1)^2+(N_v-1)^2\right)$, by considering the minimum distance to be zero and $\sqrt{8\times H_{size}^2+(N_u-1)^2+(N_v-1)^2}$ is the maximum distance in 4D space. In the first iteration, all the weights are initialized with the same value, equal to $1/|W|$, where W, is the set of clustering weights $\{w_p, w_l, w_a, w_b, w_d\}$ and $|W|$ is the number of the used clustering weights. As shown in [17], 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 $\sum w_{f\in\{p,l,a,b,d\}}=1$, in each iteration.

After assigning labels to all the pixels in 4D LFs, centroids are adjusted in terms of their features according to the current iteration as described in the next step.

C. Centroids Adjustment

In this step, the clustering features vector of each centroid $\mathbf{c}$ is adjusted iteratively until convergence is reached. After each iteration, the color feature values, $l_\mathbf{c}, a_\mathbf{c}, b_\mathbf{c}$, and the 4D position features, $x_\mathbf{c}$, $y_\mathbf{c}$, $u_\mathbf{c}$, $v_\mathbf{c}$, of each centroid are adjusted by the mean values of all pixels that belong to the corresponding hyperpixel, H_i, where $i\in\{1,\ldots,K\}$ as (11):

$$t_\mathbf{c} = \frac{1}{|H_i|}\sum_{\mathbf{p}\in H_i} t_\mathbf{p}, \tag{11}$$

where $t_\mathbf{p}$ is the feature value of a pixel, $\mathbf{p}$, in 4D space, and $t\in\{x,y,u,v,l,a,b\}$. Notice that, different than the existing LF over-segmentation methods, the proposed method also adjusts the angular coordinates. This is useful especially for the objects that exist only in some LF views and are occluded (partially or completely) or non-existent in other views.

Finally, to ensure robust centroid projection in the next iteration, and similar to [17], the disparity value of each centroid, $d_\mathbf{c}$, is updated using the actual disparity value of the centroid updated 4D position (rounded to integer positions) from the input disparity maps, d, as in (12):

$$d_\mathbf{c} = d\left(x_\mathbf{c}, y_\mathbf{c}, u_\mathbf{c}, v_\mathbf{c}\right). \tag{12}$$

After adjusting the centroids, the clustering weights still need to be adapted according to the current iteration; to avoid biased or non-optimal over-segmentation as explained in the next step.

D. Clustering Weights Adaptation

As the last step in each iteration and after the centroids are adjusted, the clustering weights are adapted according to the LF content and the current iteration. This step is beneficial especially when the features differ in their ranges. Moreover, selecting certain fixed values for clustering weights that suit different datasets without considering their content is a challenging, time-consuming task and may generate non-optimal over-segmentations. Since the use of adaptive weights has been shown to improve over-segmentation performance in [17] and [30], a similar technique is exploited here.

As in [30], the feature discriminability principle states that the features with the smaller within-cluster variances (i.e., the total sum of the feature distances from each pixel to its centroid in all hyperpixels) are more discriminative. Hence, it is beneficial to assign a larger weight to these features to properly influence the over-segmentation. The discriminability of each clustering feature can be computed by finding the normalized within-cluster variance for each feature, f, as in (13):

$$WV_f = \sum_{i=1}^{K}\sum_{\mathbf{p}\in H_i} D_f\left(\mathbf{p},\mathbf{c}_i\right)^2, \tag{13}$$

where K is the number of hyperpixels, $\mathbf{p}$ is a pixel in 4D space that belongs to hyperpixel H_i, $\mathbf{c}_i$ is the centroid of H_i in 4D space, D_f is the feature distance from each pixel, $\mathbf{p}$, and the centroid, $\mathbf{c}_i$, and $f\in\{p,l,a,b,d\}$. Unlike the technique in [30], but similar to [17], in this paper, the input 4D LF image and disparity maps are normalized before clustering. Therefore, we did not divide WV_f by the feature ranges, which is needed in [30] to normalize WV_f. After computing WV_f for each feature, the clustering weights are updated by assigning higher weight values to the features with smaller WV_f values using (14):

$$w_f = \frac{1}{\sum_{j\in\{p,l,a,b,d\}}\left(WV_f/WV_j\right)^{\frac{1}{|W|-1}}}, \tag{14}$$

where j represents each clustering feature and $|W|$ is the number of the used clustering weights.

E. Convergence Criterion

After applying the above steps, the iterative 4D clustering will continue until convergence or the maximum number of iterations is reached. To check for convergence, after each iteration, the average displacement of all centroids, D_{avg}, is computed by finding the 4D Euclidean distance between the previous centroid position in 4D space and the current 4D position. In this paper, we set the maximum number of iterations to 20 as will be discussed in the following section. Additionally, to improve the performance (in terms of the needed number of iterations), we considered a convergence threshold for D_{avg} of 0.7% of H_{size} (this value has been determined empirically after exhaustive testing). By choosing this threshold, we noticed, especially in dense 4D LFs, that the over-segmentation can converge before reaching the maximum number of iterations without a significant difference in accuracy.

F. 4D Versus 2D K-Means

In this section, the differences between the proposed 4D K-means clustering method and the 2D K-means clustering used in most of the available 4D LF over-segmentation clustering-based methods are briefly explained.

In the proposed method the centroids are initialized, before clustering, in the central view and in occluded regions in off-central views, as explained in Section IV-A. Other methods initialize centroids only in the central view, e.g., [11] and [17].

Besides the color feature, in the proposed method the 4D pixel position and disparity features are also considered during the clustering for all LF views. Other methods, either do not use disparity information as a clustering feature but merely for enforcing consistent centroids projection [11], or do not exploit the angular dimensions during the clustering [11], [15], [16], [17].

During the clustering, the centroids positions can be adjusted not only spatially but also angularly. In all other available methods [11], [15], [16], [17] the centroids are fixed angularly. Moreover, in the proposed method, disparity values are adjusted from the input disparity maps for each centroid after updating its 4D position. However, in most available methods, centroid disparity values are either never adjusted even when a centroid changes its position [11], or are adjusted to the mean disparity value of all pixels in the LF segment [15], [16].

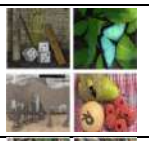
The proposed energy minimization function considers clustering weights for each feature to either penalize or increase its importance, with the weights being adapted to the LF content, similar to ALFO [17], which does not happen in other methods that rely on fixed values for clustering weights.

Consequently, the proposed method is truly 4D in nature and the creation of hyperpixels is based on grouping similar pixels in the 4D LF space. All other LF over-segmentation methods rely on projecting 2D superpixels in the center view to other LF views and then applying a final optimization.

V. Experimental Results

To evaluate the proposed 4D LF over-segmentation method, from here on simply called hyperpixels method, in various aspects, both dense and sparse, synthetic and real world LF datasets are used. Additionally, to validate the results, qualitative and quantitative comparisons with state-of-the-art methods are presented. In the following sub-sections, the used 4D LF datasets, benchmark methods to compare with and the used evaluation metrics are detailed. To clearly notice cross-view consistency, we highly encourage the reader to see the extended results on entire LFs in the supplemental materials for dynamic visualizations available online (please note that not all LF views are presented in this paper but can be found in the supplemental materials).[1]

TABLE II
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 dataset [31]: Buddha, Papillon, Horses, and StillLife	768×768 except for Horses: 1024×576	9×9	
MMSPG dataset [32]: Sphynx, Bikes, and Sophie	625×434	15×15	
Our generated dataset for sparse LFs: Kitchen, Room, Balloons, Antique, Car, Chess and Leisure	512×512 except for Leisure: 1280×720	9×9	

A. Used 4D LF Datasets and Experimental Setup

In this paper, three different datasets are used to generate hyperpixels for densely and sparsely sampled LFs as shown in Table II. In the case of dense LFs, the synthetic HCI 4D LF dataset [31], which contains Ground Truth (GT) disparity maps and 4D LF segmentation labels, is used. Moreover, only the central 11×11 views of the real world EPFL MMSPG dataset captured with a Lytro Illum camera [32] are used to eliminate the vignetting effects in corner LF views (i.e., darkening of the edges of the captured microimages).

For sparse 4D LFs, there is currently no available 4D LF sparse dataset with GT segmentation labels, GT disparity and depth maps for all LF views, which are needed for quantitative evaluation. For this reason, by using Blender software with Cycles rendering [33], LF Blender tools proposed by Honauer et al. [34], and some publicly available 3D models in [35], [36], and [37], we generated a new synthetic dataset accompanied by GT disparity maps, depth maps and segmentation labels, in order to enable the numerical evaluation. Our dataset has disparity values between adjacent views within the range $[-125, 125]$ and consists of 11 4D

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

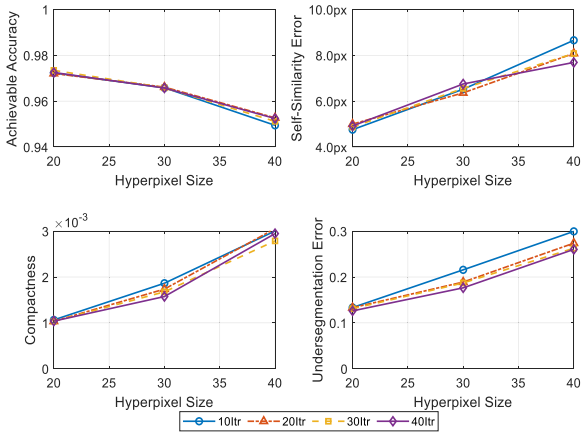


Fig. 8. Average quantitative evaluation of used test 4D LFs with different hyperpixel sizes and number of iterations.

LFs with (9×9) angular resolution and either (512×512) or (1280×720) spatial resolution. Our dataset contains several objects and challenging regions for segmentation, for example, non-Lambertian objects (e.g., glass and metal), complex textures, uneven lighting and overlapping objects with similar colors. As such, it can be used to evaluate various LF applications (the IT-4DLF dataset is available for download at: http://www.img.lx.it.pt/IT-4DLF/). In this paper, 7 challenging sparse 4D LFs and 7 dense 4D LFs from other commonly used datasets are used.

It is worth noting that our proposed method relies neither on any experimentally set clustering weights nor on any post-processing step. Most existing methods require cleaning or optimization as a post-processing step to fill unlabeled pixels due to inaccurate over-segmentation or to regularize the over-segmentation results across views. Like in existing clustering-based LF over-segmentation methods, the hyperpixel size is assigned by the user according to the desired application. It was observed that using adaptive 4D clustering enhances over-segmentation convergence [17]. The proposed hyperpixels method converges most of the time within 10 iterations. However, the maximum number of iterations was chosen to be 20 to ensure accurate labeling even for complex scenes. This value was selected after comparing the average performance for the used test images generated after 10, 20, 30 and 40 iterations. Since there was no significant improvement in the performance after 20 iterations, as shown in Fig. 8, this value was chosen as a convergence criterion. Our implementation is not optimized yet, but it has been shown in the literature [2], [10], [16] that K-means clustering can be parallelized for fast over-segmentation, which may be required for some applications.

B. Benchmark Methods and Experimental Parameters

In this paper, we compared our results with all the existing 4D LF over-segmentation methods listed in Section II namely: the Superray [11]; LFSP [15]; VCLFS [16]; HVLFS [18]; and ALFO [17] methods. The used software for these methods was obtained and used as detailed in [17]. To generate the superrays in [11], numerous parameters are required as input, such as the disparity range between adjacent views, and compactness weight (e.g., a weight that controls superrays compactness). The disparity range is obtained from the estimated disparity in [38] and [39] (as used for our method), for each test image independently and the compactness weight is set to 10, as it shows superior performance in [29], for different superrays sizes. As input to the LFSP method [14], [15], different methods are used by the authors of the LFSP method for estimating only the central disparity map without significantly affecting the performance, such as [40] and [41]. In this paper, the input disparity map of the central view that is used for the LFSP method is the same as the one used for our hyperpixels method for dense and sparse LFs. For the VCLFS [16], the maximum disparity parameter is merely changed according to each LF and this value is set using the same disparity maps that are used for our method. For the HVLFS method [18], we only have results provided by the author for dense synthetic LFs and superpixel size belonging to $[20, 45]$; hence, we could not compare this method with sparse LFs or compute its execution time. For ALFO method [17], disparity maps for all 4D LF views are required as input. Therefore, the used disparity maps for our method are also used for ALFO method.

Regarding the input hyperpixel size (a.k.a. cluster/segment size), H_{size}, for dense and sparse LFs, several sizes were tested on the HCI and our generated datasets (i.e., 20, 25, 30, 35, 40). For the MMSPG dataset, since there is no labeling GT available, only $H_{size} = 20$ is presented.

C. Evaluation Metrics

To generate the quantitative results, the evaluation metrics comprehensively described for 4D LF in [17] are used. Namely, the Achievable Accuracy (AA), Boundary Recall (BR), Under-segmentation Error (UE), Compactness (CP), Self-Similarity error (SS), and number of Labels per Pixel (LP). Notice that the existing consistency metrics used in [17] do not adequately consider regions that exist in other views but are occluded or non-existent in the central view, especially in sparse LFs. To overcome this limitation, the recently proposed LF Angular Consistency (LFAC) metric for style transfer applications [21] is adapted and modified to compute the consistency of sparse LF over-segmentation more accurately. Different from LFAC [21], where the consistency of RGB stylized LFs (i.e., composed LF in the style of another image) is compared with an original one and where the estimated disparity of the original image is used, in this paper, a labeled 4D LF is used to compute the angular consistency assisted with the GT disparity maps and segmentation label images for all LF views.

Labeling–LF Angular Consistency (LLFAC)– Given a GT 4D LF disparity map, D_{GT}, and GT segmentation label images, L_{GT}, the angular consistency is computed by initially grouping the hyperpixels into object-level using L_{GT}. To achieve that, each hyperpixel in the hyperpixel labeled image, L, is assigned to the label of the segment in L_{GT} that has the largest overlap with the current hyperpixel. Afterwards, the local angular variance map, $\sigma^2(L)$, is initially computed

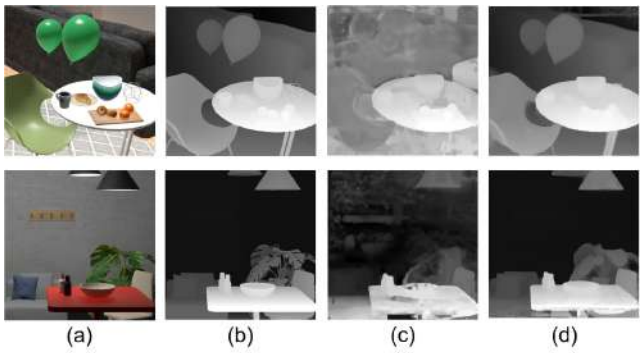


Fig. 9. Estimated disparity for sparse LFs: a) The central LF view for which the disparity estimated; b) GT disparity with range [−35.3, 8.7]; c) Results by using the deep learning based method in [39]; d) Results by applying our proposed modification on [39] to improve the accuracy and angular local consistency.

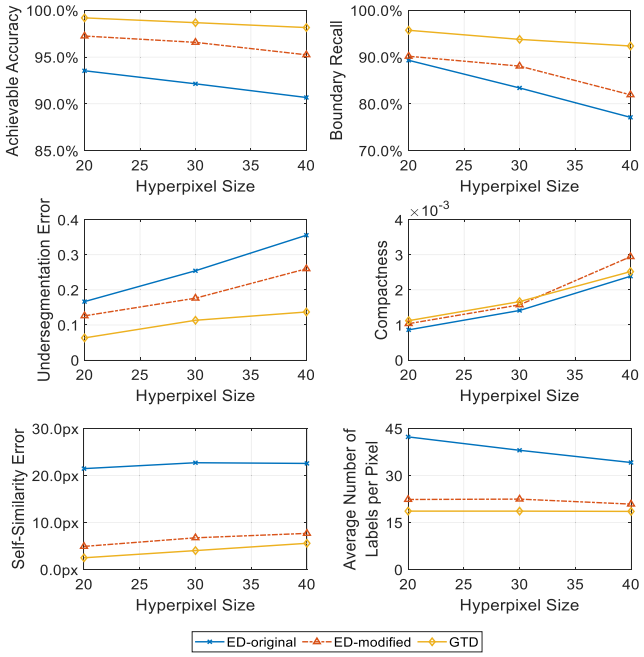


Fig. 10. Quantitative comparison on our proposed method using different estimated disparities namely, Estimated Disparity using (ED-original) [39]; modified Estimated Disparity (ED-modified); Ground Truth Disparity (GTD). Better disparity maps can significantly improve the hyperpixels performance.

as follows [21]:

$$\sigma^2(L) = \frac{1}{N_u \times N_v} \sum_{u,v}^{M,N} \frac{1}{|C_{u,v}|} \times \left\{ \sum_{u',v' \in C_{u,v}} occ_{u',v'}^{u,v} \left(w_{u',v'}^{u,v} \left(L_{u',v'} \right) - \overline{L_{u,v}} \right)^2 \right\},$$

$$LAC(L) = 10 \log_{10} \left(r^2 \Big/ \overline{\sigma^2(L)} \right), \quad (15)$$

where N_u, N_v are the number of horizontal and vertical views, $C_{u,v}$ is the closest 8 neighboring views of labeled view $L_{u,v}$, $occ_{u',v'}^{u,v}$ represents per-pixel weights where occluded regions between two adjacent views are set to 1 and 0 elsewhere, $w_{u',v'}^{u,v}$ represents the warping function as explained in [21], to warp a given view using a disparity map between view

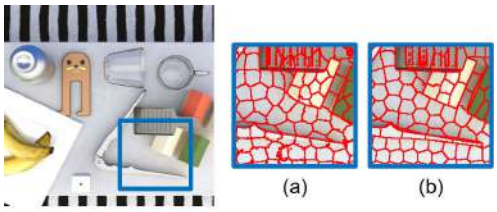


Fig. 11. Example of inaccurate over-segmentation of a non-Lambertian region of the nutcracker using different disparity maps during the clustering: a) Using estimated disparity; b) Using ground truth disparity. Accurate disparity maps can improve the over-segmentation performance.

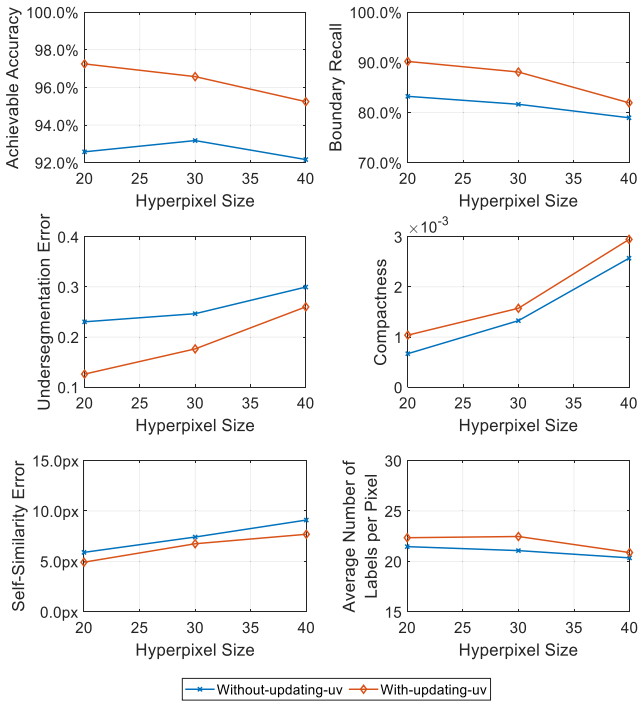


Fig. 12. Quantitative comparison of our proposed method with and without adjusting the centroids angular location during the clustering for sparse LFs.

$L_{u',v'}$ and $L_{u,v}$, $\overline{L_{u,v}}$, is the mean of all the LF neighboring views warped into view $L_{u,v}$, r is the pixels values range, and $\overline{\sigma^2(L)}$ is the mean of $\sigma^2(L)$. A higher *LLFAC* indicates better angular consistency.

D. Disparity Maps Estimation

As input, the proposed hyperpixels method requires disparity maps for all 4D LF views, to fully exploit LF cues during the 4D clustering. In the case of dense LFs, the recently proposed view-consistent depth estimation method in [38] is used. This method [38] relies heavily on the EPI structure and is designed only for dense LFs. In the case of sparse LFs, to the best of the authors' knowledge, only the deep learning based disparity estimation method proposed in [39] can estimate disparity (for all dense and sparse LF views, considering and ensuring cross-view consistency), with promising performance and has an open-source software. This method relies on initially estimating the corner views using a fine-tuned Flow Net 2.0 [42], [39]. Afterwards, the inner views disparity maps are synthesized and propagated using an occlusion-aware soft 3D reconstruction method proposed in [43] based on the corner

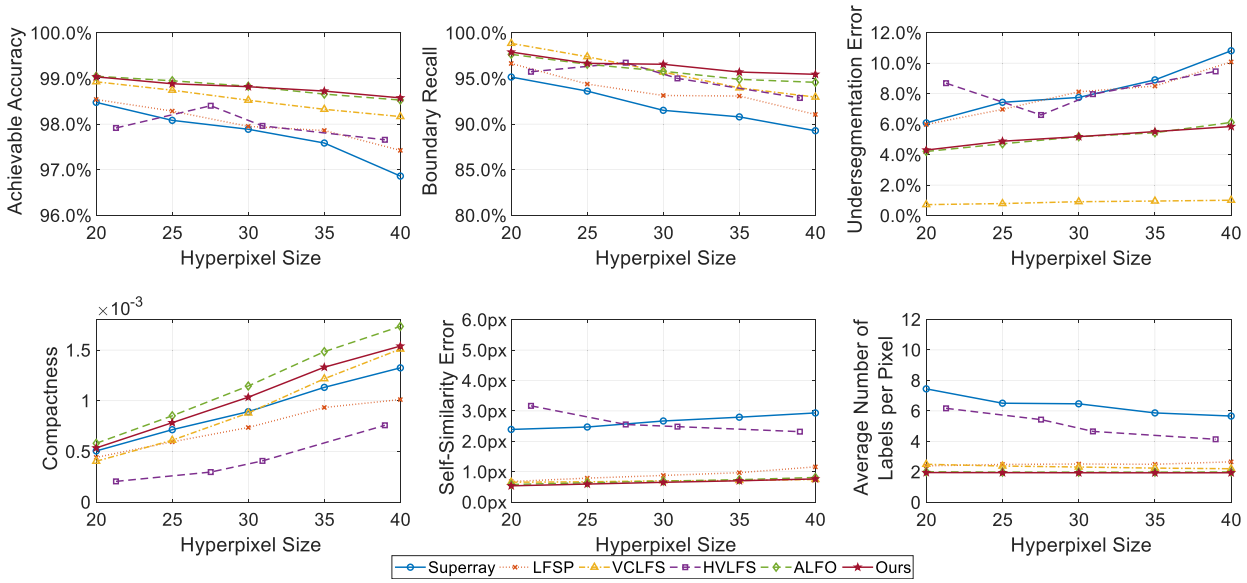


Fig. 13. Average quantitative evaluation on all 4D LFs of the dense HCI 4D LF dataset listed in Table II for different 4D LF over-segmentation methods.

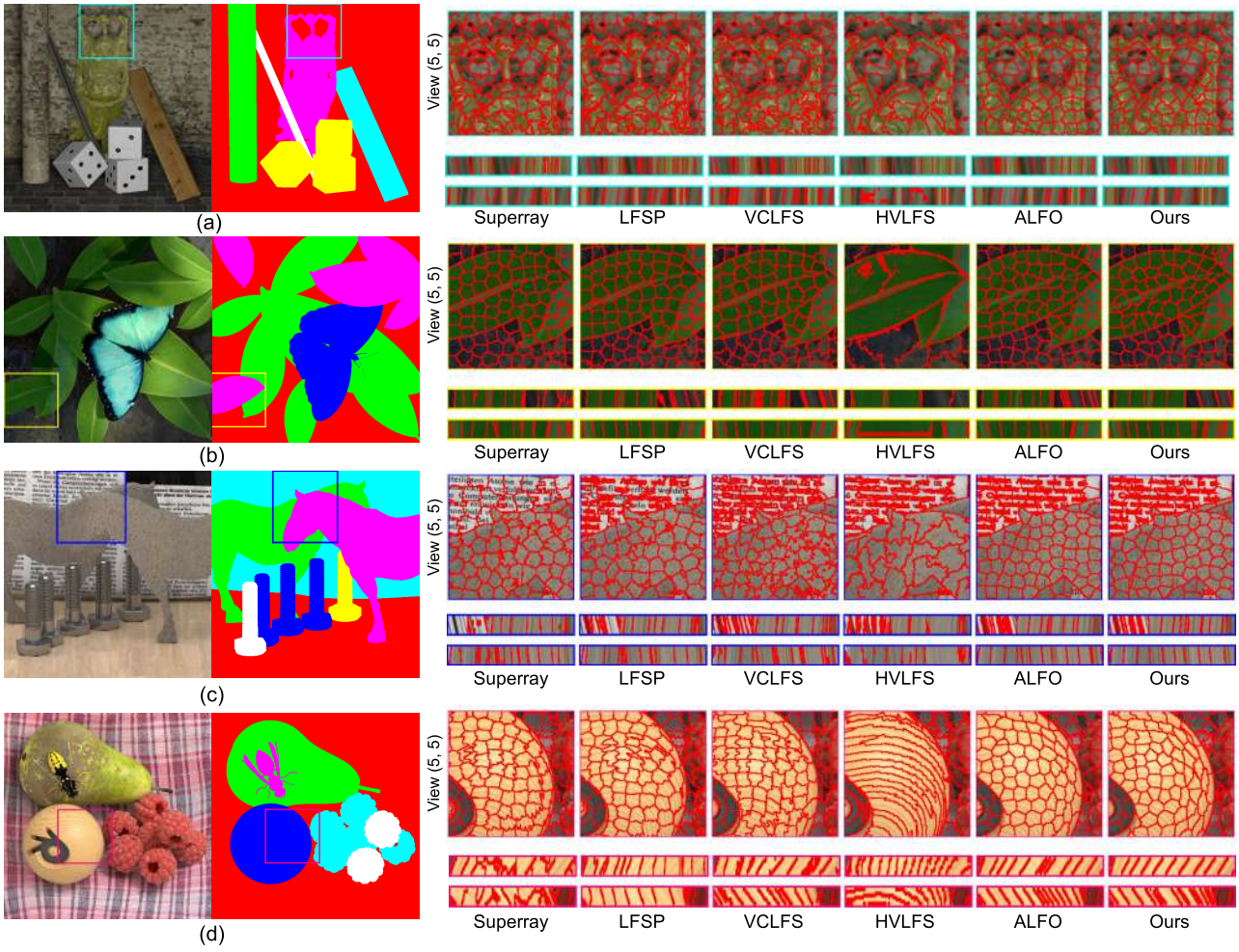


Fig. 14. Qualitative results using the densely sampled HCI 4D LF dataset. Challenging regions are selected to evaluate the robust balancing between spatial accuracy, compactness and cross-view consistency. For each LF, the central view, the vertical and horizontal EPIs are presented, respectively. As can be seen, our results adhere well to object boundaries and can accurately segment overlapping objects as in (b) and (c) and maintain compact and consistent across all views (as can be seen in the supplemental dynamic results). $H_{size} = 20$.

views. This method can generate accurate disparity maps for LFs with limited disparity ranges. However, the accuracy of the estimated disparity is significantly negatively affected when large displacements exist between the corner views, especially for sparse LFs, which can dramatically affect the over-segmentation results. The authors extended this method

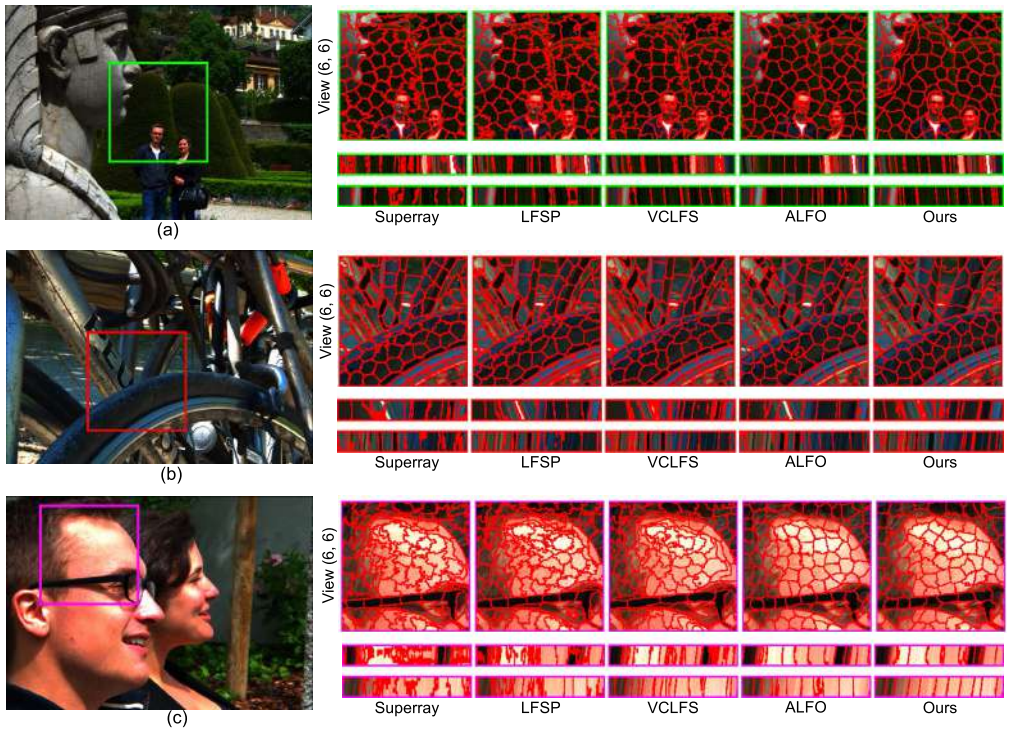


Fig. 15. Qualitative results using the densely sampled MMSPG 4D LF dataset. For each LF, the central view, the vertical and horizontal EPIs are presented, respectively. Regardless of the noise that exists in real LF views and non-even lighting, our results can adhere to object boundaries and can accurately segment challenging cases such as non-even lighting with complex texture and non-Lambertian regions and preserve compact and consistent across all views. $H_{size} = 20$.

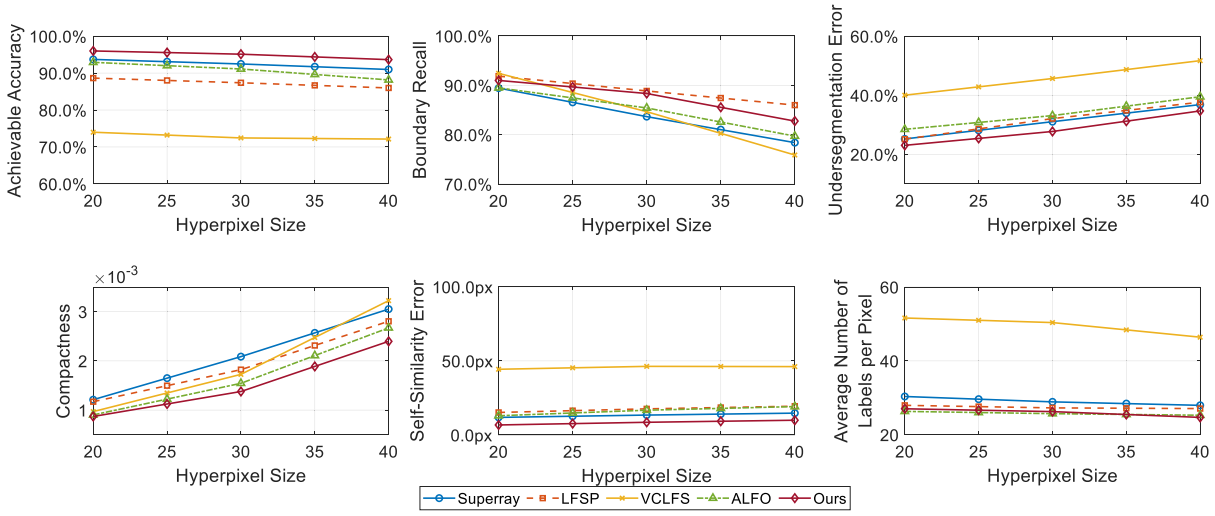


Fig. 16. Average quantitative evaluation on all 4D LFs of our sparse 4D LF dataset listed in Table II for different 4D LF over-segmentation methods.

in [44] to flexibly select any anchor views (e.g., not only corner views), but the disparity for only one target view can be estimated, hence no local or global angular consistency is considered when applying it for all LF views.

Therefore, the method in [39] is adopted in our experiment and the improved disparity estimation is used for all methods for sparse LFs. To ensure accuracy and local consistency in sparse LFs, instead of estimating the disparity for corner views and then propagating it to inner views that may include large-occluded regions, we estimate the disparity maps for every 4 adjacent views (e.g., 2×2) with step size equals to 2. This way, there is no need for propagation using 3D reconstruction as in [43], and a significant improvement in disparity estimation accuracy is achieved, as can be seen in Fig. 9. Consequently, our over-segmentation performance is further improved in terms of hyperpixel accuracy, compactness, and cross-view consistency as shown in Fig. 10 and as discussed in the following section.

In conclusion, inaccurate disparity estimation can affect the hyperpixels results, as shown in Fig. 11, and the proposed hyperpixels method is positively affected by using more accurate disparity maps.

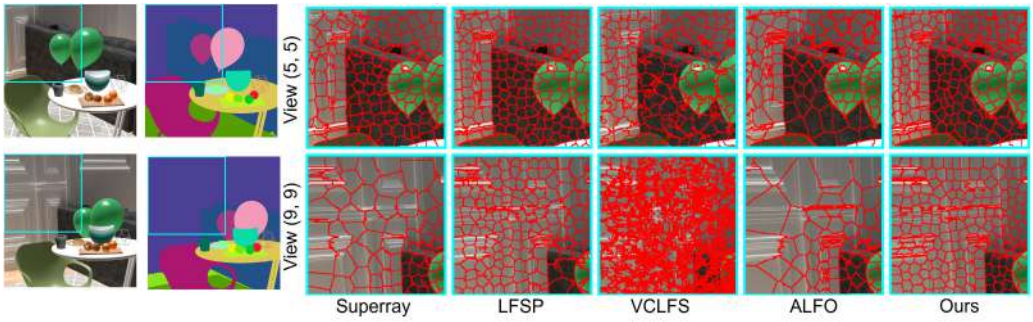


Fig. 17. Example of LF over-segmentation behavior for several methods for regions that do not exist in the central view. As can be seen inside the cyan square, a portion of the white region in view (9, 9) does not exist in the central view, i.e., view (5, 5). Our proposed method initializes centroids for these regions in 4D space before clustering. Therefore, hyperpixels remain with regular and similar sizes in all LF views and the accuracy and consistency are considered during the clustering for those regions.

E. Qualitative and Quantitative Results

In this section, our results are presented and compared to the benchmark methods for several dense and sparse 4D LF datasets. All the results in Fig. 10 – Fig. 18 are generated using estimated disparity maps as explained in Section V-D and not the GT ones. The GT disparity maps are only used for computing the quantitative evaluations.

Before comparing our results with the existing methods, it is worth showing the effect of updating the centroids angular location during the clustering. In the case of dense LFs, the disparity range is narrow and in our experiments the disparity ranges were always less than the H_{size}. Therefore, almost all hyperpixels have a 2D slice in all LF views. Consequently, the over-segmentation performance is not significantly affected by adjusting the centroids angular locations. However, in the case of sparse LFs, not all hyperpixels have a slice in all LF views; hence, the effect of updating the centroids angular location can be noticed. The importance of updating the centroids angular location during the clustering is shown in Fig. 12 for sparse LFs. In Fig. 12, the average performance in terms of accuracy, compactness and angular consistency is notably improved.

The performance evaluation of our method compared with other existing methods presented in (Fig. 13 – Fig. 18, where hyperpixel size is the same as cluster/segment size in other methods) can be summarized based on each metric as follows:

- **Achievable Accuracy** (↑)– This metric shows that using accurate disparity maps can affect the accuracy as seen in Fig. 10, where GT and different estimated disparity maps are used during the over-segmentation. As can be seen in Fig. 13 – Fig. 18, the hyperpixels method achieves outperforming accuracy by using adaptive 4D clustering along with hybrid spatio-angular features, for both dense and sparse LFs. The significance of exploiting disparity information as a clustering feature becomes apparent in challenging cases, such as overlapping objects with low color difference but at different depths. In Fig. 14b and Fig. 14c, overlapping leaves and the horses' heads are examples of this type of challenging regions.
- **Boundary Recall** (↑)– Our results robustly preserve the boundaries in dense and sparse LFs even in challenging regions, such as the horse heads in Fig. 14c, and non-Lambertian objects, as the glass cup in Fig. 18a. However, if inaccurate disparity values are estimated, the *BR* results can be negatively affected as clearly presented in Fig. 10.
- **Under-segmentation Error** (↓)– The proposed hyperpixels method balances the tradeoff between accuracy, shape

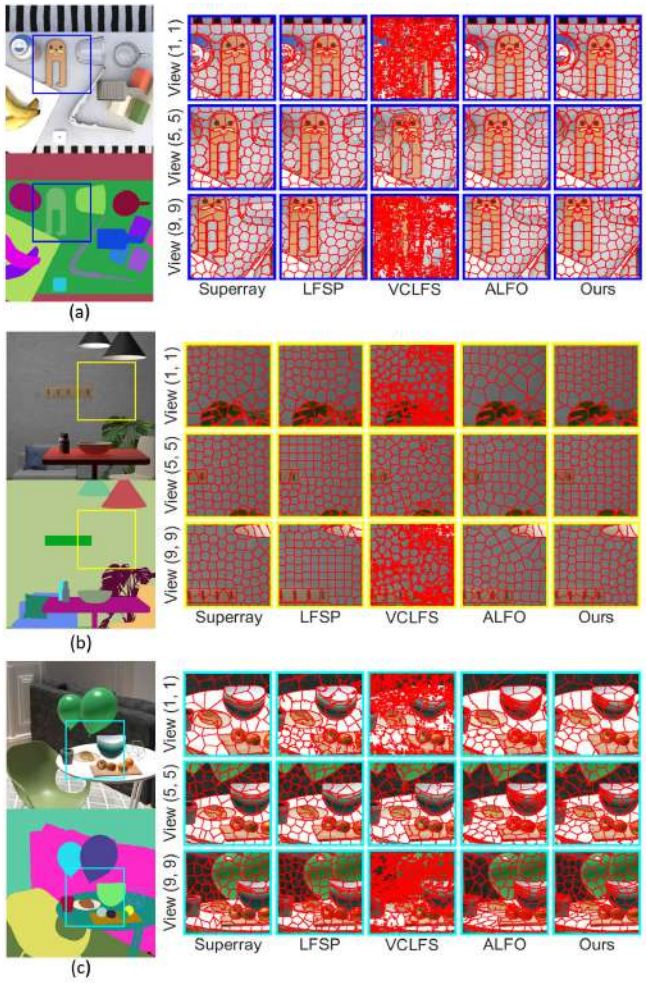


Fig. 18. Qualitative results using our generated sparse 4D LF dataset. Challenging regions are selected to evaluate the robust balancing between spatial accuracy, compactness, and cross-view consistency such as transparent glass, objects and large untextured regions as in the wall. Regardless of the wide disparity range in this dataset, the proposed hyperpixels are robust and consistent across views. $H_{size} = 20$.

TABLE III
LABELING–LF ANGULAR CONSISTENCY (LLFAC) FOR DENSE AND SPARSE LIGHT FIELDS (↑)

		H_{size}														
		20					30					40				
4D LF	Disparity range	Superray [11]	LFSP [15]	VCLFS [16]	ALFO [17]	Ours	Superray [11]	LFSP [15]	VCLFS [16]	ALFO [17]	Ours	Superray [11]	LFSP [15]	VCLFS [16]	ALFO [17]	Ours
Buddha	[-8.5, 1.5]	39.00	**39.53**	39.19	39.52	39.42	38.53	39.28	38.92	**39.47**	39.21	38.03	**39.20**	38.91	39.01	38.94
Papillon	[-1.2, 0.9]	36.63	36.71	36.38	36.86	**36.94**	36.61	36.68	36.53	36.92	**36.99**	36.58	36.62	36.65	36.95	**37.02**
Horses	[-1.4, 0.9]	36.67	37.11	37.86	37.91	**38.12**	36.71	37.11	**37.43**	37.19	37.40	36.31	36.66	37.08	**37.24**	36.89
StillLife	[-2.7, 2.6]	35.76	36.47	37.03	**37.20**	37.13	35.30	36.34	36.78	37.11	**37.15**	33.74	36.01	36.93	37.12	**37.25**
Kitchen	[3.1, 13.5]	**36.13**	35.57	26.99	35.24	35.27	**35.56**	34.98	27.07	34.81	34.67	**35.08**	34.99	27.60	34.29	33.52
Room	[-18.3, 8.9]	30.04	28.86	27.24	29.85	**30.62**	29.97	29.34	27.50	29.94	**30.64**	30.36	29.74	27.78	29.99	**30.78**
Balloons	[-35.3, 3.2]	25.06	30.85	27.90	32.22	**32.51**	24.88	31.10	28.06	31.84	**32.38**	25.50	30.51	27.95	31.89	**32.36**
Antique	[-5.44, 1.17]	40.94	**41.16**	37.64	39.67	39.15	**40.70**	40.03	37.90	38.98	38.73	**40.94**	39.13	35.72	37.65	37.94
Car	[-1.55, 70.31]	29.44	28.58	27.17	29.73	**31.01**	29.03	28.71	25.98	29.63	**30.70**	28.53	28.51	26.10	29.29	**30.92**
Chess	[-15.8, 9.8]	**31.73**	29.90	28.12	30.99	31.61	**31.58**	30.03	28.54	30.83	31.46	31.23	30.22	27.92	30.02	**31.38**
Leisure	[-34.28, 123.34]	26.12	25.95	23.52	26.31	**26.90**	26.10	26.08	23.77	26.45	**26.95**	26.25	26.28	24.18	26.55	**26.98**
Average		33.41	33.70	31.73	34.14	**34.43**	33.18	33.61	31.68	33.92	**34.21**	32.96	33.44	31.53	33.64	**34.00**

TABLE IV
AVERAGE CPU TIME FOR VARIOUS CLUSTERING-BASED METHODS OVER SEVERAL LF DATASETS AND SIZES (IN SECONDS FOR ALL LF VIEWS)

H_{size}	4D LF dataset	Superray [11]	LFSP [15]	VCLFS [16]	ALFO [17]	Ours
20	HCI dataset	109.35	237.19	1837.95	443.56	630.93
	MMSPG dataset	76.84	153.19	1075.98	252.26	404.91
	Our generated dataset for sparse LFs	48.47	134.83	5753.73	169.94	279.96
40	HCI dataset	85.46	175.61	1663.29	453.74	745.32
	MMSPG dataset	61.37	122.99	1009.83	274.15	468.15
	Our generated dataset for sparse LFs	37.55	95.36	5721.17	250.48	340.22

regularity (i.e., compactness) and consistency by using the clustering weights adaptation. Hyperpixels results generate competitive *UE* in dense LFs and outperform the benchmark methods for sparse LFs, as shown in Fig. 13 and Fig. 16. Using accurate disparity maps can reduce *UE* as in Fig. 10. While the LFSP and VCLFS methods lead to lower under-segmentation errors for dense LFs, this is not necessarily true in terms of accuracy or consistency metric performance, as in Fig. 13.

- **Compactness (↑)**– This metric reflects over-segmentation shape regularity that can be controlled during the clustering weights adaptation step. In most benchmark methods, the compactness parameter is either an input set by the user or is empirically set to a fixed value. However, in this paper, the clustering weight that affects the compactness is automatically adapted according to the LF content. The proposed method achieves competitive *CP* when compared to other benchmark methods for dense LFs. However, due to the new centroids creation in off-central views, we noticed that the benchmark methods achieve better *CP*. In some of the benchmark methods, when a region lacks a centroid projection, pixels in that region are grouped to the nearest segment, resulting in larger and more compact segments in off-central views as in Fig. 17. This situation increases the average compactness results and may affect the *AA* and *UE* performance. Hyperpixels compactness can be improved by using accurate disparity maps as in Fig. 10.
- **Consistency metrics: SS (↓), LP (↓), LLFAC (↑)**– LF Over-segmentation consistency is an essential property that can drastically affect subsequent editing tasks. The state-of-the-art methods have different techniques to ensure consistency, such as enforcing the continuity in the EPI space or using graph optimization. In this paper, we exploit per-pixel disparity to effectively project centroids across views and achieve cross-view consistency. As can be seen in Fig. 13, Fig. 16 and Table III, the proposed method achieves outperforming results in terms of *SS* and *LP* in dense and sparse LFs. Since there are no GT maps for the real LFs, angular consistency is not evaluated numerically. However, Fig. 14 and Fig. 15 show the angular consistency through the EPIs. Moreover, the angular consistency can be clearly noticed in the videos of the supplemental materials. Given the fact that in *SS* and *LP* metrics, the consistency is computed after warping the views into the central one and discarding the largely occluded region, we computed the *LLFAC* to fairly evaluate the labeling angular consistency for sparse 4D LFs. As seen in Table III and Fig. 16, the proposed method achieves the best angular consistency for sparse LFs. Additionally, as in Fig. 10, using better disparity estimation can improve cross-view consistency due to the accurate centroids projections.

To sum up, the proposed method achieves a robust balance between all the metrics for all tested 4D LFs without using any post-processing step to correct labeling the hyperpixels. For sparse LFs, we noticed that the methods that rely on post-processing optimization, such as the superray and LFSP methods, can generate compact and accurate over-segmentation for sparse LFs but are not necessarily

consistent across views. Moreover, a significant reduction is noticed in the VCLFS method performance when sparse LFs are used. Since the VCLFS method relies on the EPI structure and cannot adequately handle the irregular EPI structure in sparse LFs.

Finally, existing limitations in this proposed method, in some 4D LFs where the disparity is not accurately estimated (e.g., in real world 4D LF scenes and when non-Lambertian objects exist), inconsistent or inaccurate hyperpixels may be generated. As an example, Fig. 11 shows a failure case in a part of the metallic object that has inaccurate disparity values. To avoid that, using better disparity maps can significantly improve the final results. Finally, the current implementation is not optimized since this was out of this paper scope. Nevertheless, the average CPU time required by each method to over-segment 4D LFs using several LF datasets is shown in Table IV.

VI. Conclusion

In this paper, the concept of hyperpixel for 4D LFs is initially defined. After that, a 4D LF over-segmentation method based on 4D K-means clustering is proposed to be used for sparse and dense 4D LFs. Moreover, our proposed method initializes the centroids in an occlusion-aware manner and uses an adaptive weighted 4D K-means clustering based on hybrid features.

The proposed hyperpixels method can be used as a pre-processing step for sparse and dense LF processing and editing, such as semantic segmentation and saliency detection. Quantitative and qualitative results show outperforming over-segmentation performance for dense and sparse 4D LFs.

In the future, we will further investigate how to exploit the non-linearities in the EPI space for sparse LFs and non-Lambertian objects, to enforce hyperpixel consistency across views. Additionally, we will consider further extending our method to generate hyperpixels for 5D LF videos.

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 Mr. Numair Khan for publishing the software of all the used evaluation metrics that facilitated their comparisons.

References

[1] G. Wu et al., "Light field image processing: An overview," *IEEE J. Sel. Topics Signal Process.*, vol. 11, no. 7, pp. 926–954, Oct. 2017.

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

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

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

[5] 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.

[6] 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.

[7] M. Hamad, C. Conti, A. M. de Almeida, P. Nunes, and L. D. Soares, "SLFS: Semi-supervised light-field foreground-background segmentation," in *Proc. Telecoms Conf. (ConfTELE)*, Leiria, Portugal, Feb. 2021, pp. 1–6.

[8] S.-S. Huang, Z.-Y. Ma, T.-J. Mu, H. Fu, and S.-M. Hu, "Supervoxel convolution for online 3D semantic segmentation," *ACM Trans. Graph.*, vol. 40, no. 3, pp. 1–15, Jun. 2021.

[9] 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.

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

[11] 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.

[12] 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.

[13] M. Hamad, C. Conti, P. Nunes, and L. D. Soares, "View-consistent 4D light field style transfer using neural networks and over-segmentation," in *Proc. IEEE 14th Image, Video, Multidimensional Signal Process. Workshop (IVMSP)*, Nafplio, Greece, Jun. 2022, pp. 1–5.

[14] 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.

[15] 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.

[16] 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.

[17] M. Hamad, C. Conti, P. Nunes, and L. D. Soares, "ALFO: Adaptive light field over-segmentation," *IEEE Access*, vol. 9, pp. 131147–131165, Sep. 2021.

[18] 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.

[19] 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.

[20] 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. 7–55, 1987.

[21] D. Egan, M. Alain, and A. Smolic, "Light field style transfer with local angular consistency," in *Proc. IEEE Int. Conf. Acoust., Speech Signal Process. (ICASSP)*, Toronto, ON, Canada, Jun. 2021, pp. 2300–2304.

[22] R. Malik, "Learning a classification model for segmentation," in *Proc. 9th IEEE Int. Conf. Comput. Vis.*, Nice, France, Oct. 2003, pp. 10–17.

[23] 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.

[24] P. Li and W. Ma, "OverSegNet: A convolutional encoder–decoder network for image over-segmentation," *Comput. Electr. Eng.*, vol. 107, Apr. 2023, Art. no. 108610.

[25] C. Xu and J. J. Corso, "LIBSVX: A supervoxel library and benchmark for early video processing," *Int. J. Comput. Vis.*, vol. 119, no. 3, pp. 272–290, Sep. 2016.

[26] S.-H. Lee, W.-D. Jang, and C.-S. Kim, "Superpixels for image and video processing based on proximity-weighted patch matching," *Multimedia Tools Appl.*, vol. 79, nos. 19–20, pp. 13811–13839, May 2020.

[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., Sep. 2018, pp. 1–12.

[28] T. Herfet, K. Chelli, T. Lange, and R. Kremer, "Fristograms: Revealing and exploiting light field internals," 2021, *arXiv:2107.10563*.

[29] R. Achanta, A. Shaji, K. Smith, A. Lucchi, P. Fua, and S. Susstrunk, "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.

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

[31] 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.
[32] M. Rerabek and T. Ebrahimi, "New light field image dataset," in *Proc. 8th Int. Conf. Quality Multimedia Exper. (QoMEX)*, Lisbon, Portugal, Jun. 2016, pp. 1–2.
[33] *Blender—A 3D Modelling and Rendering Package*. Accessed: Dec. 9, 2021. [Online]. Available: http://www.blender.org
[34] K. Honauer, O. Johannsen, D. Kondermann, and B. Goldluecke, "A dataset and evaluation methodology for depth estimation on 4D light fields," in *Proc. Asian Conf. Comput. Vis. (ACCV)*, Taipei, Taiwan, Nov. 2016, pp. 19–34.
[35] *Chocofur Main Page*. Accessed: Dec. 28, 2021. [Online]. Available: https://chocofur.com/
[36] *BlenderKit—Get Free 3D Models, Materials & More Directly in Blender*. Accessed: Dec. 9, 2021. [Online]. Available: https://www.blenderkit.com/
[37] *3D Models for Professionals: TurboSquid*. Accessed: Dec. 9, 2021. [Online]. Available: https://www.turbosquid.com/
[38] N. Khan, M. H. Kim, and J. Tompkin, "View-consistent 4D light field depth estimation," in *Proc. Brit. Mach. Vis. Conf. (BMVC)*, Sep. 2020, pp. 1–13.
[39] X. Jiang, J. Shi, and C. Guillemot, "A learning based depth estimation framework for 4D densely and sparsely sampled light fields," in *Proc. IEEE Int. Conf. Acoust., Speech Signal Process. (ICASSP)*, Brighton, U.K., May 2019, pp. 2257–2261.
[40] T. Wang, A. A. Efros, and R. Ramamoorthi, "Depth estimation with occlusion modeling using light-field cameras," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 38, no. 11, pp. 2170–2181, Nov. 2016.
[41] 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.
[42] E. Ilg, N. Mayer, T. Saikia, M. Keuper, A. Dosovitskiy, and T. Brox, "FlowNet 2.0: Evolution of optical flow estimation with deep networks," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Honolulu, HI, USA, Jul. 2017, pp. 1647–1655.
[43] E. Penner and L. Zhang, "Soft 3D reconstruction for view synthesis," *ACM Trans. Graph.*, vol. 36, no. 6, pp. 1–11, Nov. 2017.
[44] 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.

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 include 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. She acts as a reviewer for IEEE ACCESS journal.

Caroline Conti (Member, IEEE) received the B.Sc. degree in electrical engineering from Universidade de São Paulo (USP), Brazil, in 2010, and the Ph.D. degree in information science and technology from Instituto Universitário de Lisboa (ISCTE-IUL), Portugal, in 2017. She is currently a Senior Researcher with the Multimedia Signal Processing Group, Instituto de Telecomunicações, and an Assistant Professor with the Department of Information Science and Technology, Instituto Universitário de Lisboa (ISCTE-IUL), Portugal. Her research interests include immersive visual technologies and image and video processing, including light field processing and coding. She has contributed more than 25 papers to international journals and conferences in these areas. She serves as an Associate Editor for IEEE TRANSACTIONS ON IMAGE PROCESSING. She has been a Guest Editor for *Signal Processing: Image Communication* (Elsevier). She actively participates as a reviewer for various IEEE and EURASIP journals and conferences.

Paulo Nunes (Member, IEEE) received the degree in electrical and computers engineering from 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 Department of Information Science and Technology, 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 for the European Commission. 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 a member of the program and organizing committees of various international conferences. He has contributed more than 70 papers to international journals and conferences in these areas.

Luís Ducla Soares (Senior Member, IEEE) received the Licenciatura and Ph.D. degrees in electrical and computer engineering from 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 Department of Information Science and Technology, 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 70 papers to international journals and conferences in these areas. 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). In parallel, he acts as a reviewer for several IEEE, IET, and EURASIP journals and conferences.

Chapter 5

SLFS: Semi-supervised Light-field Foreground-background Segmentation

Conference: Telecoms Conference (ConfTELE)

Date of Conference: February 2021

Location: Leiria, Portugal

Pages: 1-6

DOI: 10.1109/ConfTELE50222.2021.9435461

SLFS: Semi-supervised Light-field Foreground-background Segmentation

Maryam Hamad
Instituto Universitário de Lisboa (ISCTE-IUL), Instituto de Telecomunicações,
Lisboa, Portugal
maryam.hamad@lx.it.pt

Caroline Conti
Instituto Universitário de Lisboa (ISCTE-IUL), Instituto de Telecomunicações,
Lisboa, Portugal
caroline.conti@lx.it.pt

Ana Maria de Almeida
Instituto Universitário de Lisboa (ISCTE-IUL), ISTAR, Lisboa, Portugal, CISUC-Center for Informatics and Systems of the University of Coimbra
ana.almeida@iscte-iul.pt

Paulo Nunes
Instituto Universitário de Lisboa (ISCTE-IUL), Instituto de Telecomunicações,
Lisboa, Portugal
paulo.nunes@lx.it.pt

Luís Ducla Soares
Instituto Universitário de Lisboa (ISCTE-IUL), Instituto de Telecomunicações,
Lisboa, Portugal
lds@lx.it.pt

***Abstract*—Efficient segmentation is a fundamental problem in computer vision and image processing. Achieving accurate segmentation for 4D light field images is a challenging task due to the huge amount of data involved and the intrinsic redundancy in this type of images. While automatic image segmentation is usually challenging, and because regions of interest are different for different users or tasks, this paper proposes an improved semi-supervised segmentation approach for 4D light field images based on an efficient graph structure and user's scribbles. The recent view-consistent 4D light field superpixels algorithm proposed by Khan et al. is used as an automatic pre-processing step to ensure spatio-angular consistency and to represent the image graph efficiently. Then, segmentation is achieved via graph-cut optimization. Experimental results for synthetic and real light field images indicate that the proposed approach can extract objects consistently across views, and thus it can be used in applications such as augmented reality applications or object-based coding with few user interactions.**

Keywords—light field segmentation, foreground-background segmentation, superpixels, graph-cut, semi-supervised segmentation

I. Introduction

When humans look at images, their brains can easily classify the objects in the scene by distinguishing the object's borders and understand the content. However, this task is much harder for computers which consider the scene as an array of pixels. To analyze the scene and understand its content by identifying meaningful objects, computers typically must start by applying image segmentation, which is the process of partitioning an image into smaller parts with homogenous properties. In computer vision, there are low-level, mid-level and high-level image segmentation techniques depending on the semantic meanings of the resulting segments. Basically, low-level image segmentation divides the image into smaller regions automatically with similar visual characteristics, such as color or depth, but not necessarily with a semantic meaning, and it can be used as a pre-processing step for object tracking or image editing [1], [2]. Mid-level image segmentation divides the image into a smaller number of larger regions (i.e., objects), it may be assisted with user interaction, however, it does not have semantic labels for the objects [3]. In addition to the mid-level segmentation output, the high-level image segmentation, can be assisted with high-level knowledge or learning process to obtain semantic meaning for the objects (e.g., a car, a flower, etc.) [4], which is out of this paper's scope. In this paper, a combination of low-level image segmentation and user scribbles are considered to obtain mid-level (e.g., foreground-background segmentation) without having pre-defined semantic labels for the objects.

Fig. 1. Example of the proposed segmentation approach: a) a reference image with user's foreground and background scribbles; b) the segmented object based on the scribbles.

Although image segmentation is usually considered as a challenging problem, certain conditions can make it even harder, such as overlapping between objects with poor contrast or the huge amount of data, as in the 4D Light Field (LF) images, specifically when pixels are used as graph nodes. 4D LF images can be obtained by an array of cameras or by a single camera equipped with a special microlens array in front of the sensor or a moving camera gantry to capture different viewpoint images at different times. LF images record not only the intensity of light but also the angular direction of light rays [5]. The resulting 4D LF image, which can have a very large number of pixels, can be interpreted as a 2D array of 2D views and parametrized as $L(x, y, u, v)$ where x, y are the spatial geometry of pixels in each view and u, v are the angular geometry of views. The 2D views are obtained from slightly different perspectives. While the 4D LF images contain a huge number of pixels, the similarity between pixels in different views can be used to reduce the computational complexity [1]. Furthermore, one of the most important advantages of 4D LF imaging is that it inherently includes depth information in its structure, which can be used in clustering and label propagation. In general, when traditional 2D segmentation is applied to 4D LF images, the information from adjacent views is not considered to resolve object occlusions, thus resulting in inconsistent segmentation across views. In order to cope with these challenges, the 4D LF image structure should be adequately considered. Various LF segmentation techniques have been proposed in the literature [3], [6]–[10]. However, most 4D LF segmentation techniques are either time-consuming, not interactive, not proposed for full consistent 4D LF segmentation or relying on accurate depth estimation.

This work was funded by FCT/MCTES through national funds and when applicable co-funded by EU funds under the project UIDB/50008/2020.

To overcome the existing limitations and because the regions of interest are different for different users or tasks, an improved interactive Semi-supervised 4D LF Foreground-background Segmentation (SLFS) solution is proposed (see Fig. 1). This approach can be widely applied in object-based LF coding, augmented reality applications, or object extraction. Similar concepts to the segmentation algorithm proposed in [9], such as the graph-based image segmentation and the graph-cut optimization technique are used in this paper. However, different superpixel algorithm (i.e., the state-of-the-art View Consistent Light Field Superpixel (VCLFS) [10]) is exploited as graph nodes, enabling a dramatic reduction in the size of the graph and to effectively propagate the segmentation consistently across views, without the need for extra accurate depth estimation algorithm.

The remainder of the paper is organized as follows: Section II briefly reviews the related work on 4D LF image segmentation available in the literature; Section III describes the proposed approach in detail; Section IV evaluates the SLFS performance through a series of experimental results; Section V concludes the paper with some final remarks and proposes directions for future work.

II. Related work

Image segmentation is a fundamental task in computer vision, and it has been attracting the attention of researchers for many years. Several image segmentation solutions for 2D images have already been proposed, however, only a few solutions have been proposed to tackle the 4D LF challenges, such as the huge amount of data and the need for ensuring the segmentation consistency across views. For low-level image segmentation, 4D LF superpixels/superrays have been proposed in [1], [8], [10] and can be used to enhance LF editing tasks (e.g., by propagating the edits into a 4D LF superpixel instead of a single pixel). For the case of mid-level image segmentation, Wanner et al. [3] proposed the first variational framework for multi-label segmentation, where the color and disparity cues of input seeds are used to train a machine learning classifier (i.e., random forest) that is used to predict the label of each pixel. However, the segmentation is not performed on the full 4D data (only the central view is segmented), the authors mentioned that the optimization step can take ~5 minutes on a modern GPU if applied for all views. Mihara et al. [6] improved Wanner's approach by building a graph in 4D space with spatial and angular neighbors and then using graph-cut for multi-label segmentation. Due to the huge number of graph nodes and the high computational time, only a fraction of the LF views (i.e., 5×5) were considered in the experiments. To reduce the graph size, Hog et al. [7] proposed a novel graph representation that utilizes the ray bundle (i.e., a set of all rays describing the same 3D scene point) as a graph node and exploited the redundancy in the LF data, decreasing the running time of the Markov Random Field (MRF) optimization and achieving entire 4D LF views segmentation. However, their approach depends on quite accurate depth estimation on all the views, thus, inaccurate individual depth maps greatly increase the running time and decrease the segmentation coherence. Additionally, the segmentation results can be very sensitive to the noise in real LF images.

It has been proven the efficiency of achieving mid-level and high-level segmentation based on low-level (e.g., superpixel) segmentation [2]. Lv et al. [9] recently proposed a novel hypergraph representation for 4D LF multi-label segmentation by exploiting the superpixels proposed in [8] as hypernodes to reduce the graph size. However, Lv et al.'s approach relies on superpixel segmentation that requires depth estimation from extra algorithm, hence, it can be time-consuming. Additionally, it is not as accurate for real LF images as for the synthetic LF images due to the lack of accurate estimated depth map. Our approach is different from the recent work in [9], by replacing the used superpixels and simplifying the graph structure and size. Our approach is designed to interactively extract foreground from background similar to the recent work in 2D images [2], however, the segmentation is applied for all 4D LF data to achieve effective interactive segmentation of user's region of interest.

III. Proposed light field segmentation approach

In order to achieve foreground-background 4D LF image segmentation, the proposed approach consists of four major steps (see Fig. 2):

A. LF superpixel extraction

In contrast to the widely used 2D superpixel algorithms, such as Simple Linear Iterative Clustering (SLIC) in [11], which divide an image into smaller clusters with similar visual appearance and spatial geometry, 4D LF image segmentation algorithms need to consider the depth information to extract consistent 4D LF superpixels. From the few proposed 4D LF superpixel algorithms, the state-of-the-art VCLFS algorithm is used in our proposed algorithm for the following reasons. Firstly, the VCLFS algorithm does not require an external depth estimation algorithm, since it implicitly estimates the disparity by computing the slopes of Epipolar Plane Image (EPI) lines for all LF views [10]. Secondly, the occluded objects where the foreground and background lines are intersected in the EPI are considered in the VCLFS algorithm and properly detected to prevent wrong segmentation. Finally, it outperforms other LF superpixel algorithms, notably [8], that is used in the recent 4D LF multi-label segmentation algorithm [9], in terms of boundary adherence, view consistency and running time [10], which is important for later foreground and background segmentation.

The VCLFS algorithm consists of three major steps: i) line extraction from the EPIs of central horizontal and vertical views of a 4D LF image; ii) occlusion-aware EPI segmentation; and iii) spatio-angular clustering by projecting the EPI segments of the central views into the central view and firstly clustering the central view using K-means algorithm, where the CIELAB color space, position and disparity are used. Afterward, the clustering labels are propagated across all views based on the EPI segments and disparity. After superpixels are extracted, the texture is characterized by using histograms of the superpixels' intensities. To compute the histograms, the image is converted to the Hue, Saturation and Value (HSV) color space first. The HSV color space is designed to approximate the human vision perception and it is widely used for image analysis and segmentation [12]. To achieve luminance invariance, the value channel is not considered, and the histogram is computed using only the hue and saturation channels. For each superpixel, a 2D histogram of hue and saturation values is computed. Each superpixels' histogram is normalized by dividing it by its sum. The obtained superpixels and the corresponding histograms will be used in the next step to create the graph representation.

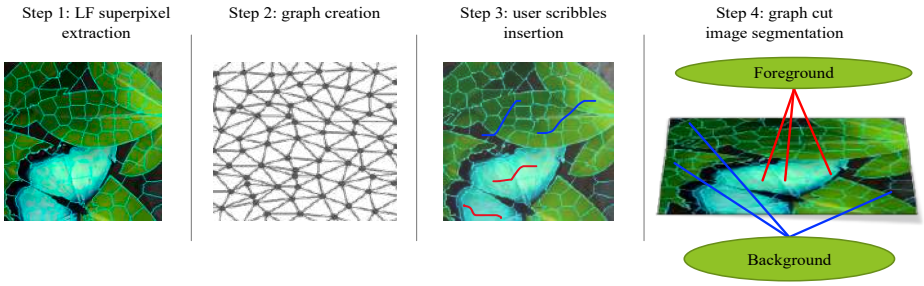


Fig. 2. Overview of the proposed SLFS algorithm: step 1) LF superpixels are extracted using the VCLFS algorithm; step 2) a graph is created using superpixels as graph nodes; step 3) scribbles are inserted by the user to initially label foreground and background superpixels; step 4) a graph-cut optimization is performed to the central view and propagated to the entire 4D LF views to iteratively achieve interactive foreground-background segmentation.

B. Graph creation

Since our goal is to improve the 4D LF segmentation, the theory of graphs can be applied similarly to what has been done for 2D image segmentation. However, in the context of the 4D LF segmentation, several algorithms used a graph representation of the 4D LF image by representing each pixel as a graph node [6]. Due to the huge size of a LF image, the number of resulting graph nodes is also massive, leading to a high computational complexity not suitable for 4D LF interactive applications. In contrast, the hypergraph concept which is conceptually defined and used in [9] is similarly used in our approach and significantly reduces the graph size by defining the extracted 4D LF superpixels as graph nodes, however, we did not consider the angular neighbors or the multiple-target nodes as in [9]. Generally, a hypergraph is one type of graph representation that uses a set of nodes as one hypernode as well as the connected edges between two hypernodes as one hyperedge (see Fig. 3). Additionally, the hypergraph is coarsened into a planar graph by considering all corresponding superpixels across views as one hypernode.

In our graph representation, a planar graph is created on the central view superpixels and conceptually represented a hypergraph, where each hypernode in the central view graph includes all corresponding superpixels across views. The central view is chosen for two reasons: i) in dense 4D LF images, there is only a slight shifting across views and according to the Lambertian assumption, the 3D point of the scene is corresponding to a straight line in the EPI [10]. Thus, most superpixels in the central view having corresponding superpixels in all LF views with small disparities; and ii) the user is usually interested in segmenting frontal objects instead of small occluded objects. The corresponding superpixels across views are computed in the VCLFS by changing the spatial position of the central view superpixels based on the angular location of the view and the superpixels' disparities, and it assigns a same numeric label to the corresponding superpixels. The final segmentation will be propagated by assigning the corresponding superpixels across views, the same foreground or background labels as central view superpixels. In Fig. 3, a simplified hypergraph illustration is shown. In the red rectangle, there is an edge between two superpixels, similarly, the red edge exists in all 4D LF views in Fig. 3. The hypernodes S_i, S_j can be shown in the two circles below and connected with a hyperedge. In order to represent a graph, we need to define the edges between the graph nodes and compute their weights. Since superpixels' shapes are irregular in most situations, the Delaunay Triangles algorithm[1] [13] is used to find the graph edges between neighboring superpixels' centroids. The Delaunay algorithm provided in the open-source Python library Sci-Py [14], [15] is used here.

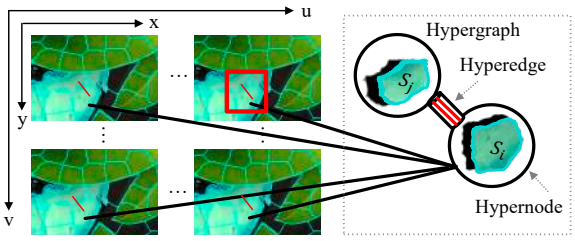


Fig. 3. The hypergraph representation where all corresponding superpixels across views are represented as one hypernode as in S_i *and* S_j. The red lines represent edges between two neighboring superpixels and, similarly, all corresponding edges between two hypernodes are represented as one hyperedge.

To create the graph G and perform graph-cut optimization to achieve foreground and background segmentation, the LF superpixels are used as nodes of the graph. Furthermore, two target nodes are added to the graph, for the foreground T_f (source node) and the background T_b (sink node), respectively (see Fig. 2). The maximum flow from the source to the sink is determined by the bottleneck (i.e., the edges minimum cut). Additionally, two different edge types are defined: i) target edges (i.e., the edges between the superpixel and the target nodes); and ii) neighboring edges (i.e., edges between spatially neighboring superpixels). After defining the types of the nodes and edge, we build a graph $G = (\nu, \varepsilon)$ of the central view, where ν represents both superpixels and target nodes, and ε represents edges between nodes. Each edge between superpixels is weighted by comparing the adjacent histograms using average Kullback-Leibler Divergence (KLD) [16] to compute the relative difference between histograms as in (1):

$$\mathcal{W}(S_i, S_j) = \mathcal{W}(S_j, S_i) = \lambda - \frac{1}{2}\left(\sum_x H_i(x) \log\left(\frac{H_i(x)}{H_j(x)}\right) + \sum_x H_j(x) \log\left(\frac{H_j(x)}{H_i(x)}\right)\right), \quad (1)$$

where $H_i(x)$ and $H_j(x)$ are, respectively, the hue and saturation 2D histograms of spatially adjacent superpixels S_i and S_j in the central view (as a complexity tradeoff in this

[1] The Delaunay algorithm finds a subdivision of a set of points into a non-overlapping set of triangles, such that no point is inside the circumcircle of any triangle.

paper, summations are over 20 histogram bins), and λ is a control parameter that helps in the graph-cut optimization process (after extensive testing, in our experiments a default value of $\lambda = 25$ was used since it led to the best subjective results); this parameter is especially useful in case of very small or null difference between the superpixel histograms.

C. User scribbles insertion

For semi-supervised interactive segmentation, a user can insert different scribbles to indicate the region of interest on the reference view. In this paper, the central view is selected as a reference view, since almost all views contain central view content with slight shifting. All superpixels under the scribbles are labeled either foreground or background, according to the scribble's label, and utilized as initial seeds to label unlabeled superpixels in the graph-cut step, where the cumulative foreground and cumulative background histograms are used. When user scribbles are inserted, graph target edges between labeled superpixels and target nodes are generated. Considering a superpixel under foreground scribbles, the edge weights between superpixel node S_f and the target nodes T_f and T_b represent the self-penalty D_S (i.e., the cost of labelling each superpixel as either foreground or background) as in (2) and (3):

$$D_{S_f}(T_f) = D_{S_i}(0) = 0\,, \qquad (2)$$

$$D_{S_f}(T_b) = D_{S_i}(1) = \mathrm{W}_{max}, \qquad (3)$$

where $D_{S_f}(T_f)$ is the edge weights between the foreground labeled superpixel and T_f (labeled as zero), and $D_{S_f}(T_b)$ is the edge weights between the foreground labeled superpixel and T_b (labeled as one). A small value is assigned for foreground target edge if the superpixel is under foreground scribbles, while a high value is assigned for the background target edge to increase the self-penalty. In our experiments we fixed W_{max} to 100 as a high value. The same approach is used for superpixels under background scribbles.

D. Graph-cut image segmentation

Generally, image segmentation can be formulated as the minimization of an energy cost function with two additive terms: i) the self-penalty (a.k.a data cost); and ii) the neighboring penalty (a.k.a smooth cost). Self-penalty represents the cost of labelling each superpixel as either foreground or background. Furthermore, the neighboring penalty ensures that neighboring superpixels are smooth and penalizes neighbors that have different labels.

To achieve the segmentation, graph-cut optimization is used, which is effective and handles image segmentation in terms of energy minimization [9]. The cumulative foreground and background histograms (H_{CF}, H_{CB}) of the superpixels under the user scribbles are computed separately after the user's insertion. In order to assign a label for each unlabeled superpixel, the KLD is used to compute the relative difference between cumulative target histograms and the superpixel histogram as in (4):

$$D_{S_i}\left(T_{f\ or\ b}\right) = \sum_x H_{CF\ or\ CB}(x) \log\left(\frac{H_{CF\ or\ CB}(x)}{H_i(x)}\right), \qquad (4)$$

where $D_{S_i}\left(T_{f\ or\ b}\right)$ is the self-penalty, and $H_{CF\ or\ CB}$ is foreground or background cumulative histogram. Suppose L is a label vector, which includes foreground (0) and background (1) labels for all the N superpixels $\mathrm{L} \in \{0,1\}^N$. The energy function is computed by summing the data cost and smooth costs for assigning label l_i to superpixel S_i considering the labels of the neighbors $\mathcal{N}_i$ as in (5) [17]:

$$\mathrm{E(L)} = \sum_{S_i \in I} D_{S_i}(l_i) + \sum_{(i,j)\in \mathcal{N}_i} \mathcal{W}(S_i, S_j)\left|l_i - l_j\right|. \qquad (5)$$

Finally, the graph-cut algorithm is applied to minimize the energy function to obtain the segmented result $\mathcal{S}$ as in (6):

$$\mathcal{S} = \arg\min_{\mathrm{L}} \mathrm{E(L)}, \qquad (6)$$

where the energy function E(L) is the cost of assigning label l_i to each superpixel S_i in the image I by summing the data cost and the smooth cost for each superpixel S_i and its spatially neighboring superpixels S_j, where $\mathcal{N}_i$ is the set of neighboring superpixels of S_i. In our algorithm, we take advantage of the optimized PyMaxFlow library to apply the graph-cut that implements the algorithm in [17] for central view. Since each superpixel in the central view conceptually represents a hypernode of all self-similar superpixels across views, the superpixels' labels from the central view are propagated to the entire 4D LF views by assigning each superpixel related to the hypernode to the label of the superpixel in the central view. The graph-cut optimization is interactively continued after each user's scribble insertion of both foreground and background scribbles, and the calculation of the cumulative target histograms are updated until the object segmentation is achieved according to the user's decision. Finally, the border's noise is removed from the final mask using median filtering with kernel size of (7×7) and simple morphological operation (i.e., opening), with kernel size of (3×3). The used filters may slightly affect the spatial accuracy, but visually obtain smoother boundaries and reduce the noise.

TABLE I. IMAGE DATASETS USED IN THE EXPERIMENTAL RESULTS

4D LF image dataset	**View resolution $(x \times y)$ pixels**	**Number of views**	**Thumbnail**
HCI benchmark dataset [18]: *Papillon, Monasroom, Still life, Horses and Buddha*	768×768 pixels, except for horses: 1024×576 pixels	9×9	
EPFL MMSPG LF images dataset [19]: *Friends 4, Sphynx, and Sophie and Vincent 3*	625×434 pixels	15×15	

IV. EXPERIMENTAL RESULTS

To evaluate the proposed approach, we implemented the proposed SLFS algorithm on a macOS computer with Intel i5 2.3 GHz processor and 8GB LPDDR3 memory. We used both synthetic 4D LF images [18] and 4D LF data captured with a Lytro Illum camera [19] as shown in Table I. The algorithm is implemented using Python programming language and the

open-source code for the VCLFS algorithm [20] was used to extract the 4D LF superpixels. The segmentation results are presented in Fig. 4 and Fig. 5, for synthetic and real LF images, respectively. Several parameters can affect the segmentation result, such as the superpixel size and image texture complexity. In the VCLFS algorithm, the segmentation size of x will generate average superpixel size of x^2 pixels per superpixel (assuming a square shape) [20]. In our experiments (see Fig. 4 and Fig. 5), we set the segmentation size of the VCLFS to 30, to generate an average superpixel size of 900 pixels. This size of superpixel generates consistent and accurate segmentations with a reasonable computational complexity. In Fig. 6, we changed the size of superpixels to study its effect on segmentation. Larger sizes make the segmentation faster in terms of graph-cut optimization. However, it results in inaccurate segmentation due to the larger clusters that cannot be divided. On the other hand, smaller sizes result in a more accurate segmentation, but will increase the graph size and complexity. In Fig. 6, the segmentation graph-cut takes around 8 ms, when using VCLFS with a segmentation size of 100, but it takes around 35 ms and 82 ms for a segmentation size of 30 and 15, respectively. According to the image texture, images with complex texture require more scribbles than those with non-complex texture. In Fig. 4, the *Monasroom* image presents a complex texture requiring more user scribbles and interaction than in the *Papillon* image.

To compare our results with the other 4D LF segmentation algorithms that target multi-label segmentation, we used all the published segmentation masks in [7]. We were not able to compare with the recent work in [9] since there is no published masks or open-source code for the algorithm, additionally, there is no enough implementation details to reproduce it. Furthermore, similar work targeting foreground-background segmentation has been proposed for 2D images [2], and its comparison here would be unfair due to the 4D LF structure and propagation consistency. To enable the comparison with multi-label segmentation, we considered the targeted object (e.g., the yellow horse in Fig. 7) as a foreground and other labeled objects as background, hence, binary masks from the segmentation masks in [7] and the HCI segmentation ground truth in [18] are generated instead of multi-label masks. The comparison results are displayed in Table II, we used test images and their ground truth from the HCI dataset [18].

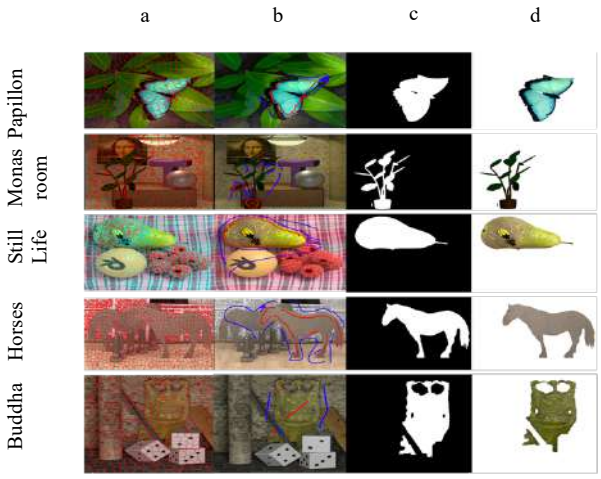


Fig. 4. SLFS results on the HCI dataset: a) central view with superpixels; b) user's foreground and background scribbles (blue for background and red for foreground); c) segmentation mask after graph-cut optimization; d) the segmented object.

Fig. 5. SLFS results on the EPFL MMSPG dataset: a) central view with superpixels; b) user's foreground and background scribbles; c) segmentation mask after graph-cut optimization; d) the segmented object.

Fig. 6. Segmentation results for different superpixel's sizes: a) size = 100; b) size = 30; c) size = 15; (larger superpixels may create inaccurate segmentation results due to the larger cluster that cannot be divided while very small superpixels improves the accuracy and increases the graph complexity).

By using the hypergraph concept with the VCLF superpixels to represent the 4D LF image, a significant reduction in graph size is achieved. For example, the *Buddha* image has 4.77×10^7 pixels, the algorithm in [7] reduced the graph size to 8.19×10^5 nodes. Additionally, the algorithm in [9] reduced the graph size to 1.46×10^3 nodes, while our algorithm reduced the graph size to only 625 nodes with similar accuracy as in Table II. Additionally, the segmentation result is consistent across views and adhere to the object's boundaries. Fig. 8, shows the consistent segmentation results, where our results show better visual consistency in some parts (e.g., the horse's hoof) compared to [7]. The VCLFS algorithm takes ~ 250s and ~273s for HCI and EPFL datasets respectively for superpixels extraction with superpixel size of 30, the graph-cut for the central view takes ~35 ms and the propagation to all LF views takes ~3s.

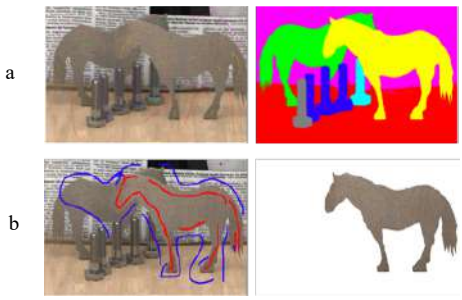


Fig. 7. Results from different interactive segmentation algorithms for *Horses* LF image: a) the state-of-the-art multi-label 4D LF segmentation result [9]; b) SLFS foreground-background segmentation result.

TABLE II. ACCURACY RESULTS FOR THE DIFFERENT ALGORITHMS UNDER ANALYSIS FOR VARIOUS 4D LF TEST IMAGES

	Results of [7]	Results of SLFS
Papillon	99.86%	99.66%
Still life	99.89%	99.87%
Horses	99.95%	99.59%
Buddha	99.57%	99.34%
Average	99.82%	99.62%

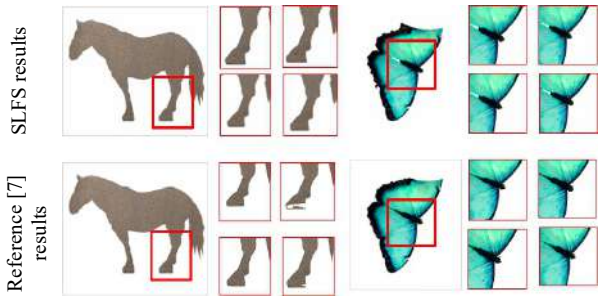


Fig. 8. Results from different 4D LF segmentation algorithms for *Papillon* and *Horses* 4D LF images. These images are selected to show the consistency across views where *Papillon* has uniform colors and *Horses* has complex texture. For each image: a red rectaglue on the central image and zoomed patches from the top-left view, top-right view, bottom-left view and bottom-right view are shown, however, all the 4D LF views are segmented.

V. FINAL REMARKS

In this paper, an improved interactive 4D LF foreground-background segmentation solution – *SLFS* – is proposed and evaluated. Firstly, the 4D LF superpixels are extracted efficiently using the VCLFS algorithm. Afterward, a hypergraph based on superpixels is created. Then, the segmentation problem is treated as an energy function optimization where a graph-cut technique is applied to optimize the segmentation result. Finally, the segmentation result is propagated to all 4D LF views consistently.

Experimental results were conducted on both real and synthetic 4D LF images and show the effectiveness of the proposed approach with comparable results even after the dramatic reduction in the graph complexity. Additionally, the experimental results show that the segmentation can be affected by the superpixel size, the image complexity and the graph-cut parameters.

The proposed approach can be used in several interesting applications where object extraction is needed, such as augmented and mixed reality, and object-based coding. For future work, the best compromise superpixel size to be used for this algorithm and the optimal parameters for graph creation and segmentation could be further optimized and will be considered. Additionally, the graph structure can be used for other LF editing applications, such as in inpainting where the space after object extraction can be filled consistently by novel pixels in one view and propagated to the 4D LF views. Furthermore, this algorithm can be further improved to include the segmentation of the sparse 4D LF images where the nodes of the large occluded objects are handled particularly in the graph creation step.

REFERENCES

[1] M. Hog, N. Sabater, and C. Guillemot, "Superrays for Efficient Light Field Processing," *IEEE J. Sel. Topics Signal Processing*, vol. 11, no. 7, pp. 1187–1199, Oct. 2017.

[2] W. Yu, Z. Hou, P. Wang, X. Qin, L. Wang, and H. Li, "Weakly supervised foreground segmentation based on superpixel grouping," *IEEE Access*, vol. 6, pp. 12269–12279, Feb. 2018.

[3] S. Wanner, C. Straehle, and B. Goldluecke, "Globally Consistent Multi-label Assignment on the Ray Space of 4D Light Fields," in *2013 IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, Portland, OR, USA, June 23-28, 2013, pp. 1011–1018.

[4] S. Jegou, M. Drozdzal, D. Vazquez, A. Romero, and Y. Bengio, "The One Hundred Layers Tiramisu: Fully Convolutional DenseNets for Semantic Segmentation," in *2017 IEEE Computer Society Conf. on Computer Vision and Pattern Recognition Workshops*, Honolulu, HI, USA, July 21-26, 2017, pp. 1175–1183.

[5] M. Levoy and P. Hanrahan, "Light field rendering," in *23rd annual conf. on Computer graphics and interactive techniques*, NY, USA, Aug. 1, 1996, pp. 31–42.

[6] H. Mihara, T. Funatomi, K. Tanaka, H. Kubo, Y. Mukaigawa, and H. Nagahara, "4D light field segmentation with spatial and angular consistencies," in *2016 IEEE International Conf. on Computational Photography (ICCP)*, Evanston, IL, USA, May 13-15, 2016, pp. 1–8.

[7] M. Hog, N. Sabater, and C. Guillemot, "Light Field Segmentation Using a Ray-Based Graph Structure," in *European Conf. on Computer Vision (ECCV)*, Amsterdam, Netherlands, Oct. 8, 2016, pp. 35–50.

[8] H. Zhu, Q. Zhang, and Q. Wang, "4D Light field superpixel and segmentation," in *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, Honolulu, HI, USA, July 21-26, 2017, pp. 6709–6717.

[9] X. Lv, X. Wang, Q. Wang, and J. Yu, "4D Light Field Segmentation from Light Field Super-pixel Hypergraph Representation," *IEEE Trans. Vis. Comput. Graph.*, early access, Mar. 2020, doi: 10.1109/TVCG.2020.2982158.

[10] N. Khan, Q. Zhang, L. Kasser, H. Stone, M. H. Kim, and J. Tompkin, "View-Consistent 4D Light Field Superpixel Segmentation," in *IEEE/CVF International Conf. on Computer Vision (ICCV)*, Seoul, Korea, Oct. 27-Nov. 2, 2019, pp. 7810–7818.

[11] 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.

[12] N. A. Ibraheem, M. M. Hasan, R. Z. Khan, and P. K. Mishra, "Understanding Color Models: A Review," *ARPN J. Sci. Technol.*, vol. 2, no. 3, pp. 265–275, Apr. 2012.

[13] M. de Berg, M. van Kreveld, M. Overmars, and O. C. Schwarzkopf, "Delaunay Triangulations," in *Computational Geometry*, 2nd ed., Berlin, Germany: Springer, 2000, pp. 183–210.

[14] C. B. Barber, D. P. Dobkin, and H. Huhdanpaa, "The quickhull algorithm for convex hulls," *ACM Trans. Math. Softw.*, vol. 22, no. 4, pp. 469–483, Dec. 1996.

[15] P. Virtanen *et al.*, "SciPy 1.0: fundamental algorithms for scientific computing in Python," *Nat. Methods*, vol. 17, no. 3, pp. 261–272, Mar. 2020.

[16] S. Kullback and R. A. Leibler, "On Information and Sufficiency," *Ann. Math. Stat.*, vol. 22, no. 1, pp. 79–86, Mar. 1951.

[17] Y. Boykov and V. Kolmogorov, "An experimental comparison of min-cut/max-flow algorithms for energy minimization in vision," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 26, no. 9, pp. 1124–1137, Sep. 2004.

[18] S. Wanner, S. Meister, and B. Goldlücke, "Datasets and Benchmarks for Densely Sampled 4D Light Fields," *VMV*, vol. 13, pp. 225–226, Sep. 2013.

[19] M. Rerabek and T. Ebrahimi, "New Light Field Image Dataset," in *8th International Conf. on Quality of Multimedia Experience (QoMEX)*, Lisbon, Portugal, June 6-8, 2016.

[20] N. Khan, Q. Zhang, L. Kasser, H. Stone, M. H. Kim, and J. Tompkin, "Repository for the ICCV 2019 paper: View-consistent 4D Light Field Superpixel Segmentation, by Khan et al." [Online]. Available: https://github.com/brownvc/lightfieldsuperpixels. [Accessed: 30-Mar-2020].

Chapter 6

Unsupervised Angularly Consistent 4D Light Field Segmentation using Hyperpixels and a Graph Neural Network

Journal: IEEE Open Journal of Signal Processing
Volume: Accepted for publication
Pages: 1-15
DOI: 10.1109/OJSP.2025.3545356

Unsupervised Angularly Consistent 4D Light Field Segmentation using Hyperpixels and a Graph Neural Network

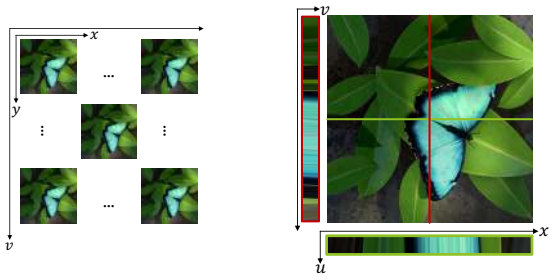
Maryam Hamad, *Graduate Student Member, IEEE*, Caroline Conti, *Member IEEE,* Paulo Nunes*, Member, IEEE* and Luís Ducla Soares*, Senior Member, IEEE*

Abstract Image segmentation is an essential initial stage in several computer vision applications. However, unsupervised image segmentation is still a challenging task in some cases such as when objects with a similar visual appearance overlap. Unlike 2D images, 4D Light Fields (LFs) convey both spatial and angular scene information facilitating depth/disparity estimation, which can be further used to guide the segmentation. Existing 4D LF segmentation methods that target object level (i.e., mid-level and high-level) segmentation are typically semi-supervised or supervised with ground truth labels and mostly support only densely sampled 4D LFs. This paper proposes a novel unsupervised mid-level 4D LF Segmentation method using Graph Neural Networks (LFSGNN), which segments all LF views consistently. To achieve that, the 4D LF is represented as a hypergraph, whose hypernodes are obtained based on hyperpixel over-segmentation. Then, a graph neural network is used to extract deep features from the LF and assign segmentation labels to all hypernodes. Afterwards, the network parameters are updated iteratively to achieve better object separation using backpropagation. The proposed segmentation method supports both densely and sparsely sampled 4D LFs. Experimental results on synthetic and real 4D LF datasets show that the proposed method outperforms benchmark methods both in terms of segmentation spatial accuracy and angular consistency.

***Index Terms*— Light field, unsupervised segmentation, deep learning, angular consistency, graph neural network**

I. Introduction

Light field (LF) imaging has attracted increasing attention from researchers due to the rich information it includes and its potential for immersive applications [1], [2]. LFs contain information about both the intensity and direction of light rays and can be represented as an array of views captured from different perspectives. To represent that array of views, a 4D function $I(x, y, u, v)$ can be used, where (x, y) and (u, v) are, respectively, the spatial and angular coordinates of each view. By fixing one angular and one spatial coordinate, an Epipolar Plane Image (EPI) (i.e., the unique 2D spatio-angular LF slice typically containing a regular structure with several slanted lines [1]) can be obtained, which corresponds to the depth/disparity cues, as presented in Fig. **1**. Depth/disparity cues in 4D LFs can help improve different computer vision tasks, such as in scene segmentation, by using these cues as a discriminative feature, notably, when visual information alone is not sufficient.



(a) (b)

Fig. 1. In 4D LFs, each LF view (i.e., a slice of 4D LF in a particular angular plane (u, v)) captures the scene from a different view perspective as in (a). This results in shifted light rays across views as can be seen in the EPIs with green and red borders, shown below and to the left of the central view in (b).

Image segmentation is a fundamental task that aims at dividing image data into perceptual and homogenous regions according to specific criteria. By segmenting an image, we can isolate and identify individual components or objects, which is essential for several applications, such as image compression, object detection, autonomous driving, medical imaging and scene understanding [3]. Image segmentation, in 2D images, has been widely investigated with different solutions including traditional methods, e.g., clustering and graph-cut optimization techniques [3], or deep learning-based methods [3]. Most of the deep learning-based 2D

This work is funded by FCT/MECI through national funds, and when applicable co-funded by EU funds, under UID/50008: Instituto de Telecomunicações and under project PTDC/EEI-COM/7096/2020 (https://doi.org/10.54499/PTDC/EEI-COM/7096/2020).

M. Hamad, C. Conti, P. Nunes and L. D. Soares are with Instituto Universitário de Lisboa (ISCTE-IUL), Instituto de Telecomunicações, Lisboa, Portugal (e-mail: {maryam.hamad, caroline.conti, paulo.nunes, lds}@lx.it.pt).

This paper has supplementary downloadable material available at https://github.com/MaryamHamad/LFSGNN, provided by the author. The material includes videos for the dynamic results. Contact maryam.hamad@lx.it.pt for further questions about this work.

image segmentation methods are supervised, relying on Ground Truth (GT) label images. Since generating pixel-wise annotations for large datasets can be labor-intensive and costly, the development of fully unsupervised methods or the fine-tuning of pre-trained foundation models that have been trained on large datasets to extract deep features for image segmentation tasks became a growing research direction with promising performance [3], [4].

Although 2D image segmentation is an active research area, 4D LF segmentation remains relatively unexplored, with additional challenges and performance requirements to be considered. While segmentation accuracy is important in 2D images, segmentation angular consistency in 4D LFs is also essential. More precisely, when segmenting 4D LFs, the corresponding pixels across all LF views must have the same segmentation label. Otherwise, the sudden label changes when navigating through the views can lead to unwanted flickering. Coupled with the huge amount of data involved, and the lack of 4D LF segmentation datasets for training and evaluation, this makes 4D LF segmentation a more complex task than conventional 2D image segmentation.

Existing 4D LF segmentation methods can be categorized into three main categories according to the level of the semantic meaning of the obtained segments (as detailed in Section II): i) Low-level unsupervised over-segmentation methods, where similar pixels are grouped into perceptually meaningful atomic regions, without the need for label annotations or user scribbles, e.g., [5]–[10]; ii) Mid-level semi-supervised segmentation, where the semantic labels of the segmented objects are not included, e.g., [11]–[15]; and iii) High-level supervised semantic segmentation methods, e.g., [16]–[18], where semantic labels are also predicted for each pixel. This paper focuses on achieving mid-level multi-label segmentation in a fully unsupervised manner using a deep-learning approach.

Low-level and mid-level 4D LF segmentation methods typically rely on classical or basic machine learning techniques. On the other hand, high-level segmentation methods adopt deep learning techniques for training. Most deep learning-based 4D LF segmentation methods are applied only to the central view without considering other objects in the side views. However, when using sparse LFs, such as in immersive applications, other LF views, where disocclusions and additional objects may exist, must be considered. Moreover, the available deep neural networks for high-level semantic segmentation are often supervised, and, thus, inevitably demand pixel-wise GT segmentation labels for the training [16], [17], [19], which are challenging to obtain for all LF views, especially for real world LF datasets. Nevertheless, the use of deep learning has shown promising results in supervised 4D LF semantic segmentation and also in weakly-supervised and unsupervised 2D image segmentation [20]. Therefore, fully unsupervised 4D LF segmentation methods using deep learning are becoming increasingly appealing. The reason is that it may help in extracting relevant features from the LF and reduce the effort of manually defining precise features, which can be quite challenging. Another possible approach is to adapt pre-trained 2D foundation models, e.g. [21], for 4D LF data for extracting deep features and exploiting them for segmentation tasks. Pre-trained 2D foundation models capture rich semantic information from large-scale data, thus, by exploiting the pre-trained knowledge, zero-shot or few-shot image segmentation can be achieved. However, an adaptation is needed when applied for 4D LF data to consider ensuring angular consistency constraints. Applying unsupervised mid-level segmentation can overcome the GT availability limitation by learning deep features from the input itself and enabling segmentation based on intrinsic features. Moreover, pseudo-segmentation labels for segmentation are often generated in mid-level unsupervised segmentation, starting with many classes, and then the pixel labels and feature representations are jointly optimized by updating the network parameters using gradient descent.

To sum up, this paper aims to overcome the main limitations in most existing (mid/high-level) 4D LF segmentation methods, namely: i) Relying on the user scribbles or supervision; ii) Only supporting densely sampled 4D LFs; iii) Only applying segmentation to the central view; and iv) Not adequately exploiting LF view correlation or ensuring angular consistency across LF views. Accordingly, the main contributions of this paper are:

- **Proposal of a novel unsupervised angularly consistent 4D LF segmentation method for dense and sparse LFs** – In this paper, 4D LFs are segmented into (mid-level) objects without any prior supervision or user scribbles. To the best of the authors' knowledge, this is the first (mid-level) 4D LF segmentation technique that exploits deep features to segment objects without supervision for both dense and sparse LFs. Additionally, the segmentation is applied simultaneously to all LF views that compose a 4D LF, ensuring angular consistency throughout. This is achieved by initially over-segmenting 4D LFs into hyperpixels (where corresponding pixels across LF views are grouped according to their similarity in terms of color/texture, position and depth/disparity into the same hyperpixel) using the method proposed by the authors in Hamad *et al.* [10] to provide a compact LF representation.
- **Use of Graph Neural Networks (GNNs) for 4D LF segmentation** – In this paper, to efficiently deal with the large amount of 4D LF data, a novel hypergraph representation based on 4D LF over-segmentation is used. To exploit the advantage of deep learning techniques, a GNN is used on the graph-structured 4D LF. While GNNs have shown promising results in node classification and 2D image segmentation, to the best of the authors' knowledge, this is the first time a GNN has been applied to unsupervised 4D LF segmentation.
- **Proposal of a set of complementary metrics for evaluating segmentation angular consistency** – Although both spatial accuracy and angular consistency should be considered when evaluating 4D LF segmentation methods, existing 4D LF mid/high-level

segmentation methods are often only evaluated in terms of spatial accuracy. Therefore, this paper proposes a set of complementary metrics that together enable evaluating the segmentation angular consistency, to be used in addition to spatial accuracy evaluation metrics. These metrics can be used for both dense and sparse LFs.

The remainder of the paper is organized as follows. Section II briefly reviews the related work on 4D LF segmentation. Section III describes the proposed unsupervised 4D LF segmentation method. Section IV presents the proposed segmentation angular consistency metrics. Section V includes experimental results to evaluate the proposed method. Finally, Section VI concludes the paper with final remarks and directions for future work.

II. Related Work

In the past decade, several proposals have been made for 4D LF segmentation, which can be categorized into three main categories depending on the level of semantic meaning of the obtained segments, as briefly reviewed in this section:

Low-level unsupervised over-segmentation methods, e.g., [5]–[10], mainly group pixels into atomic regions, which share similar characteristics, e.g., color/texture, position and depth/disparity, without the need for label annotations or user scribbles. These regions are often used as a pre-processing step for subsequent tasks. Available low-level 4D LF over-segmentation methods can be classified as either clustering-based or graph-based, depending on the approach used to divide 4D LFs into homogeneous regions. In the case of clustering-based methods, the *K*-means clustering algorithm is often used. *K*-means is usually applied to all 4D LF views with different approaches, such as starting *K*-means clustering in the central view and applying label propagation into all other LF views, as in [6], [7], or applying *K*-means clustering for the entire 4D LF as in the hyperpixels method [10]. In the case of graph-based methods, the 4D LF is represented by a weighted undirected 4D graph where each pixel is considered as a graph node, as proposed by Li *et al.* [8]. Afterwards, LF over-segmentation is achieved by maximizing the graph entropy in the 4D LF domain. While 4D LF over-segmentation can be achieved using graph techniques, applying graph optimization on a huge number of pixels requires extensive computational resources.

Mid-level semi-supervised segmentation methods group the pixels into objects without including semantic labels, e.g., [11], [14], [15]. In this case, user scribbles are usually inserted in the central view and the entire LF views are segmented accordingly [11]–[15]. For mid-level 4D LF segmentation, a common approach is to represent the 4D LF as a graph and apply classical graph-cut optimization assisted by the user scribbles (a.k.a. semi-supervised or scribble-supervised segmentation). However, representing each light ray as a graph node leads to a huge number of nodes, and thus can increase the processing complexity [12]. To reduce the number of graph nodes, corresponding pixels across LF views that represent the same 3D point (a.k.a. a ray bundle) are represented by a graph node [13]. To further reduce the graph size, the 4D LF can be represented by a hypergraph by exploiting the spatio-angular correlation across views [14], [15]. To achieve that, low-level 4D LF over-segmentation is first applied. Then, a hypergraph is created where 4D segments (i.e., corresponding pixels in all views that locally share similar criteria and represent the same 3D region) are represented by a hypernode. Although these methods reduce the graph size significantly, the over-segmentation methods they use to create hypergraphs are only suitable for dense LFs but not adequate for sparse LFs with large occlusions. Mid-level unsupervised segmentation methods can also be found in the literature, e.g., [22], [23], for specific applications such as transparent object segmentation and soft color segmentation, which are out of the scope of this paper.

High-level supervised semantic segmentation methods, e.g., [16]–[18], also predict semantic labels for each pixel. However, due to the lack of available LF datasets with GT segmentation labels for training deep neural network models, this has been a challenging research field in the past. New datasets for LF semantic segmentation have been proposed recently to support this research direction [17], [19], enabling the use of deep learning for this task [19], [17], [16], [18], [24]. To achieve supervised semantic segmentation, LF datasets with label annotations are required for training and evaluation. Therefore, a dataset with 400 real world LFs annotated for three foreground objects was created by Jia *et al.* [19] to train a Convolutional Neural Networks (CNN) based model. Later, Shen *et al*. [17] proposed a new dense 4D LF dataset for urban scenes (UrbanLF) annotated for 14 semantic classes. After the UrbanLF dataset was published, various 4D LF supervised semantic segmentation methods were proposed for urban scenes, e.g., [16], [18], [24]. Existing methods in this category can segment only specific objects (e.g., cars, buses and people). Additionally, they rely on supervision using GT segmentation labels of densely sampled LFs. As the exploitation of weakly-supervised approaches to achieve high-level semantic segmentation shows promising results in 2D images [20], adapting these approaches for 4D LF could reduce the reliance on expensive and time-consuming fully annotated data.

III. Proposed 4D Light Field Segmentation Method

This paper proposes an unsupervised and angularly consistent mid-level 4D LF segmentation method for both dense and sparse static LFs. Given a 4D LF, the proposed method consists of four main steps, as summarized in Fig. 2. Each step is detailed in the following subsections. It is worth noting that the first two steps are considered pre-processing steps using existing methods, and the third and fourth steps include the contributions of this paper:

1) Disparity Estimation – Angularly consistent disparity maps are estimated for all LF views using an efficient disparity propagation method [25].
2) 4D LF Over-segmentation – The 4D LF is over-segmented into hyperpixels, which are consistent over the entire LF [10].

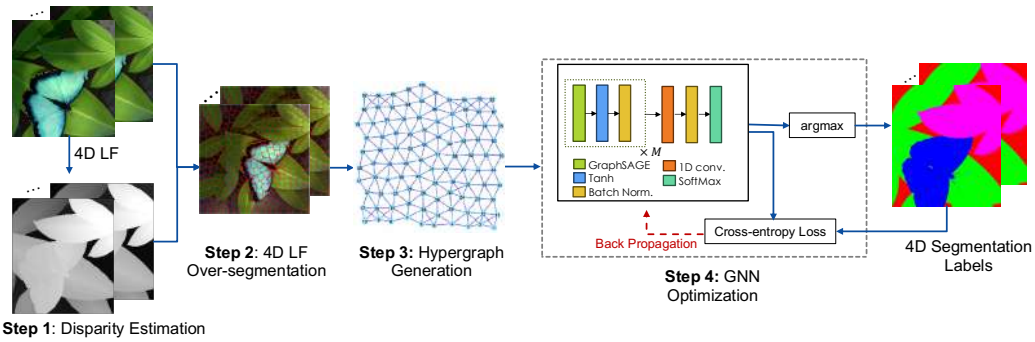


Fig. 2. Main steps of the proposed 4D LF segmentation method. Given a 4D LF, the corresponding disparity maps for all views are initially estimated (Step1), next, the 4D LF is over-segmented into hyperpixels (Step2); after that, a hypergraph is generated, where each hyperpixel is represented as a hypernode (Step3); finally, a GNN optimization is performed in an unsupervised manner to obtain the 4D segmentation labels (Step4).

3) Hypergraph Generation – Once the hyperpixels are obtained, the 4D LF is represented as an undirected hypergraph, where each 4D hyperpixel is represented by a hypernode and two neighboring hyperpixels are connected by a hyperedge. Each hypernode is represented by a feature vector.
4) GNN Optimization – Finally, using the hypergraph as input, a GNN model is initialized and iteratively optimized, generating this way an unsupervised 4D LF segmentation, i.e., assigning a label for each hyperpixel, without any annotation effort or using user scribbles for supervision.

In this paper, we consider 4D LFs, however, the proposed method can be adapted and applied to other imaging modalities that can be represented by a graph (e.g., point clouds) and the GNN model may be adapted as well for other LF applications, such as LF inpainting and color editing.

A. Disparity Estimation

As shown in Fig. 2 (Step 1), initially, disparity maps are estimated for all 4D LF views (with respect to its adjacent right view). Disparity information is inversely related to object depth and represents the difference in position of the same 3D point between two views. Therefore, it is a rich feature to guide the segmentation in terms of reducing edge ambiguity, especially when objects have similar colors or texture but different depths. Integrating disparity with other features during the segmentation helps deep learning models to learn spatial structures, leading to better performance, namely in terms of segmentation accuracy and angular consistency. The accuracy of the used disparity maps can affect the subsequent steps in terms of accuracy and angular consistency. Therefore, in this paper, to ensure disparity map angular consistency (i.e., corresponding pixels across views that represent the same 3D point must have the same disparity value), the disparity map of the central view is computed first using the method proposed by Shi *et al.* [26]. After that, the disparity of all other LF views is consistently propagated using the proposed disparity propagation method in [25]. For this step, any disparity estimation method that generates angularly consistent disparity maps for all views can be used.

B. 4D LF Over-segmentation

As shown in Fig. 2 (Step 2), given the input 4D LF and the estimated disparity maps for all LF views, 4D LF over-segmentation is applied as a pre-processing step using the proposed method in Hamad *et al.* [10] to generate "hyperpixels". This step is useful for the proposed LFSGNN method since it handles the spatial shifts across LF views, due to the viewing angle, by grouping corresponding and similar pixels into hyperpixels in both dense and sparse LFs. As mentioned above, 4D LF over-segmentation into hyperpixels enables generating a more compact graph representation and reduces the number of nodes significantly (e.g., compared to using each pixel/light ray as a graph node). Moreover, using regular square segments of an image (a.k.a batches) to represent graph nodes as proposed in [4] may result in non-smooth borders; instead, it is more robust to use homogenous regions that adhere well to object boundaries and ensure angular consistency across all LF views. Since applying traditional 2D over-segmentation to each LF view independently will not ensure angular consistency, 4D hyperpixel over-segmentation is adopted in this paper [10].

The 4D hyperpixel over-segmentation method is used since it outperforms other existing 4D LF over-segmentation methods in terms of spatial accuracy and angular consistency [10]. Moreover, it enables a flexible and adaptive over-segmentation over the entire 4D space for both dense and sparse LFs. It is shown in [10] that using accurate disparity maps improves the 4D hyperpixel over-segmentation and enables better adherence to the object boundaries, subsequently, improving the final mid-level segmentation, as explained in Section V. The default hyperpixel size in [10] (i.e., 20) is used when applying 4D LF over-segmentation.

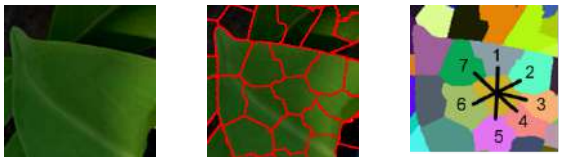
Each hyperpixel is represented by different features including color, texture, and disparity of the original 4D LF. Both RGB and CIELAB color spaces are used to represent the color feature since it has been shown that there is an advantage in combining those two color spaces [27]. With respect to the CIELAB color space, in this paper, only the chromatic components are used (i.e., a and b channels) to reduce the impact of variations in illumination. Moreover, a texture feature is computed using the Local Binary Pattern

(LBP) texture descriptor [28] over the grayscale version of the original RGB 4D LF[1]. The LBP computes the local variation of pixels with high discriminative power and robustness to illumination changes, being a widely adopted effective approach in many computer vision applications. The LBP adopted in this paper uses up to 256 unique patterns and the texture descriptor of each hyperpixel is the pixel histogram over these 256 bins. Moreover, for the RGB, CIELAB and disparity features, the arithmetic mean values computed for each hyperpixel are used as hyperpixel features. Thus, this results in a 262-dimension feature vector, $h = [\bar{R}, \bar{G}, \bar{B}, \bar{a}, \bar{b}, \bar{D}, (LBP_{hist})_{\times 256}]$, for each hyperpixel. Given the differences in feature ranges, all features are normalized to $[0, 1]$. Normalizing these features will be useful later for the GNN optimization step, ensuring stable learning and proper optimization.

C. Hypergraph Generation

As shown in Fig. 2 (Step 3), the hyperpixels that were computed in the previous step are used to generate a hypergraph. Each hyperpixel is represented as a hypernode of a hypergraph denoted by $G = (v, \varepsilon)$, where v represents the set of hypernodes and ε represents the set of undirected and unweighted hyperedges that represent the adjacent relationship between the nearest neighboring hyperpixels. Every hypernode is, therefore, represented by a 262-dimension feature vector as explained in the previous step. In this paper, all hyperpixels that share a common boundary are considered neighbors, as illustrated in Fig. 3.

The hypergraph, hypernode and hyperedge concepts are used in this paper since a hyperpixel typically has a 2D slice in some or all LF views. Hence, each slice can be considered as a classical graph node. However, since all hyperpixel slices represent the same hyperpixel, using the hypernode concept enables a more compact 4D LF representation. The hypergraph generation is flexible to any input resolution and the number of hypernodes may differ from one LF to another according to the input resolution and hyperpixel size. Therefore, there is no explicit adjustment needed to handle different 4D LF datasets. The hypergraph generated in this step is used as input to the GNN model, as detailed in the next step.



(a) (b) (c)

Fig. 3. Example of a single hyperpixel neighbors (only a 2D slice is shown): a) Part of the original LF; b) Corresponding hyperpixels boundaries; c) Set of neighbors that share common boundaries with a given hyperpixel. This is illustrated for a single hyperpixel but applies to all hyperpixels.

D. GNN Optimization

The main goal of the proposed method is to classify all pixels in a 4D LF into an arbitrary number of classes c ($C_{min} \leq c \leq C_{max}$), where C_{min} is the minimum number of classes/labels and C_{max} is the initial maximum number of classes/labels, as detailed in Section V-B. To do so, the proposed method uses a GNN to achieve an unsupervised 4D LF segmentation by merging the initial hyperpixels and labeling the 4D LF with c unique labels based on the LF content.

1) Graph Neural Network – A GNN is a neural network designed to process graph-structured data. The key idea behind GNNs is to enable each node to aggregate information from its neighbors through edges. Therefore, GNNs can capture complex dependencies within the graph data at different levels of abstraction. Each layer of a GNN typically comprises two primary operations: message passing and aggregation. In the message passing operation, node information is gathered and exchanged between neighboring nodes. In the aggregation operation, all the information gathered from the previous operation is fused for each node into one message to update its current state. GNNs have shown appealing performance in various applications, including node classification (e.g., in social networks and multi-label image segmentation). In this paper, the inductive and scalable Graph Sampling and Aggregation (GraphSAGE) framework is adopted which is widely used for node classification [29]. The key idea of GraphSAGE is to generate a node embedding (i.e., a low-dimensional vector representation of nodes in a graph) by learning an aggregation function from the representation of its neighbor nodes. The reason for using GraphSAGE lies in its ability to effectively capture local structural information of graph nodes and its scalability to process large-scale graph data and handle high-dimensional feature spaces [29]. Moreover, it enables sampling only a subset of neighboring nodes (which can be randomly selected or by using other advanced methods) to conduct propagation instead of using all the neighborhood information. This can help in reducing the computational complexity and makes the model less likely to overfit to specific structures in the training data. In GraphSAGE, the message passing operation can be considered a generalization of the traditional CNN on regular grids, where the convolution operation is replaced by the aggregator function. The mean aggregator (a.k.a. convolutional aggregator) is used in this paper, which processes each node in the graph as formulated in (1):

$$h_v^k = W_1 h_v^{k-1} + W_2 \sum_{u \in \mathcal{N}(v)} \frac{h_u^{k-1}}{|\mathcal{N}(v)|}, \quad (1)$$

where h_v^k is the current state of node v, h_v^{k-1} and h_u^{k-1} are the previous states of node v and u, respectively, $\mathcal{N}(v)$ is the set of neighbors of node v ($|\mathcal{N}(v)|$ represents the number of its neighbors), and W_1 and W_2 are matrices of the network parameters that need to be optimized. The main objective during training is to optimize W_1 and W_2 to make the node representations as

[1] https://docs.opencv.org/3.4/de/d25/imgproc_color_conversions.html

informative and predictive as possible, based on their local graph structure. Optimizing W_1 and W_2 leads to minimizing the final loss through backpropagation.

2) Proposed GNN Model Architecture – The high-level architecture of the proposed model is shown in Fig. 2 (Step 4) and contains M consecutive components, each of which contains a GraphSAGE operator, a hyperbolic tangent activation (tanh) function, and a batch normalization function. The batch size corresponds to one full LF hypergraph. The input of the proposed model is one matrix with the features of all hypernodes and the adjacency list (i.e., a list of neighbors for each hyperpixel). The size of the hidden channels of all GraphSAGE operators, as well as the output channels, is set to s. Therefore, s-dimensional feature maps are computed from the hypergraph G. The value of s is set in this paper to the same value as C_{max}. Afterwards, the SoftMax operator is applied to the model output to obtain the probability distribution over predicted output classes. Since GT segmentation labels are not used in the proposed method, pseudo-segmentation labels are obtained using the argmax operator to find the dimension with maximum probability. The maximum probability value for each hypernode is selected and, when applying argmax for all hypernodes, the pseudo-segmentation labels are obtained. These pseudo-segmentation labels are used as target labels in the loss function as shown in Fig. 2. The loss is then computed between the model output and the pseudo-segmentation labels that are obtained in each epoch. For the loss function, L, the cross-entropy loss is used since it is effective and widely used in multi-label classification.

$$L(q,i) = -\sum_{c=1}^{C_{max}} \log \frac{\exp(q_c)}{\sum_{j=1}^{C_{max}} \exp(q_j)}, \quad (2)$$

where q_c represents the probability of the pseudo-segmentation label, c, and q_j denotes the probability of the j^{th} class and $i, j \in \{1,2,\ldots,C_{max}\}$. The unsupervised 4D LF segmentation is achieved by applying both forward and backward passes with respect to a loss function to optimize model parameters. In the forward pass, the segmentation labels are predicted using fixed network parameters. However, in the backward pass, the network parameters are trained with fixed-label predictions as in [30]. The error signal is finally backpropagated to update the learnable parameters, which are initialized by default with Kaiming He initialization [31]. The model is iteratively trained until the maximum number of epochs or the minimum number of labels, C_{min}, is reached. Finally, the predicted labels for the hypernodes are then mapped back to represent the 4D segmentation map. This is achieved by assigning the predicted label of the corresponding hypernode to all pixels in 4D space that belong to the corresponding hyperpixel.

IV. Segmentation Angular Consistency Metrics

Different from conventional 2D image segmentation where only the segmentation accuracy is important, in 4D LF segmentation angular consistency must also be considered.

To evaluate the segmentation angular consistency, we used the Labels per Pixel (LP) metric that was proposed to evaluate angular consistency for LF over-segmentation [7]. Initially, $LP = 1$ for all pixels in the central view (i.e., one unique label in the central view). To compute the LP metric, all LF views are warped into the central view using GT disparity maps. Afterwards, for each pixel position, the number of labels that have different values than the label in the central view is counted and then added to the initial LP value. Then, the $\overline{LP}$ value for all pixels in the central view is computed (higher value indicates worse angular consistency). However, the LP metric is adequate for dense LFs only, since when warping the views, all pixels in off-central views that are not seen in the central view due to the viewing angle are discarded. Hence, it is not adequate for sparse LFs with a large disparity range.

Therefore, inspired by the LP metric, we propose a set of complementary metrics to evaluate the 4D LF segmentation angular consistency for both dense and sparse LFs: i) Segmentation Angular Consistency (SAC); ii) Percentage of Inconsistent Pixels (IP); and iii) Average Local LP for Inconsistent Pixels ($\overline{LLP}_{IP}$). Each metric is explained below. All the proposed metrics rely on computing LP in a local window. To achieve that, LF views are warped into a locally central view within a local window of views (i.e., 3×3 views) and then the LP is computed locally, termed Local LP (LLP) in this paper. To consider all local windows in a 4D LF, this process is repeated by sliding the window one angular position each time and computing the LLP for each window. To ensure accurate warping and adequately consider occlusions, we project a pixel from each view into the window's central view only if both have the same GT segmentation label. Moreover, as proposed in [7], when pixel overlapping occurs during the warping (i.e., projecting pixels of different objects from the off-central view into the same pixel position in the target view due to an occlusion), the foreground pixel is considered (i.e., the one with highest disparity). After warping all views of the window into the window's central view, the LLP metric is computed for each local window and the average LLP, $\overline{LLP}$, is calculated for all windows in the LF.

After computing the $\overline{LLP}$, the SAC metric is computed as formulated in (3). In a local window, N, represents the total number of views, which also corresponds to the maximum possible LLP value:

$$SAC = \frac{N - \overline{LLP}}{N - 1}. \quad (3)$$

This metric measures the segmentation angular consistency of light rays, where a higher value indicates better angular consistency. For example, if all views in a local window of $N = 3 \times 3$ have the same segmentation labels for corresponding light rays, i.e., $\overline{LLP} = 1$, then the SAC will have its highest value, i.e., $SAC = 1$, regardless of the N value; on the other hand, if each LF view has a different label

for the same light ray (worst case scenario), then $\overline{LLP} = N$ and $SAC = 0$, which implies no angular consistency.

As a complement to the SAC metric, IP and $\overline{LLP}_{IP}$ are computed in this paper to highlight the percentage of inconsistent pixels across LF views (i.e., pixels where $\overline{LLP} > 1$) and the average LLP in those inconsistent pixels, respectively. As can be seen in Fig. 4, larger window sizes show more inconsistent pixels, especially in sparse LFs. The influence of using different window sizes on the proposed metrics is presented in Section V-E.

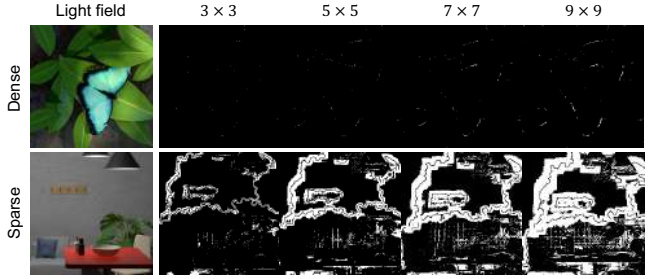


Fig. 4. Visualization of the LLP for two LFs central views, where white pixels indicate pixel positions with $\overline{LLP}$>1, using different window sizes.

V. Experimental Results

In this section, the proposed 4D LF Segmentation method using a GNN, from here on simply called LFSGNN, is evaluated both quantitively and qualitatively. Different 4D LF datasets are used in our experiments, including dense and sparse, synthetic, and real world LF datasets. Moreover, to evaluate the segmentation results in terms of spatial accuracy and angular consistency, different metrics that rely on the availability of the GT segmentation labels and disparity maps are considered. Since the real LF dataset does not have GT segmentation labels or disparity maps, only visual results are presented in this paper for this dataset. Regarding the segmentation angular consistency, in this paper, only the central view and central EPIs are shown to illustrate the angular consistency. However, to be able to observe the angular consistency across all LF views, we highly encourage the reader to observe the dynamic results, where LF views are scanned in serpentine order and presented as videos, in the supplemental materials available online for all test LFs[2].

A. 4D LF Datasets

The proposed unsupervised 4D LF segmentation method does not target a specific domain (e.g., urban scenes or medical images). Therefore, any 4D LF dataset can be used to evaluate the proposed method. However, to quantitatively evaluate the segmentation accuracy and consistency, the GT disparity maps and GT segmentation labels for all LF views are needed. Therefore, the synthetic HCI [32] and IT-4DLF [33] datasets are used since they provide the GT disparity maps and segmentation labels for all views of the dense and sparse LFs, respectively. Moreover, to validate our results on real world LFs, the MMSPG dataset captured with a Lytro Illum camera [33] is used, considering the central 9×9 views to eliminate the vignetting effects (i.e., saturation or darkening at the edges of a lenslet image compared to the center). It is worth noting that the 4D LF datasets designed for supervised semantic segmentation, summarized in [17] are not adequate for evaluating the proposed method. The reason is that multiple objects with different visual appearances may be classified to the same semantic label (e.g., blue and red cars have the same semantic label), which is not the case envisaged in the proposed method. A summary of the used LF datasets can be found in Table I.

TABLE I
LIGHT FIELD DATASETS USED IN THE EXPERIMENTAL RESULTS

4D LF dataset	View resolution in pixels ($N_x \times N_y$)	Number of views ($N_u \times N_v$)	Disparity range
HCI dataset [32]: Papillon, Buddha, , StillLife and Horses	768×768 except for Horses: 1024×576	9×9	[-4, 4]
MMSPG dataset [33]: Poppies and Swans	625×434	15×15	[-1, 1]
IT-4DLF dataset [10]: Kitchen, Room and Antique	512×512	9×9	[-18, 18]

B. Implementation Details

Firstly, the proposed method does not require splitting datasets for training and testing since it is unsupervised, and the learning parameters are optimized for each LF independently. Moreover, since hypergraphs are used to represent LFs, hypergraph generation is flexible to different input resolutions. In this paper, all experiments were run on a desktop computer with a 64-bit Ubuntu operating system, AMD® Epyc 7282 16-core CPU, NVIDIA GeForce RTX 3090 and 256 GB RAM. Our network is implemented using Pytorch (2.1.1) and the network is optimized using a Stochastic Gradient Descent (SGD) optimizer with an initial learning rate of 0.05. Momentum and weight decay are set to 0.9 and 0.01, respectively. The learning rate scheduler decays the learning rate by multiplying it by 0.95 every 50 epochs. The maximum number of epochs is set to 1000. The number of M components of the network is set to 2. The maximum number of classes, C_{max}, is set to 128 as a large number to start the segmentation process, and the minimum number of labels, C_{min}, is set to 5 as a reasonable number of objects in the test datasets.

C. Benchmark Methods

As there are currently no available methods that target fully unsupervised 4D LF mid-level segmentation, we compare our results with state-of-the-art unsupervised 2D image segmentation methods applied on 4D LF content without changing their model architectures. The first method is proposed by Kim *et al.* [30], which is fully unsupervised and adopts a conventional 2D CNN to extract deep features.

[2] https://github.com/MaryamHamad/LFSGNN

Afterwards, segmentation labels are assigned according to the response vector using an argmax function. Then the segmentation labels are used as pseudo-segmentation labels to compute the final loss. Finally, image segmentation is achieved by iteratively minimizing the loss function until a maximum number of epochs or the minimum number of labels is reached. The second method is proposed by Aflalo *et al*. [4], in which deep features are extracted from an available pre-trained vision transformer. The used transformer divides an image into square patches (patch size is constant as detailed in [4]) where each patch represents extracted features. Those patches are then used to represent an image as a graph, where each patch represents a graph node. The created graph is then input to a lightweight GNN model with one graph convolutional layer to apply unsupervised segmentation for 2D images. Although this method exploits an existing pre-trained model to extract the features, the idea of representing an image as a graph based on local regions in the image and applying the GNN technique makes it directly related to the proposed method.

Both above-mentioned methods are designed for 2D images. To use them on 4D LF data, they are applied to each LF view independently. However, to promote view consistent segmentation, for each method, the same initial values for all training parameters are used for each LF view and the random values generated using a fixed seed to ensure reproducibility for each LF view. This ensures similar behavior for feature extraction according to the used initialization of the training parameters. Moreover, the benchmark methods were executed using their available Pytorch implementation and, for a fair comparison, the minimum number of segmentation labels is set to 5 for all methods.

This paper did not make comparisons with the high-level supervised 4D LF segmentation methods since they rely on segmenting objects according to their semantic labels, which is not the case considered in the proposed method.

D. Evaluation Metrics

The segmented 4D LFs are evaluated, in this paper, in terms of spatial accuracy and angular consistency considering all views. As explained in Section V, the LP and SAC metrics are used to evaluate the non-local and local angular consistency, respectively, for both dense and sparse LFs. To evaluate the segmentation accuracy, the mean Intersection over Union (mIoU) metric is used, which is a common metric widely used to evaluate mid-level segmentation accuracy. The mIoU metric measures the amount of overlap between the GT and predicted segmentation labels. Since the proposed method is unsupervised, the predicted segmentation labels are not necessarily the same as the GT segmentation labels in terms of their values or number (this is valid for all benchmark methods). Therefore, to calculate the mIoU metric, for each label in the GT label images, the largest overlapping between that label and all the predicted labels in the corresponding location is considered, as described in [30].

E. Influence of Disparity Maps Quality on the Proposed Method

To highlight the importance of the disparity map quality on the segmentation accuracy and angular consistency, the proposed method is tested by using estimated and GT disparity maps (for both the proposed method and the hyperpixels over-segmentation on which it relies).

To evaluate the angular consistency, the proposed SAC, IP and $\overline{LLP}_{IP}$ metrics are computed considering different values of N (i.e., 3×3, 5×5, 7×7 and 9×9) to explore the window size influence. As presented in Table II, the proposed metrics are influenced by the window size for both dense and sparse LFs due to the maximum possible number of labels in a window of views and the discarding of occluded or non-existent (due to the viewing angle) pixels in the central view when computing the LLP. Additionally, knowing that ideally, the GT segmentation labels have $\overline{LLP} = 1.0$ and $SAC = 1.0$ for a given N helps in identifying how close the angular consistency of the predicted labels is to the GT labels. In all upcoming experiments (if N value is not specified), the metrics are computed for $N = 3 \times 3$; to reduce discarding pixels that are occluded or non-existent in the window central view, especially for sparse LFs.

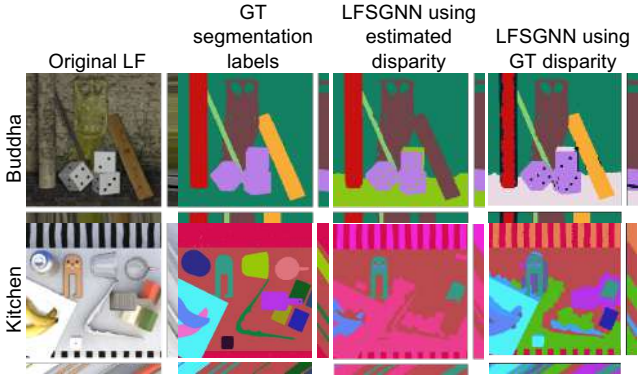


Fig. 5. Examples of the influence of disparity map quality on the proposed method. The estimated disparity maps have smoother borders between objects and hence can merge different objects easier and faster especially when the C_{min} is less than the number of objects in the scene.

As has been demonstrated in [10], more accurate disparity maps can positively affect the hyperpixels angular consistency. Since the proposed method relies heavily on the hyperpixels, the quality of the used disparity maps also influences the segmentation performance in terms of accuracy and angular consistency as shown in Table II, Table III, Table IV and Fig. 5. As can be noticed from Table II, Table III and Table VI the proposed method in most cases achieves higher angular consistency when using more accurate disparity maps (e.g., GT disparity maps in this experiment), which indicates that most of the pixels in the central view of a given window have the same label across all views of that window. For sparse LFs, such as the Room LF (which contains large occlusions), the accuracy of the used disparity map significantly affects the segmentation angular consistency, as shown in Table II and Table III (in this paper, bold style indicates better performance for all tables). Notice that in some cases (as in StillLife in Table II),

the angular consistency metrics show better performance when using estimated disparity maps. The reason for this is the smooth disparity values in estimated disparity maps can merge objects easier and faster (i.e., ends up segmentation with a fewer number of labels) compared to using sharp GT disparity maps, thus reducing the unique labels and warping error when computing the segmentation angular consistency metrics. Moreover, for the same reason, the $\overline{LLP}_{IP}$ is higher when using GT disparity maps in most LFs.

F. Ablation Study

Before comparing our results to the benchmark methods, an ablation study to investigate different configurations of the proposed method is presented. Initially, to study the influence of using the texture feature, the proposed method is tested with and without using the LBP texture descriptor in hypergraph generation and during the GNN optimization. As shown in Fig. 6, by incorporating texture features, the learning process in most LFs converged faster to C_{min} value, in contrast to when the texture feature was not used (converges slower, did not approach the C_{min} value and stopped based on the number of epochs). This can be noticed in Fig. 6 where the segmented LF has more labels than C_{min} when the texture feature was not used. Hence, it was stopped due to the epoch criteria being reached first. The rationale behind that is that the texture feature helps the model to learn a meaningful representation of the LF, and the evaluation metrics reflect that numerically, as in Table V. The reason for Kitchen and Room achieving better accuracy when the texture feature is omitted is that the final segmented LF has more unique labels compared to when using the texture, which means considering most objects in the LF.

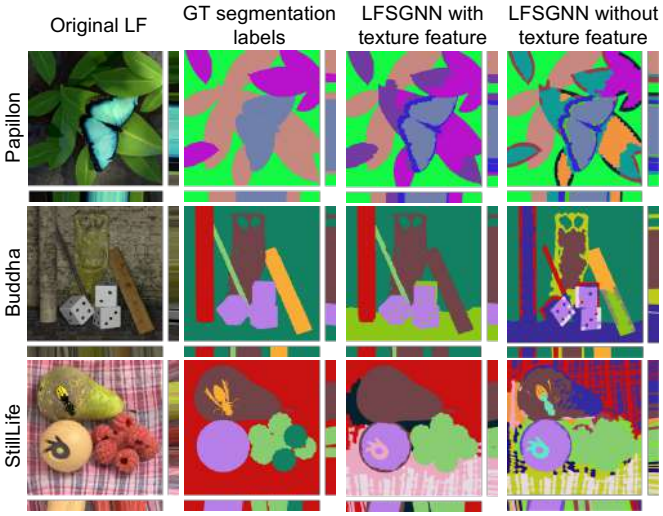


Fig. 6. Examples of the influence of using the texture feature during the hypergraph generation and GNN optimization steps.

TABLE II
LIGHT FIELD SEGMENTATION ANGULAR CONSISTENCY METRICS WITH DIFFERENT WINDOW SIZES USING ESTIMATED DISPARITY MAPS

LF \ N	3×3			5×5			7×7			9×9		
	SAC	IP [%]	$\overline{LLP}_{IP}$	SAC	IP [%]	$\overline{LLP}_{IP}$	SAC	IP [%]	$\overline{LLP}_{IP}$	SAC	IP [%]	$\overline{LLP}_{IP}$
Papillon	**1.00**	**0.04**	**2.35**	1.00	0.14	2.93	1.00	0.27	3.52	1.00	0.42	4.18
Buddha	**1.00**	**0.05**	**2.52**	1.00	0.15	3.15	1.00	0.29	3.91	1.00	0.48	4.21
StillLife	**1.00**	**0.13**	**2.55**	1.00	0.35	3.32	1.00	0.64	3.98	1.00	1.06	4.30
Horses	**1.00**	**0.11**	**2.77**	1.00	0.48	3.49	1.00	0.89	4.47	1.00	1.46	5.50
Kitchen	**1.00**	**0.11**	**2.99**	1.00	0.29	3.86	1.00	0.49	4.92	1.00	0.74	6.20
Room	**0.98**	**8.91**	**2.98**	0.97	19.15	5.34	0.95	28.78	8.59	0.94	38.65	12.52
Antique	**1.00**	**0.17**	**2.39**	1.00	0.54	2.79	1.00	1.09	3.17	1.00	1.70	3.64
Avg.	**1.00**	**1.36**	**2.65**	1.00	3.01	3.55	0.99	4.63	4.65	0.99	6.36	5.79

TABLE III
LIGHT FIELD SEGMENTATION ANGULAR CONSISTENCY METRICS WITH DIFFERENT WINDOW SIZES USING GT DISPARITY MAPS

LF \ N	3×3			5×5			7×7			9×9		
	SAC	IP [%]	$\overline{LLP}_{IP}$	SAC	IP [%]	$\overline{LLP}_{IP}$	SAC	IP [%]	$\overline{LLP}_{IP}$	SAC	IP [%]	$\overline{LLP}_{IP}$
Papillon	**1.00**	**0.02**	**2.60**	1.00	0.04	3.81	1.00	0.05	5.01	1.00	0.09	6.14
Buddha	**1.00**	**0.03**	**2.99**	1.00	0.05	4.46	1.00	0.06	5.92	1.00	0.10	6.36
StillLife	**1.00**	**0.18**	**2.72**	1.00	0.30	4.35	1.00	0.40	6.16	1.00	0.59	7.11
Horses	**1.00**	**0.04**	**3.07**	1.00	0.08	4.96	1.00	0.13	6.96	1.00	0.17	8.70
Kitchen	**1.00**	**0.07**	**2.94**	1.00	0.19	3.98	1.00	0.31	5.65	1.00	0.45	7.94
Room	**1.00**	**0.07**	**2.71**	1.00	0.17	3.39	1.00	0.33	4.57	1.00	0.70	6.15
Antique	**1.00**	**0.01**	**2.56**	1.00	0.03	3.18	1.00	0.05	4.07	1.00	0.08	4.44
Avg.	**1.00**	**0.06**	**2.80**	1.00	0.12	4.02	1.00	0.19	5.48	1.00	0.31	6.69

TABLE IV
QUANTITATIVE RESULTS USING ESTIMATED AND GT DISPARITY MAPS

	LFSGNN using estimated disparity		LFSGNN using GT disparity	
LF	mIoU	LP	mIoU	LP
Papillon	**0.63**	1.01	0.62	**1.00**
Buddha	0.75	1.02	**0.86**	**1.01**
StillLife	**0.51**	**1.04**	0.43	1.04
Horses	0.28	1.07	**0.35**	**1.01**
Kitchen	0.20	1.04	**0.32**	**1.03**
Room	**0.26**	5.45	0.16	**1.04**
Antique	0.17	1.04	**0.18**	**1.00**
Avg.	0.40	1.67	**0.42**	**1.02**

TABLE V
QUANTITATIVE RESULTS WITH AND WITHOUT USING THE TEXTURE FEATURE DURING THE HYPERGRAPH GENERATION AND GNN OPTIMIZATION STEPS

	LFSGNN with texture feature					LFSGNN without texture feature				
LF	mIoU	LP	SAC	IP [%]	$\overline{LLP}_{IP}$	mIoU	LP	SAC	IP [%]	$\overline{LLP}_{IP}$
Papillon	**0.63**	**1.01**	**1.00**	**0.04**	**2.35**	0.62	1.04	1.00	0.05	2.39
Buddha	**0.75**	**1.02**	**1.00**	**0.05**	**2.52**	0.46	1.03	1.00	0.05	2.58
StillLife	**0.51**	**1.04**	**1.00**	**0.13**	**2.55**	0.48	1.05	1.00	0.23	2.60
Horses	0.28	**1.07**	**1.00**	**0.11**	2.77	**0.31**	1.07	1.00	0.12	**2.70**
Kitchen	0.20	1.04	**1.00**	0.11	2.99	**0.24**	**1.02**	**1.00**	**0.07**	**2.89**
Room	0.26	5.45	**0.98**	8.91	**2.98**	**0.33**	**5.27**	**0.98**	**8.48**	**2.98**
Antique	0.17	**1.04**	**1.00**	**0.17**	**2.39**	**0.22**	1.06	1.00	0.20	2.42
Avg.	**0.40**	1.67	**1.00**	1.36	**2.65**	0.38	**1.65**	1.00	**1.31**	**2.65**

TABLE VI
QUANTITATIVE RESULTS USING A DIFFERENT NUMBER OF NETWORK MODEL COMPONENTS (M)

	LFSGNN when $M = 2$					LFSGNN when $M = 4$					LFSGNN when $M = 6$				
LF	mIoU	LP	SAC	IP [%]	$\overline{LLP}_{IP}$	mIoU	LP	SAC	IP [%]	$\overline{LLP}_{IP}$	mIoU	LP	SAC	IP [%]	$\overline{LLP}_{IP}$
Papillon	**0.63**	**1.01**	**1.00**	0.04	2.35	0.57	1.01	**1.00**	0.04	2.35	0.53	1.01	**1.00**	**0.03**	**2.33**
Buddha	**0.75**	1.02	**1.00**	0.05	**2.52**	0.72	1.02	**1.00**	0.04	2.56	0.54	**1.01**	**1.00**	**0.04**	2.56
StillLife	0.51	1.04	**1.00**	0.13	2.55	0.42	1.05	**1.00**	0.20	2.62	**0.53**	**1.02**	**1.00**	**0.06**	**2.28**
Horses	0.28	1.07	**1.00**	0.11	2.77	**0.31**	1.05	**1.00**	0.10	**2.68**	0.27	**1.02**	**1.00**	**0.03**	2.73
Kitchen	0.20	1.04	**1.00**	0.11	2.99	**0.27**	1.06	**1.00**	0.14	2.81	0.26	**1.02**	**1.00**	**0.07**	**2.46**
Room	**0.26**	5.45	0.98	8.91	2.98	0.14	3.29	0.99	3.79	**2.89**	0.13	**2.61**	**0.99**	**3.10**	2.92
Antique	**0.17**	1.04	**1.00**	0.17	**2.39**	0.14	1.04	**1.00**	0.14	2.42	0.16	**1.02**	**1.00**	**0.08**	**2.39**
Avg.	**0.40**	1.67	**1.00**	1.36	2.65	0.37	1.36	**1.00**	0.64	2.62	0.35	**1.25**	**1.00**	**0.49**	**2.52**

Hence, it can benefit the evaluation metrics (especially when the number of objects in the LF is larger than the minimum number of labels parameter) without necessarily improving the visual results. As can be seen in Table V, there are no significant differences in segmentation angular consistency metrics in most LFs. The reason for this is that the segmentation angular consistency relies on the disparity estimation and 4D LF over-segmentation steps, and those steps do not rely on the texture feature. Hence, when using the same disparity maps, LFs are represented based on hyperpixels where the consistency is ensured similarly. The reason for the slight difference in the angular consistency metrics is typically due to the reached number of unique labels in the final segmented LFs (this can be noticed in Fig. 6 where segmented LFs without using the texture feature have more unique labels). Moreover, the rounding error in pixel projection when computing those metrics can also affect their results. To study the influence of the used number of components, M, in the proposed model (where the texture feature is included), different values of M are used and the results are reported in Table VI which shows better performance in terms of angular consistency metrics when using larger M values without necessarily improving the segmentation accuracy (mIoU). This is typically because of a limitation in GNNs when increasing the GraphSAGE layers, the model over-smoothens the predicted segments, which leads to the merging of unrelated objects (resulting in a smaller number of labels). Accordingly, when the M value increases different objects are merged as shown in Fig. 7. Thus, the possibility of error occurrence in pixel projection when computing the angular consistency metrics is decreased. To avoid the over-smoothing effect, the value of $M = 2$ is adopted to lead to a reasonable balance between segmentation accuracy and angular consistency.

G. Comparison with the Benchmark Methods

Our results are compared with the benchmark methods on different LF datasets as shown in Fig. 8, Fig. 9 and Table VII. The proposed method and the benchmark methods are based on unsupervised learning; hence the values of the predicted labels differ from the GT segmentation labels. Therefore, to facilitate the visual comparisons, the labels of GT segmentation maps in the synthetic LFs are mapped to the labels of each method as shown in Fig. 9. The real world LF dataset does not have GT segmentation labels. Hence, the colors of the predicted labels across the used methods are not related to each other, as can be seen in Fig. 8.

In Table VII, the proposed segmentation method outperforms the benchmarks in terms of accuracy and consistency in most LFs. To visually notice the segmentation angular consistency, the central horizontal and vertical EPIs are presented for all methods. The segmentation angular consistency of all LF views can be more clearly noticed in the dynamic results in the supplemental material. Kim *et al.* [30] and Aflalo *et al*. [4] methods suffer from discontinuity in the predicted segments, where sparse pixels with wrong labels can be noticed in Fig. 9.

Although the same initial values are used for the training parameters in each of the benchmark methods to consider the same segmentation behavior for all LF views, the evaluation metrics still indicate inconsistent results with high values of

LP. This highlights the importance of explicitly considering the angular correlation in 4D LFs and the effectiveness of the used hypergraph representation (which allows applying segmentation to the entire 4D LF simultaneously). As can be noticed in Table VII, benchmark results of mIoU in some LFs (e.g., in Kitchen) are the same or better than the proposed method but significantly worse in terms of angular consistency metrics. This is because the mIoU metric considers the largest overlapping between the GT segmentation label and all the predicted labels in LF views (as described earlier in this paper).

However, the angular consistency metrics are significantly affected by the inconsistency across LF views, which are essentially noticeable at the boundaries of objects. Moreover, for some LFs, the benchmark methods terminate before reaching the minimum number of labels, C_{min}, since they have different architectures and differ in the used features.

Thus, they can end up with more unique labels than our method which may positively affect the used accuracy metric if the content of LF has more objects than the C_{min} value. Finally, using hyperpixels as the starting point allows exploiting the entire LF data during the segmentation and ensures segmentation angular consistency. Moreover, the proposed method is trained in an unsupervised manner which makes it suitable for different applications. One possible direction for improving the performance of the proposed method is to fine-tune the used features for domain-specific tasks (e.g., LF medical imaging). A major limitation of the proposed method is that it does not inherently determine the number of objects in an LF. Hence, a technique that adequately estimates the minimum number of labels based on the LF content would be extremely useful to further improve the segmentation results. Moreover, optimizing the implementation of the included steps can reduce the required computational time.

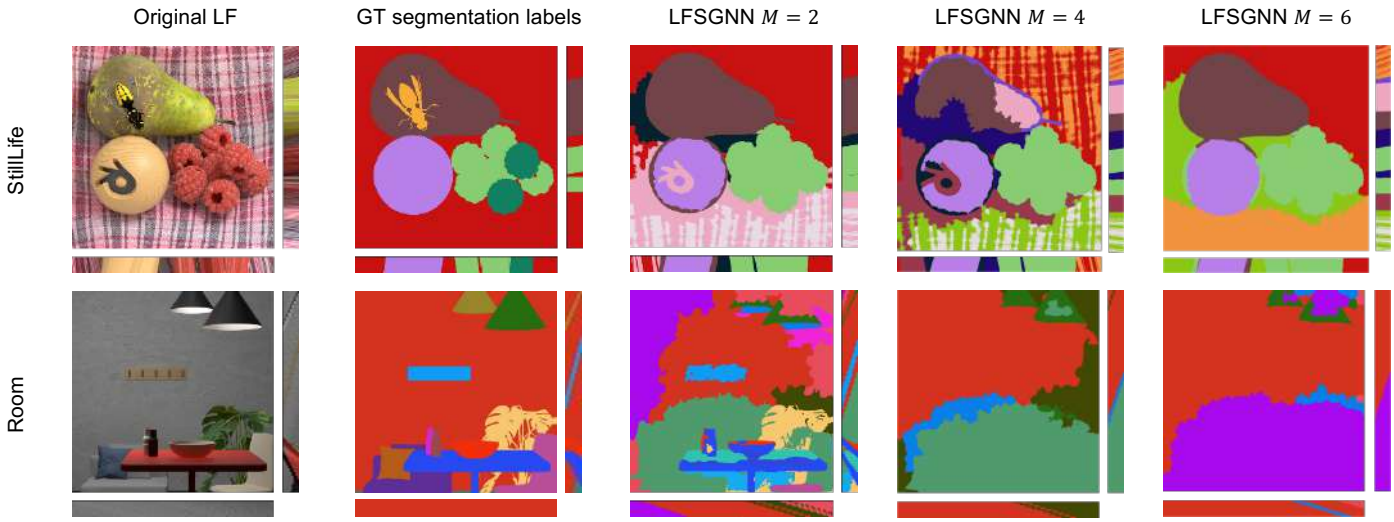


Fig. 7. Examples of using the proposed method with a different number of components, M. In some LFs, such as Room, a higher value of M can result in over-smoothing and different objects can be merged which negatively affects the segmentation accuracy.

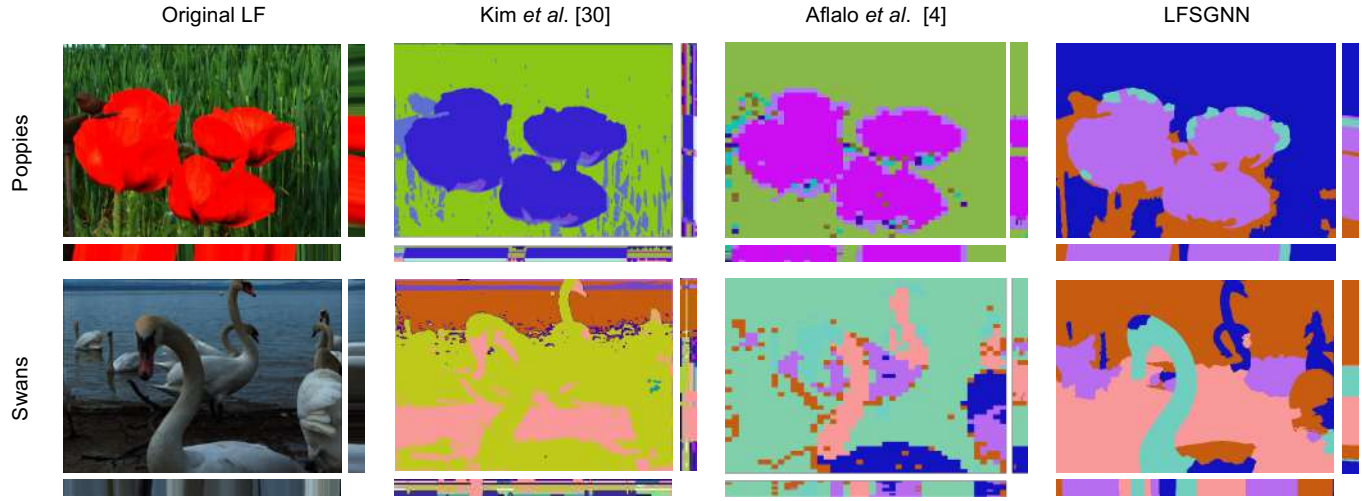


Fig. 8. Examples of unsupervised 4D LF segmentations on real world LFs for different methods. For each LF, the central view and the central horizontal/vertical EPIs are presented to show the segmentation angular consistency across LF views. The minimum number of labels is set to 5. Our results ensure angular consistency as can be seen in the presented EPIs (composed of mostly regular slanted lines) and adhere to object boundaries according to the reached number of labels (e.g., the swan head).

Fig. 9. Examples of unsupervised 4D LF segmentation on synthetic LFs for different methods. For each LF, the central view and the central horizontal and vertical EPIs are presented to show the segmentation angular consistency across LF views. The minimum number of labels is set to 5. Depending on the reached number of labels in each LF view and the actual number of objects in each LF, benchmark methods in some 4D LFs achieve better visual accuracy in terms of adhering to the object boundaries. This happens especially in sparse LFs (e.g., Kitchen and Room) where the estimated disparity maps used for LFSGNN are not accurate for all pixels. However, for accurate disparity maps, such as in the dense LFs (e.g., Papillon, Buddha, StillLife and Horses), LFSGNN achieves better separation between objects that share similar color or texture but vary in their depth (e.g., leaves in Papillon and the left pillar in Buddha). The angular consistency is better in both sparse and dense LFs compared to the benchmark methods, as can be seen in the central EPIs (composed of mostly regular slanted lines).

TABLE VII
QUANTITATIVE RESULTS USING OUR RESULTS AND BENCHMARK METHODS

	Kim *et al.* [30]					Aflalo *et al.* [4]					LFSGNN				
LF	mIoU	LP	SAC	IP [%]	$\overline{LLP}_{IP}$	mIoU	LP	SAC	IP [%]	$\overline{LLP}_{IP}$	mIoU	LP	SAC	IP [%]	$\overline{LLP}_{IP}$
Papillon	0.52	12.98	0.93	22.91	3.41	0.53	10.72	0.89	26.87	4.32	**0.63**	**1.01**	**1.00**	**0.04**	**2.35**
Buddha	0.48	23.80	0.68	70.08	4.69	0.44	6.80	0.93	18.72	3.81	**0.75**	**1.02**	**1.00**	**0.05**	**2.52**
StillLife	0.41	3.16	0.97	8.56	3.50	**0.54**	7.12	0.94	18.32	3.57	0.51	**1.04**	**1.00**	**0.13**	**2.55**
Horses	0.24	11.73	0.91	19.89	4.36	**0.39**	10.10	0.91	24.66	3.77	0.28	**1.07**	**1.00**	**0.11**	**2.77**
Kitchen	**0.22**	18.70	0.77	47.91	4.85	0.20	31.63	0.78	61.60	3.86	0.20	**1.04**	**1.00**	**0.11**	**2.99**
Room	0.23	31.85	0.65	81.46	4.43	0.12	33.19	0.70	80.62	3.90	**0.26**	**5.45**	**0.98**	**8.91**	**2.98**
Antique	0.16	25.35	0.78	52.17	4.36	0.13	17.79	0.83	42.98	4.16	**0.17**	**1.04**	**1.00**	**0.17**	**2.39**
Avg.	0.32	18.22	0.81	43.28	4.23	0.34	16.76	0.86	39.11	3.91	**0.40**	**1.67**	**1.00**	**1.36**	**2.65**

TABLE VIII
AVERAGE RUNNING TIME IN SECONDS PER VIEW

LF dataset	Kim *et al.* [30]	Aflalo *et al.* [4]	LFSGNN				
			Step 1	Step 2	Step 3	Step 4	Total time
HCI dataset [32]	83.60	0.62	1.90	8.96	7.82	7.59	26.27
MMSPG dataset [33]	20.12	0.66	1.83	3.33	2.67	1.09	8.93
IT-4DLF dataset [10]	43.93	0.45	1.75	4.29	3.62	2.04	11.71

To compare the computational complexity between the proposed method and the benchmark methods, all methods were run on the same computer and the GPU is being used under similar conditions for all methods. The running times are reported in Table VIII. The breakdown running time of all the steps of the proposed LFSGNN method is divided by the number of views to obtain running time per view for each step and reported in Table VIII. The summation of all steps of the proposed method is also computed. In Table VIII, although both our proposed method and Kim *et al.* [30] method iterate for 1000 epochs, our proposed method reduces the running time compared to Kim *et al.* [30] significantly. The method proposed by Aflalo *et al.* [4] iterates only for 10 epochs since it relies on an available pre-trained vision transformer to extract the deep features before performing image segmentation, thus it has the lowest running time in Table VIII. Training their lightweight model beyond 10 epochs was also tried (i.e., up to 1000) but the performance did not show significant improvements. In fact, this shows one advantage, in terms of computational complexity, of extracting the deep features from pre-trained foundation models and then applying fine-tuning for a few iterations while performing a specific task as image segmentation. Considering the number of epochs, our proposed method has shown improvement in both segmentation accuracy, angular consistency and a significant reduction in the computational complexity.

VI. CONCLUSION

In this paper, a novel unsupervised angularly consistent 4D LF segmentation method is proposed for both dense and sparse LFs. Initially, the 4D LF is represented as a hypergraph based on 4D hyperpixel over-segmentation. Afterwards, a GNN model is designed to extract deep features of the hypergraph and to group the hypernodes into objects by applying message passing and aggregation iteratively until convergence is reached. Different from existing 4D LF segmentation methods, the proposed method is fully unsupervised, represents 4D LFs robustly and efficiently, exploits the power of deep learning of graph-structured data and supports both dense and sparse 4D LFs. Experimental results show outperforming segmentation performance for most dense and sparse 4D LFs in terms of segmentation accuracy, angular consistency, and computational complexity.

For future work, the proposed LFSGNN method can be adapted for other imaging modalities, such as point clouds and multi-view images. Additionally, the inclusion of pre-trained foundation models in the 4D LF segmentation task can be also an interesting direction for future work. Moreover, the resulting 4D LF segmentation can be used for other applications such as in augmented reality where a segmented 4D object can be inserted in other 4D LFs. Finally, extending the proposed LFSGNN method to LF videos by considering the temporal dimension is also an interesting research direction that requires further investigation.

REFERENCES

[1] M. Levoy and P. Hanrahan, "Light field rendering," in *the 23rd Annu. Conf. Comput. Graph. Interact. Techn. (SIGGRAPH),* New Orleans, LA, USA, 1996, pp. 31–42.

[2] M. Zhao *et al.*, "A survey for light field super-resolution," *High-Confidence Comput.*, vol. 4, no. 1, p. 100206, Mar. 2024.

[3] S. Minaee, Y. Y. Boykov, F. Porikli, A. J. Plaza, N. Kehtarnavaz, and D. Terzopoulos, "Image Segmentation Using Deep Learning: A Survey," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 44, no. 7, pp. 1–1, Jul. 2021.

[4] A. Aflalo, S. Bagon, T. Kashti, and Y. Eldar, "DeepCut: Unsupervised Segmentation using Graph Neural Networks Clustering," in *IEEE/CVF Int. Conf. Comput. Vis. (ICCV)*, Paris,

France, 2023, pp. 32–41.
[5] M. Hog, N. Sabater, and C. Guillemot, "Superrays for Efficient Light Field Processing," *IEEE J. Sel. Top. Signal Process.*, vol. 11, no. 7, pp. 1187–1199, Oct. 2017.
[6] H. Zhu, Q. Zhang, Q. Wang, and H. Li, "4D Light Field Superpixel and Segmentation," *IEEE Trans. Image Process.*, vol. 29, pp. 85–99, Dec. 2020.
[7] N. Khan, Q. Zhang, L. Kasser, H. Stone, M. H. Kim, and J. Tompkin, "View-Consistent 4D Light Field Superpixel Segmentation," in *IEEE/CVF Int. Conf. Comput. Vis. (ICCV),* Seoul, Korea, 2019, pp. 7810–7818.
[8] 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.
[9] M. Hamad, C. Conti, P. Nunes, and L. D. Soares, "ALFO: Adaptive Light Field Over-Segmentation," *IEEE Access*, vol. 9, pp. 131147–131165, Sep. 2021.
[10] M. Hamad, C. Conti, P. Nunes, and L. D. Soares, "Hyperpixels: Flexible 4D Over-Segmentation for Dense and Sparse Light Fields," *IEEE Trans. Image Process.*, vol. 32, pp. 3790–3805, Jul. 2023.
[11] S. Wanner, C. Straehle, and B. Goldluecke, "Globally Consistent Multi-label Assignment on the Ray Space of 4D Light Fields," in *IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR),* Portland, USA, 2013, pp. 1011–1018.
[12] H. Mihara, T. Funatomi, K. Tanaka, H. Kubo, Y. Mukaigawa, and H. Nagahara, "4D light field segmentation with spatial and angular consistencies," in *IEEE In. Conf. Comput. Photography (ICCP)*, Evanston, USA, 2016, pp. 1–8.
[13] M. Hog, N. Sabater, and C. Guillemot, "Light Field Segmentation Using a Ray-Based Graph Structure," in *European Conf. Comput. Vis.*, Amsterdam, Netherlands: Springer, Cham, 2016, pp. 35–50.
[14] X. Lv, X. Wang, Q. Wang, and J. Yu, "4D Light Field Segmentation From Light Field Super-Pixel Hypergraph Representation," *IEEE Trans. Vis. Comput. Graph.*, vol. 27, no. 9, pp. 3597–3610, Sep. 2021.
[15] M. Hamad, C. Conti, A. M. De Almeida, P. Nunes, and L. D. Soares, "SLFS: Semi-supervised light-field foreground-background segmentation," in *2021 Telecoms Conf., (ConfTELE),* Leiria, Portugal, 2021, pp. 1–6.
[16] D. Yang, T. Zhu, S. Wang, S. Wang, and Z. Xiong, "LFRSNet: A robust light field semantic segmentation network combining contextual and geometric features," *Front. Environ. Sci.*, vol. 10, p. 1443, Oct. 2022.
[17] H. Sheng, R. Cong, D. Yang, R. Chen, S. Wang, and Z. Cui, "UrbanLF: A Comprehensive Light Field Dataset for Semantic Segmentation of Urban Scenes," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 32, no. 11, pp. 7880–7893, Nov. 2022.
[18] Y. Li *et al.*, "Multi-view Semantic Information Guidance for Light Field Image Segmentation," in *IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Vancouver, BC, Canada, 2023, pp. 3454–3462.
[19] C. Jia *et al.*, "Semantic Segmentation With Light Field Imaging and Convolutional Neural Networks," *IEEE Trans. Instrum. Meas.*, vol. 70, pp. 1–14, Sep. 2021.
[20] K. Zhu, N. N. Xiong, and M. Lu, "A Survey of Weakly-supervised Semantic Segmentation," in *IEEE 9th Int. Conf. Big Data Security on Cloud, IEEE Int. Conf. High Perform. Smart Comput., and IEEE Int. Conf. Intell. Data Security (BigDataSecurity-HPSC-IDS)*, New York, NY, USA, 2023, pp. 10–15.
[21] M. Caron *et al.*, "Emerging Properties in Self-Supervised Vision Transformers," *Proc. IEEE Int. Conf. Comput. Vis.(ICCV)*, Montreal, QC, Canada, 2021, pp. 9630–9640.
[22] Y. Xu, H. Nagahara, A. Shimada, and R. Taniguchi, "TransCut2: Transparent Object Segmentation From a Light-Field Image," *IEEE Trans. Comput. Imaging*, vol. 5, no. 3, pp. 465–477, Jan. 2019.
[23] P. Matysiak, M. Grogan, W. Aenchbacher, and A. Smolic, "Soft Colour Segmentation On Light Fields," in *IEEE Int. Conf. Image Process. (ICIP)*, Abu Dhabi, United Arab Emirates, 2020, pp. 2621–2625.
[24] R. Cong, D. Yang, R. Chen, S. Wang, Z. Cui, and H. Sheng, "Combining Implicit-Explicit View Correlation for Light Field Semantic Segmentation," in *IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Vancouver, Canada, 2023, pp. 9172–9181.
[25] M. Hamad, C. Conti, P. Nunes, and L. D. Soares, "Efficient Propagation Method for Angularly Consistent 4D Light Field Disparity Maps," *IEEE Access*, vol. 11, pp. 63463–63474, Jun. 2023.
[26] 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.
[27] A. Borji and L. Itti, "Exploiting local and global patch rarities for saliency detection," in *IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Providence, RI, USA, 2012, pp. 478–485.
[28] T. Ojala, M. Pietikäinen, and T. Mäenpää, "Multiresolution gray-scale and rotation invariant texture classification with local binary patterns," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 24, no. 7, pp. 971–987, Jul. 2002.
[29] W. L. Hamilton, R. Ying, and J. Leskovec, "Inductive Representation Learning on Large Graphs," *Adv. Neural Inf. Process. Syst.*, vol. 2017, pp. 1025–1035, Jun. 2017.
[30] W. Kim, A. Kanezaki, and M. Tanaka, "Unsupervised Learning of Image Segmentation Based on Differentiable Feature Clustering," *IEEE Trans. Image Process.*, vol. 29, pp. 8055–8068, Jul. 2020.
[31] K. He, X. Zhang, S. Ren, and J. Sun, "Delving Deep into Rectifiers: Surpassing Human-Level Performance on ImageNet Classification," in *IEEE/CVF Int. Conf. Comput. Vis. (ICCV)*, Santiago, Chile, 2015, pp. 1026–1034.
[32] S. Wanner, S. Meister, and B. Goldlücke, "Datasets and Benchmarks for Densely Sampled 4D Light Fields," *Vision, Model. Vis.*, vol. 13, pp. 225–226, Sep. 2013.
[33] M. Rerabek and T. Ebrahimi, "New Light Field Image Dataset," in *Inter. Conf. Qual. Multimedia Exper. (QoMEX)*, Lisbon, Portugal, 2016.

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 a 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 at Middle East Technical University (METU) in the ERASMUS+ Program, Turkey. She completed her professional internship in information science and technology with the 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 include 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. She acts as a reviewer for the IEEE Access, Signal, Image and Video Processing, and IEEE Transactions on Image Processing journals.

CAROLINE CONTI (Member, IEEE) received the B.Sc. degree in electrical engineering from Universidade de São Paulo (USP), Brazil, in 2010, and the Ph.D. degree in information science and technology from Instituto Universitário de Lisboa (ISCTE- IUL), Portugal, in 2017. She is currently a Senior Researcher with the Multimedia Signal Processing Group, Instituto de Telecomunicações, and an Assistant Professor with the Department of Information Science and Technology, Instituto Universitário de Lisboa (ISCTE-IUL), Portugal. Her research interests include immersive visual technologies and image and video processing, including light field processing and coding. She has contributed more than 25 papers to international journals and conferences in these areas. She serves as an Associate Editor for IEEE TRANSACTIONS ON IMAGE PROCESSING. She has been a Guest Editor for Signal Processing: Image Communication (Elsevier). She actively participates as a reviewer for various IEEE and EURASIP journals and conferences.

PAULO NUNES (Member, IEEE) received the degree in electrical and computer engineering from Instituto Superior Técnico (IST), Universidade 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 Department of Information Science and Technology, 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 for the European Commission. 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, SPIE, and Springer conferences and journals and as a member of the program and organizing committees of various international conferences. He has contributed more than 70 papers to international journals and conferences in these areas.

LUÍS DUCLA SOARES (Senior Member, IEEE) received the Licenciatura and Ph.D. degrees in electrical and computer engineering from Instituto Superior Técnico (IST), Universidade 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 Department of Information Science and Technology, Instituto Universitário de Lisboa (ISCTE-IUL), Portugal. His research interests include coding and processing of visual information modalities, including light fields. He has contributed more than 70 papers to international journals and conferences in these areas. 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.

Chapter 7

View-consistent 4D Light Field Style Transfer using Neural Networks and Over-segmentation

Conference: IEEE 14th Image, Video, and Multidimensional Signal Processing Workshop (IVMSP)
Date of Conference: June 2022
Location: Nafplio, Greece
Pages: 1-5
DOI: 10.1109/IVMSP54334.2022.9816312.

View-consistent 4D Light Field Style Transfer using Neural Networks and Over-segmentation

Maryam Hamad, Caroline Conti, Paulo Nunes, Luís Ducla Soares
Instituto de Telecomunicações, Instituto Universitário de Lisboa (ISCTE-IUL)
Lisbon, Portugal
{maryam.hamad, caroline.conti, paulo.nunes, lds}@lx.it.pt

2022 IEEE 14th Image, Video, and Multidimensional Signal Processing Workshop (IVMSP) | 978-1-6654-7822-9/22/$31.00 ©2022 IEEE | DOI: 10.1109/IVMSP54334.2022.9816312

***Abstract*—Deep learning has shown promising results in several computer vision applications, such as style transfer applications. Style transfer aims at generating a new image by combining the content of one image with the style and color palette of another image. When applying style transfer to a 4D Light Field (LF) that represents the same scene from different angular perspectives, new challenges and requirements are involved. While the visually appealing quality of the stylized image is an important criterion in 2D images, cross-view consistency is essential in 4D LFs. Moreover, the need for large datasets to train new robust models arises as another challenge due to the limited LF datasets that are currently available. In this paper, a neural style transfer approach is used, along with a robust propagation based on over-segmentation, to stylize 4D LFs. Experimental results show that the proposed solution outperforms the state-of-the-art without any need for training or fine-tuning existing ones while maintaining consistency across LF views.**

Keywords—light field; angular consistency; deep learning; neural style transfer; superpixels

I. INTRODUCTION

Appealing paintings and artwork have attracted people for thousands of years. In the past, a skilled artist was always required to create a painting with a specific style, brush strokes and color palette, which typically took a long time. With the recent advances in learning-based techniques and the advent of style transfer, such creation is now possible to be performed by computers. Style transfer is an image editing application in which a new image is generated by combining the content of one image with the style of another one (e.g., a famous painting). Style transfer is a long-standing research area in the broader area of texture synthesis [1], [2]. Recently, with the rapid development of deep learning, neural networks are being used to solve the style transfer task. Gatys et al. [3] were the first to apply Convolutional Neural Networks (CNN) to stylize an image. In their work, CNNs are used to extract the feature maps of the content image (i.e., the image from which the content will be transferred) and style image (i.e., the image from which the style will be transferred). Afterwards, a target image (i.e., the stylized image that combines the content image with the style image) is iteratively optimized by minimizing a loss function. Johnson et al. [4] improved the performance of [3] by training a feed-forward network for each style image and generating a stylized image with only one forward pass in the testing stage. Although it is 3 times faster than [3], the solution in [4] is not flexible in terms of the number of used styles since it requires training for each style. Additionally, other neural networks have also been exploited to achieve style transfer, such as generative adversarial networks that require paired training data to learn a specific style, which is not always available and may limit their applications [1]. Moreover, Neural Style Transfer (NST) has been extended to consider videos [5] and different imaging modalities, such as stereo imaging [6] and 4D Light Fields (LF) [7], [8]; interested readers are encouraged to read the recent comprehensive reviews of the existing NST solutions in [1], [2].

4D LFs involve rich information since not only the light intensity is captured but also ray directions [9]. LFs capture the same scene from different perspectives, thus allowing interesting applications such as depth or disparity estimation (i.e., the displacement of a point between different views, which is inversely proportional to the depth), view synthesis and post-capture refocusing [9], [10]. 4D LFs can be represented as an array of views $I(x, y, u, v)$, where (x, y) are the spatial coordinates, and (u, v) are the angular coordinates of each view. When fixing one angular and one spatial coordinates, an Epipolar Plane Image (EPIs) (i.e., the unique 2D spatio-angular LF slice typically containing a regular structure with several oriented lines [11]) can be obtained as illustrated in Figure 1.

While generating stylized images that are visually pleasant is an important criterion for 2D images, maintaining cross-view consistency is also essential for 4D LFs. More precisely, directly applying 2D image or video style transfer methods to the entire 4D LF views, without considering the correlation between them, may result in inconsistent stylized LFs with highly unnatural artifacts. Only a few solutions are available in the literature that consider 4D LF cues in the style transfer application. Hart et al. [7] proposed an extension to the work of Johnson et al. [4] by adding a disparity loss term to the loss function. The disparity loss is computed by finding the difference between each stylized LF view and the stylized central view warped into that view. The disparity loss is then backpropagated through the network. This repeats for each LF view until convergence is reached. While their work considers cross-view consistency, it requires optimizing each LF view iteratively (assuming dense LFs).

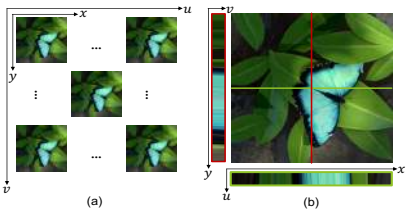


Fig. 1. Example of light field representations. a) 4D light field represented as an array of views; b) Horizontal and vertical EPIs.

This work was funded by FCT/MCTES through national funds under projects UIDB/50008/2020 and PTDC/EEI-COM/7096/2020.

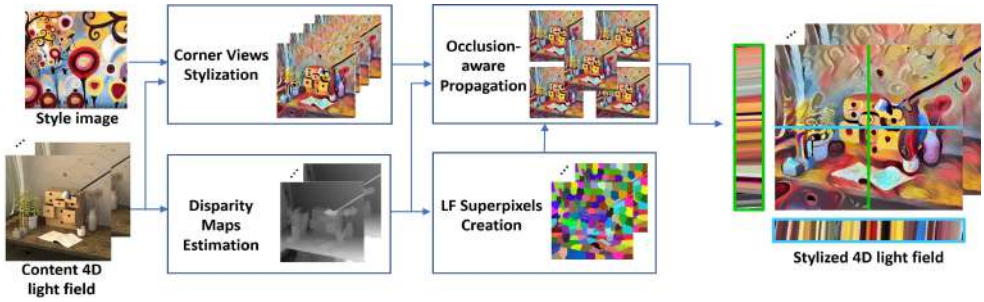


Fig. 2. Overview of the proposed method for view-consistent 4D LF neural style transfer. By combining the style of 2D image with the content of 4D LFs and applying an occlusion-aware propagation, a consistent 4D stylized LF is generated.

Moreover, although the feed-forward approach is fast, it needs to be trained for each style, hence, limiting style selection flexibility. Egan et al. [8] addressed these drawbacks and proposed a novel NST method that considers local angular consistency. Their work extended the Gatys et al. work [3] by adding the local angular consistency loss in the total loss function. Although their work ensures local angular consistency for LFs with larger disparity ranges, applying optimization using this technique for each view is very time-consuming.

The contribution of this paper is a novel 4D LF NST method that overcomes the limitations of the existing methods by:

- **Enabling NST flexibility (in terms of the number of styles that can be used) with less computational complexity**: to achieve that, the optimization-based NST [3] method (which does not require training a model for each style image) is used. To reduce the optimization-based NST complexity significantly, only a limited set of views (i.e., the four corner views) are initially stylized using the method in [3] (different from [7] and [8] that require optimizing each LF view).
- **Improving 4D LF view-consistency:** by exploiting LF over-segmentation (that adheres to object boundaries and maintains LF view-consistency), the edits from all corner views are propagated into each LF view using per-pixel disparity in an occlusion-aware manner. The proposed method outperforms the existing solutions without training or fine-tuning the existing NST models.

The remainder of this paper is organized as follows: Section II describes the proposed method in detail, and Section III evaluates its performance through a series of experimental results. Finally, Section IV concludes the paper with some final remarks and proposes directions for future work.

II. Proposed Method

The proposed method contains four main steps as presented in Figure 2. Given a style image and a 4D LF, the four corner views are initially stylized using optimization-based NST [3]. After that, disparity maps for all input LF views are estimated using [12]; to ensure spatio-angular consistency during the propagation. Next, the 4D LF is over-segmented into spatio-angular coherent regions (a.k.a superpixels), as in [13] to facilitate the propagation and respect object boundaries and occlusions. Afterwards, the stylization is propagated into all LF views through occlusion-aware back-projection from each view into all corner views. Finally, remaining isolated non-stylized pixels that emerged after back-projections due to occlusions, are filled robustly. Each step is detailed in the following subsections.

A. Corner Views Stylization

Initially, only the extreme four corner views are stylized using the approach in [3]. The corner views are selected since they typically contain the maximum scene information including dis-occlusions. The approach in [3] aims at minimizing the distances of the feature representation between the content/style image and the target one in one or more layers of the CNN. The target image is initially generated using a white noise image and iteratively optimized using the loss function, $\mathcal{L}_{total}$, defined by (1), where $\mathcal{L}_{content}$, is the content loss and $\mathcal{L}_{style}$, is the style loss. To ensure view-consistent stylization, the initial white noise is set the same for all corner views. Finally, to control the output, two weighting factors (i.e., the content weight, α, and the style weight, β) are included:

$$\mathcal{L}_{total} = \alpha\mathcal{L}_{content} + \beta\mathcal{L}_{style}. \quad (1)$$

Notice that the proposed method is independent of the used 2D NST method. However, the approach in [3] is used due to its flexibility to transfer any style and it enables controlling the target images by adjusting the weights in (1). Moreover, any number of views or angular positions can be used but the results may be influenced accordingly.

B. Disparity Maps Estimation

LF imaging provides rich information, which makes it possible to estimate a disparity map for each LF view. In this paper, the proposed method in [12] (that estimates disparities from each view to its right adjacent view) is used to estimate disparity maps for all LF views. Per-pixel disparity is used here to ensure consistent pixel projection during the propagation step.

C. Light Field Superpixel Creation

LF over-segmentation is capable of adhering to object boundaries and creating a unique label for each homogenous region to facilitate subsequent editing tasks. In this paper, the recently proposed Adaptive LF Over-segmentation (ALFO) method [13] is used to guide the propagation in an occlusion-aware manner. The ALFO method exploits color, disparity and position features to apply adaptive K-means clustering. Additionally, it can robustly balance accuracy, shape regularity and view-consistency. In our experiments, the superpixel size is set to 20 as suggested in [13] as a reasonable size for robust adhesion to the borders.

D. Occlusion-aware Propagation

Given the LF disparity maps, LF superpixels and stylized corner views, the stylization now can be propagated into all other 4D LF views. Initially, each LF view is back-projected into all corner views using its disparity map (2):

$$x_i^{ref} = x_i^{(u,v)} + d_{hor}^{(u,v)\to ref}, \quad y_i^{ref} = y_i^{(u,v)} + d_{ver}^{(u,v)\to ref}, \tag{2}$$

where $x_i^{(u,v)}, y_i^{(u,v)}$ are the spatial position coordinates of a pixel, i, which is located in a view of angular coordinates (u, v), x_i^{ref} and y_i^{ref} are the spatial position in a reference view (i.e., ref in this paper represents a single corner view, hence, the same equation is applied for all corner views independently), and $d_{hor}^{(u,v)\to ref}$, $d_{ver}^{(u,v)\to ref}$ are the horizontal and vertical disparity from view (u, v) to the reference view. The used disparity estimation method [12] estimates disparity for adjacent views, therefore, for regularly sampled 4D LF views back-projection is applied by multiplying the disparity value by $(u_{ref} - u)$, $(v_{ref} - v)$ when computing x_i^{ref} and y_i^{ref}, respectively [13]. These equations are applied in the case of parallel optical light field capturing assumption, as in [13]–[16]. Otherwise, intrinsic and extrinsic camera parameters should be considered.

Since the projected pixel coordinates may belong to $\mathbb{R}^2$, and to ensure integer indexing (since the visual information is only available for integer indices), the four neighboring pixels, $\mathcal{N}_i \in \{I_a, I_b, I_c, I_d\}$, of the back-projected pixel with integer positions $(\in \mathbb{Z}^2)$ are considered as presented in Figure 3. However, consistency is checked by comparing the label and disparity of the pixel in $\left(x_i^{(u,v)}, y_i^{(u,v)}\right)$ and all pixels in $\mathcal{N}_i$ to choose which ones to be used for the interpolation. To overcome possible projection errors, due to disparity errors or discontinuities in superpixels, two conditions are checked before interpolation:

- At least one pixel in $\mathcal{N}_i$ has the same label as the pixel in its original location $\left(x_i^{(u,v)}, y_i^{(u,v)}\right)$.
- The absolute disparity difference between a pixel disparity in view (u, v) and at least one pixel disparity in $\mathcal{N}_i$ is less than a threshold value, ε. We empirically set $\varepsilon = 0.1$; since a superpixel with size (i.e., 20) is noticed to have, typically, similar disparity values.

If any of the above conditions holds for all pixels in $\mathcal{N}_i$ or part of them, then only these pixels are valid for interpolation. Interpolation is applied by computing the bilinear interpolation of valid pixels in $\mathcal{N}_i$, otherwise, no interpolation is computed.

After computing the interpolated value from each corner view, the pixel in its original angular location (u, v) is set to the mean color value of all valid back-projections from the four corner views. The mean is used after extensive experiments since it shows the best visual and numerical results when compared to using the median or weighted sum and maintains consistency across LF views. By doing this, only very few sparse and isolated pixels that have no projection, or invisible regions due to the angle of view, remain unstylized. To fully stylize all LF views, these remaining isolated pixels are filled by applying inward interpolation using the widely used region filling based on the Laplace equation as in [17].

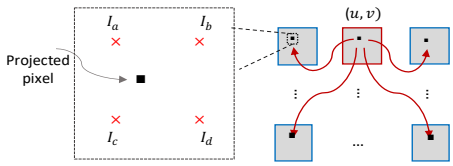


Fig. 3. Example of back-projection: a pixel in $(\boldsymbol{u}, \boldsymbol{v})$ view that needs to be stylized is back-projected into each corner view (in blue squares).

III. Experimental Results

In this section, several methods are used as benchmarks to evaluate the performance of the proposed method. Firstly, two different baseline methods are considered, as in [8]: i) by applying Independent View Stylization (IVS) using existing 2D NST [3] to all LF views independently; and ii) by applying Pseudo Video Stylization (PVS) as proposed (for videos) in [5] for styling a pseudo video sequence of 4D LF views. To the best of the authors' knowledge, only two recently proposed methods are specifically focused on tackling 4D LF challenges. The first one focuses on Global Angular Consistency Stylization (GACS) [7], and the second one focuses on Local Angular Consistency Stylization (LACS) [8]. Moreover, different synthetic and real-world LF datasets and style images are used, as shown in TABLE I. For quantitative evaluations, two different metrics are used to evaluate the view-consistency namely: i) the LF Epipolar Consistency (LFEC) metric defined in [18]; and ii) the LF Angular Consistency (LFAC) metric[1] defined in [8]. The LFEC and LFAC metrics evaluate the angular consistency by back-warping LF views into a reference view and finding the color variance. Different than the LFEC metric that back-warps all LF views into the central view, the LFAC metric back-warps into the center view of a local window of views; to robustly consider large occluded regions. Both metrics require estimating disparity to apply back-warping, therefore, we estimated per-pixel disparity maps, for our results and all benchmark methods by using [12]. We noticed that, by using [12], the metric results of the benchmark methods are improved. Moreover, the disparity loss (which is the amount of disparity changes) is evaluated by using the disparity Mean Square Error (MSE) metric defined in [7]. This metric computes the $MSE \times 100$ between the central view disparity map estimated from the original LFs and the stylized ones. As in [8], the disparity estimation method in [19] is used. Results of all metrics are presented in TABLE II. Due to the limitation in the paper size, only the central view with horizontal EPIs are presented in TABLE III. However, we encourage the reader to see our dynamic results[2] for all LF views for clear view-consistency evaluation. For the used NST implementation, we used standard GPU-based MATLAB implementation [20] and we set $\alpha = 50$, $\beta = 10^3$, the same values as used in the benchmark methods.

[1]Software implementation of all the used metrics can be found at: https://github.com/doegan32/Light-Field-Style-Transfer

[2]Dynamic results for all LF views can be found at: https://github.com/MaryamHamad/LFStyleTransfer

The proposed method generates outperforming angular consistency in both LFEC and LFAC metrics, as can be seen in TABLE II. For the MSE metric, the GACS method achieves the best average results and preserves better object boundaries; hence, generates the central disparity maps that are similar to the original LF ones. However, it requires a pre-trained NST model as input for each style image. In this paper, corner views are used to minimize the occlusions, hence, there are no large holes left after propagation in densely sampled LFs. However, our method can be extended to consider sparse LFs that may have largely occluded regions by simply adding more reference views to consider all objects in LF views. The used technique for filling the holes in dis-occluded regions after propagation may generate some artifacts (which also occur in the benchmark methods) and thus requires further investigation. For time complexity, the proposed method reduces the needed time to stylize the entire LF significantly, i.e., for a LF with 81 views instead of taking $81 \times T_s$, where T_s is the average time needed to stylize a single view, it takes less than $10 \times T_s$ including LF disparity estimation and superpixel generation. Finally, it can be observed that neither applying 2D methods for each view independently nor existing methods for video are adequate solutions for 4D LFs.

IV. Final Remarks

In this paper, a novel view-consistent 4D LF NST method is proposed. Without any further training for new deep learning models or fine-tuning existing ones, we exploited an existing optimization-based NST method to initially stylize only four corner views. Afterwards, the stylized views are propagated into all other LF views in an occlusion-aware manner by using LF superpixels. Experimental results have been shown to outperform the considered benchmark methods and produce visually appealing and consistent results across all LF views.

For future work, we will extend style transfer to sparse LFs that include wide occlusions. Additionally, we will study other applications of the proposed propagation technique, such as semantic segmentation and object removal, where the edits are applied in reference views and propagated into other LF views.

Acknowledgment

The authors would like to thank Mr. Dónal Egan for publishing the software of their method [8] including the evaluation metrics and results that facilitated their comparison.

TABLE I. Test Images Used In Our Experiments

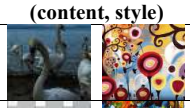
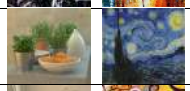
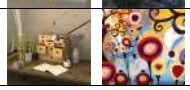
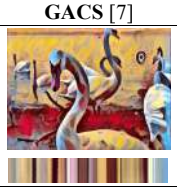
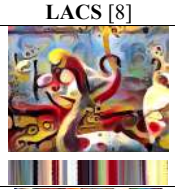
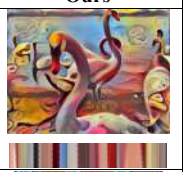
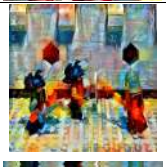
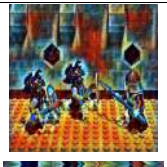
	content 4D LFs	Disparity range	Style image	Thumbnails: (content, style)
a	Swan [21]	[-1, 1]	Candy	
b	Lego knights [22]	[-3, 3]	Rain princess	
c	Bikes [23]	[-1, 1]	Rain princess	
d	Herbs [24]	[-3, 1.8]	Starry night	
e	Table [24]	[-2, 1.6]	Candy	

TABLE II. Angular Consistency (LFEC, LFAC) and Disparity Loss (MSE×100) Metrics

Metric		IVS (baseline)	PVS (baseline)	GACS [7]	LACS [8]	Ours
LFEC (↑)	a	19.29	25.24	30.31	28.43	**40.83**
	b	19.50	21.18	23.16	24.51	**29.21**
	c	22.14	23.10	33.77	27.48	**42.92**
	d	22.68	24.80	22.53	27.96	**31.29**
	e	19.02	22.03	28.19	25.72	**32.38**
Avg.		20.53	23.27	27.59	26.82	**35.33**
LFAC (↑)	a	29.74	41.27	46.48	42.72	**53.54**
	b	30.51	35.29	37.82	38.09	**44.00**
	c	33.66	37.70	48.45	42.13	**54.93**
	d	33.37	38.92	34.92	41.58	**45.49**
	e	29.65	36.69	42.54	39.81	**46.85**
Avg.		31.39	37.97	42.04	40.87	**48.96**
MSE × 100 (↓)	a	209.19	2.56	1.01	2.11	**0.54**
	b	256.36	22.15	**13.14**	19.50	18.85
	c	29.28	2.09	**0.91**	1.94	2.40
	d	96.08	11.68	8.10	8.31	**5.00**
	e	284.98	8.45	**1.39**	5.20	3.43
Avg.		175.18	9.39	**4.91**	7.41	6.04

TABLE III. Visual Comparison With Benchmark Methods

	Content LF	Style image	IVS (baseline)	PVS (baseline)	GACS [7]	LACS [8]	Ours
a							
b							

REFERENCES

[1] Y. Jing, Y. Yang, Z. Feng, J. Ye, Y. Yu, and M. Song, "Neural Style Transfer: A Review.," *IEEE Trans. Vis. Comput. Graph.*, vol. 26, no. 11, pp. 3365–3385, Nov. 2020.

[2] G. Sohaliya and K. Sharma, "An Evolution of Style Transfer from Artistic to Photorealistic: A Review," *2021 Asian Conf. Innov. Technol. (ASIANCON)*, Pune, India, Aug. 27-29, 2021, pp. 1-7.

[3] L. A. Gatys, A. S. Ecker, and M. Bethge, "Image Style Transfer Using Convolutional Neural Networks," *IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Las Vegas, NV, USA, June 27-30, 2016, pp. 2414–2423.

[4] J. Johnson, A. Alahi, and L. Fei-Fei, "Perceptual Losses for Real-Time Style Transfer and Super-Resolution," *Lect. Notes Comput. Sci. (including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics)*, vol. 9906 LNCS, pp. 694–711, 2016.

[5] M. Ruder, A. Dosovitskiy, and T. Brox, "Artistic Style Transfer for Videos," *German Conf. Pattern Recog.*, Springer Verlag, 2016, pp. 26–36.

[6] D. Chen, L. Yuan, J. Liao, N. Yu, and G. Hua, "Stereoscopic Neural Style Transfer," in *2018 IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Salt Lake City, UT, USA, June 18-23, 2018, pp. 6654–6663.

[7] D. Hart, J. Greenland, and B. Morse, "Style transfer for light field photography," *IEEE Winter Conf. Appl. Comput. Vision (WACV)*, Snowmass, CO, USA, 1-5 Mar., 2020, pp. 99–108.

[8] D. Egan, M. Alain, and A. Smolic, "Light Field Style Transfer with Local Angular Consistency," *IEEE Int. Conf. Acoust., Speech, Signal Process (ICASSP)*, Toronto, ON, Canada, June 6-11, 2021, pp. 2300–2304.

[9] G. Wu *et al.*, "Light Field Image Processing: An Overview," *IEEE J. Sel. Top. Signal Process.*, vol. 11, no. 7, pp. 926–954, Oct. 2017.

[10] S. Zhou, T. Zhu, K. Shi, Y. Li, W. Zheng, and J. Yong, "Review of light field technologies," *Vis. Comput. Ind. Biomed. Art*, vol. 4, no. 29, Dec. 2021.

[11] 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. 7–55, 1987.

[12] N. Khan, M. H. Kim, and J. Tompkin, "View-consistent 4D Light Field Depth Estimation," in *Proc. Brit. Mach. Vis. Conf. (BMVC)*, 2020.

[13] M. Hamad, C. Conti, P. Nunes, and L. D. Soares, "ALFO: Adaptive Light Field Over-Segmentation," *IEEE Access*, vol. 9, pp. 131147–131165, 2021.

[14] M. Hog, N. Sabater, and C. Guillemot, "Superrays for Efficient Light Field Processing," *IEEE J. Sel. Top. Signal Process.*, vol. 11, no. 7, pp. 1187–1199, Oct. 2017.

[15] N. Khan, Q. Zhang, L. Kasser, H. Stone, M. H. Kim, and J. Tompkin, "View-Consistent 4D Light Field Superpixel Segmentation," in *Int. Conf. Comput. Vis. (ICCV)*, Seoul, Korea, 27 Oct.-2 Nov., 2019, pp. 7810–7818.

[16] H. Zhu, Q. Zhang, Q. Wang, and H. Li, "4D Light Field Superpixel and Segmentation," *IEEE Trans. Image Process.*, vol. 29, pp. 85–99, July. 2019.

[17] "Fill in specified regions in image using inward interpolation - MATLAB regionfill - MathWorks." [Online]. Available: https://www.mathworks.com/help/images/ref/regionfill.html. [Accessed: 28-Mar-2022].

[18] P. David, M. Le Pendu, and C. Guillemot, "Angularly Consistent Light Field Video Interpolation," *IEEE Int. Conf. Multimedia Expo (ICME)*, London, UK, July 6-10, 2020, pp. 1–6.

[19] C. Shin, H. G. Jeon, Y. Yoon, I. S. Kweon, and S. J. Kim, "EPINET: A Fully-Convolutional Neural Network Using Epipolar Geometry for Depth from Light Field Images," *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit. (CVPR)*, pp. 4748–4757, Dec. 2018.

[20] "Neural Style Transfer Using Deep Learning - MATLAB & Simulink." [Online]. Available: https://www.mathworks.com/help/images/neural-style-transfer-using-deep-learning.html. [Accessed: 8-Apr-2022].

[21] M. Rerabek and T. Ebrahimi, "New Light Field Image Dataset," in *8th Int. Conf. Qual. Multimedia Exper. (QoMEX)*, Lisbon, Portugal, 2016.

[22] V. Vaish and A. Adams, "The (New) Stanford Light Field Archive," *Stanford*, 2008. [Online]. Available: http://lightfield.stanford.edu/acq.html. [Accessed: 8-Apr-2022].

[23] "Stanford Lytro Light Field Archive." [Online]. Available: http://lightfields.stanford.edu/LF2016.html. [Accessed: 8-Apr-2022].

[24] K. Honauer, O. Johannsen, D. Kondermann, and B. Goldluecke, "A Dataset and Evaluation Methodology for Depth Estimation on 4D Light Fields," in *13th Asian Conf. Comput. Vis., Taipei, Taiwan, November 20-24, 2016, Revised Selected Papers, Part III*, S.-H. Lai, Ed. Cham: Springer, 2017, pp. 19–34.

Chapter 8

Achievements and Future Directions

This chapter summarizes the core achievements of this Thesis and revisits the fundamental objectives that guided our research work. Additionally, this chapter also highlights the limitations encountered since they offer prospects for future research directions.

8.1 Discussion of achievements

Immersive visual content that provides higher DoF and enriches the end users' experience is increasingly popular in academia and industry alike. Different immersive imaging modalities exist, such as LF imaging, omnidirectional imaging, holography and volumetric imaging. While giant tech companies are providing new devices for capturing and displaying immersive imaging modalities, providing efficient processing and editing solutions for those modalities is of the utmost significance. Acknowledging this, this Thesis has addressed 4D LF imaging and considered this requirement by proposing novel methods that advance the state-of-the-art 4D LF processing and editing tasks. In this context, six main achievements were accomplished that together represent a pipeline to process and edit 4D LFs while ensuring accuracy and angular consistency.

The first achievement addressed the first Thesis objective and proposed an efficient disparity propagation method to ensure angular consistency. The proposed method in Chapter 2 enabled computing disparity maps not only for the central view but also per-pixel disparity maps for all LF views while supporting different disparity ranges. The proposed method exploited off-the-shelf state-of-the-art disparity estimation methods to estimate a disparity map for the central view, as well as for the regions in the corner views that do not appear in the central view. After that, efficient recursive propagation and refinement steps were applied to exploit the correlation in 4D LFs and compute disparity maps for all other LF views. The proposed method generated more accurate and angularly consistent disparity maps for all LF views when compared to the existing 4D LF disparity estimation methods for dense and sparse LFs while significantly reducing computational complexity.

The second and third achievements considered the second Thesis objective and proposed two different 4D LF over-segmentation methods. More precisely, the proposed method in Chapter 3 aimed at exploiting per-pixel disparity information as a clustering feature and developing an adaptive *K*-means clustering while using different clustering features (i.e., disparity, color and position). Adjusting the clustering weights automatically according to LF content has shown a balance improvement between over-segmentation accuracy, compactness (shape regularity of the clusters) and angular consistency [109]. Additionally, applying adaptive over-segmentation can lead to an optimal solution without being biased blindly to any clustering feature [109]. While this method improved the over-segmentation performance in terms of accuracy and angular consistency, it is only suitable for dense LF content.

The third achievement continued the second Thesis objective by extending, in Chapter 4, the LF superpixel/superray concepts into hyperpixels in 4D space [110] and presented a flexible 4D LF over-segmentation method for both dense and sparse 4D LFs. To achieve that, the centroids of hyperpixels were initialized in the central views and also in the regions existing in the corner views but occluded or non-existent in the central view; the over-segmentation was then applied in 4D space. Like the previous method, this method adopted an adaptive *K*-means clustering and exploited per-pixel disparity information during the over-segmentation. Besides the proposed hyperpixels LF over-segmentation method, a synthetic 4D LF dataset was generated and a modified metric to evaluate over-segmentation angular consistency was proposed to evaluate the proposed method for both dense and sparse LFs. The over-segmentation results were evaluated on various LF datasets, and the results of the proposed methods have shown over-segmentation spatial accuracy and angular consistency that outperform existing methods in most test LF datasets.

The fourth and fifth achievements addressed the third Thesis objective, which was the development of spatially accurate and angularly consistent 4D LF mid-level segmentation. In particular, in the SLFS method [111], detailed in Chapter 5, a 4D LF was represented by a hypergraph based on LF over-segmentation, where each 4D LF segment represented a hypernode in the hypergraph. Afterwards, a semi-supervised approach using classical graph cut was exploited to achieve interactive foreground-background segmentation for dense 4D LFs with the guidance of the user's scribbles. The results have shown competitive performance in terms of segmentation accuracy and angular consistency without relying on accurate disparity maps and by considerably reducing the hypergraph size (i.e., the number of hypernodes that need to be processed).

The fifth achievement continued addressing the third Thesis objective by developing multiple labels mid-level 4D LF segmentation in an unsupervised manner, as detailed in Chapter 6. The proposed method in Chapter 6 did not require the user's scribbles to identify objects from each other or any GT labeled images for training. Instead, it utilizes an unsupervised deep learning approach to optimize the model for each LF by extracting deep features. In this method, a 4D LF was represented as a hypergraph based on hyperpixels over-segmentation where each hypernode was represented by a set of features including color, texture, and disparity. Next, the hypernodes were merged into different labels using a GNN and gradient descent optimization. The results have shown outperforming performance regarding segmentation accuracy and angular consistency compared to benchmark unsupervised segmentation methods in most test LF datasets.

The sixth achievement considered the fourth Thesis objective and tied all the previous chapters together in one editing application (i.e., neural style transfer) presented in Chapter 7. The proposed 4D LF neural style transfer application [112] included steps from the previous chapters, namely, disparity estimation for all 4D LF views and 4D LF over-segmentation. Notice that if we target to apply 4D LF editing for a specific object, the methods presented in Chapter 5 and Chapter 6 can be used to apply object segmentation at first to facilitate indexing each object and ensure angular consistency. However, in Chapter 7, LF editing was applied to the entire scene, not only to specific objects. The results of the proposed method in Chapter 7 have shown outperforming performance when compared to most existing 4D LF neural style transfer methods and produced visually appealing and consistent results across all LF views.

8.2 Future directions

LF imaging still has several challenges that need further investigation, certainly not limited to developing efficient methods for LF segmentation and editing. Examples of other LF aspects that deserve further investigation include: i) Developing efficient capturing and displaying devices that are more affordable and enable the integration of LF technology in our daily lives; and ii) Developing efficient LF coding and rendering algorithms that reduce the storage and bandwidth requirements to facilitate LF usage in various practical applications.

While this Thesis has been focused on advancing specific 4D LF processing and editing tasks, it makes more sense to introduce future directions that continue what has been started in this Thesis. Some limitations and remaining work that deserves to be further improved and investigated include:

- **Development of improved disparity/depth estimation methods that efficiently consider both dense and sparse 4D LFs** – As shown previously, using accurately estimated disparity maps can significantly and positively affect the performance of 4D LF over-segmentation, segmentation and edit propagation. Some remaining limitations of the proposed disparity propagation method in Chapter 2 still need consideration. One limitation is the fact that currently only one reference disparity map is used (which is located in the center of the LF) and warped into all other LF views. Thus, to adaptatively address the best angular location of the reference view according to camera parameters or the scene content, as well as the number of needed reference views, remain for future work. Additionally, more complex refinement and accuracy checking techniques could improve the resulting disparity maps but may affect the average complexity. Using deep learning in LF disparity/depth estimation has shown appealing performance compared to classical methods. However, remaining gaps in the literature include developing one deep learning model to enable estimating disparity/depth maps for all LF views while ensuring angular consistency in dense and sparse 4D LFs. Moreover, handling the domain shift between training and testing synthetic and real 4D LF depth/disparity datasets is another challenge to investigate.
- **Extending the over-segmentation of static 4D LFs to deal with dynamic LFs** – LF videos capture the scene across different viewpoints and time instants. Therefore, LF videos contain a massive amount of data with redundant information in the spatial, angular and temporal dimensions. In this Thesis, it has been shown that applying over-segmentation to static 4D LFs leads to a compact representation that exploits the spatio-angular correlations across LF views, which facilitates subsequent tasks and helps in ensuring angular consistency. Realizing that, one future direction to extend the proposed methods in Chapter 3 and Chapter 4 is to consider dynamic LFs. This can be achieved by grouping similar pixels not only spatially and angularly, but also temporally across frames to enable efficient LF video processing and editing. Moreover, the implementation of the proposed methods can be further optimized by parallelizing some steps which can significantly reduce the required processing time.
- **Inclusion of semantic labels extracted by a pre-trained model in 4D LF processing and editing** – With the advances in deep learning techniques and the huge datasets of 2D images available for training, mature models for object detection/recognition have achieved significant progress with real-time performance. Exploiting off-the-shelf pre-

trained models to recognize objects, e.g., in a reference view, and using the obtained semantic labels as guidance for processing and editing tasks can be a future direction that requires further study. More precisely, one of the remaining limitations in the proposed segmentation method, presented in Chapter 6, is that the number of classes/labels is not automatically estimated based on the LF content before the segmentation. Thus, using a pre-trained model that recognizes objects, can enable automatically estimating the number of objects in each LF and may lead to a better segmentation performance. Moreover, another work direction is to improve the realism of the proposed neural style transfer method, presented in Chapter 7, by first recognizing the objects in the scene and then applying the styles and colors that suit each object across 4D LF views to maintain the semantic meaning (e.g., it is more realistic to stylize the skin of humans by its original color instead of blue).

Besides what has been presented above to improve what has been achieved in this Thesis, here are some other directions that can be continued in future work:

- **Generating large 4D LF datasets and benchmarks for 4D LF mid-level segmentation for domain-specific applications** – In the existing literature, only a few datasets target 4D LF segmentation compared to those available for 2D/3D imaging. Thus, generating new 4D LF datasets for segmentation and benchmarks for specific domains, such as medical LF imaging datasets and self-driving vehicles LF datasets, containing GT label images for dense and sparse LFs is encouraged.
- **Developing tools and software applications that can generate immersive 4D LF content and enable applying interactive editing** – In 2D/3D imaging, there are several user-friendly and interactive tools and applications for content creation and editing, such as Adobe Photoshop [113] and Blender [59]. On the other hand, there are limited tools and software applications that enable 4D LF creation or easy-to-use 4D LF editing. Therefore, developing such tools and software applications can empower individuals to create and edit 4D LFs, and encourage them to exploit 4D LF content in practical applications, such as in marketing and advertising. This direction may also exploit the recent advances in generative artificial intelligence technology to create high-quality 4D LF content.
- **Developing novel metrics for angular consistency evaluation** – In this Thesis, several metrics were used to evaluate 4D LF processing and editing applications. However, most of the used metrics to evaluate processing/editing accuracy or angular

consistency are either extended from the traditional 2D or 3D metrics, rely on GT disparity maps and segmentation labels, or rely on projecting all LF views into a reference view (e.g., the central view) and discarding occluded/non-existing pixels in the reference view. Hence, they may not be suitable for evaluating sparse 4D LFs with large occlusions and may not be able to evaluate real 4D LFs. Thus, proposing novel evaluation metrics that can be used for dense and sparse, real and synthetic 4D LFs in various processing and editing tasks is strongly required.

In conclusion, LF imaging has been a subject of persistent interest in the research community due to its potential to revolutionize various fields. This Thesis has contributed to advancing the state-of-the-art in several LF processing and editing tasks. We hope the remaining challenges and research directions presented in this section will inspire future researchers to embark on a journey of discovery, building upon the groundwork established in this Thesis.

References

[1] "IMMERSIVE | English meaning - Cambridge Dictionary." [Online]. Available: https://dictionary.cambridge.org/dictionary/english/immersive. [Accessed: 06-Jul-2023].

[2] M. Alain, E. Zerman, and C. Ozcinar, "Immersive Imaging Technologies: From Capture to Display," in *ACM Int. Conf. Multimedia*, New York, NY, USA, 2020, pp. 4787–4788.

[3] "Meta Quest VR Headsets, Accessories & Equipment | Meta Quest | Meta Store." [Online]. Available: https://www.meta.com/quest/. [Accessed: 08-Jul-2023].

[4] "Microsoft Store: VR Headsets ." [Online]. Available: https://www.microsoft.com/en-us/store/b/virtualreality. [Accessed: 08-Jul-2023].

[5] "VIVE European Union | Discover Virtual Reality Beyond Imagination." [Online]. Available: https://www.vive.com/eu/. [Accessed: 08-Jul-2023].

[6] "Sony Spatial Reality Display | ELF-SR1." [Online]. Available: https://electronics.sony.com/more/spatial-reality-display/p/elfsr1. [Accessed: 08-Jul-2023].

[7] M. S. Zubairy, "A Very Brief History of Light," in *Optics in Our Time*, Cham: Springer, 2016, pp. 3–24.

[8] E. H. Adelson and J. R. Bergen, "The Plenoptic Function and the Elements of Early Vision," in *Computational Models of Visual Processing*, MIT Press, 1991, pp. 3–20.

[9] J. van der Hooft *et al.*, "A Tutorial on Immersive Video Delivery: From Omnidirectional Video to Holography," *IEEE Commun. Surv. Tutorials*, vol. 25, no. 2, pp. 1336–1375, Mar. 2023.

[10] S. Zhou, T. Zhu, K. Shi, Y. Li, W. Zheng, and J. Yong, "Review of Light Field Technologies," *Vis. Comput. Ind. Biomed. Art*, vol. 4, no. 1, pp. 1–13, Dec. 2021.

[11] C. Jia, F. Shi, M. Zhao, and S. Chen, "Light Field Imaging for Computer Vision: A Survey," *Front. Inf. Technol. Electron. Eng.*, vol. 23, no. 7, pp. 1077–1097, Jul. 2022.

[12] "Lytro Timeline | LightField Forum." [Online]. Available: http://lightfield-forum.com/lytro/lytro-archive/. [Accessed: 24-Jun-2020].

[13] "Raytrix | 3D light field camera technology." [Online]. Available: https://raytrix.de/. [Accessed: 01-May-2021].

[14] G. Wu *et al.*, "Light Field Image Processing: An Overview," *IEEE J. Sel. Top. Signal Process.*, vol. 11, no. 7, pp. 926–954, Oct. 2017.

[15] T. L. T. da Silveira and C. R. Jung, "Omnidirectional Visual Computing: Foundations, Challenges, and Applications," *Comput. Graph.*, vol. 113, pp. 89–101, Jun. 2023.

[16] M. Alain, E. Zerman, C. Ozcinar, and G. Valenzise, "Introduction to Immersive Video Technologies," in *Immersive Video Technologies*, Elsevier, 2023, pp. 3–24.

[17] "Gear 360 | Samsung Business Portugal." [Online]. Available: https://www.samsung.com/pt/business/mobile-accessories/gear-360-c200-sm-c200nzwatph/. [Accessed: 07-Jul-2023].

[18] "Insta360 Store: The Official Store for Insta360 Cameras, Accessories and Services." [Online]. Available: https://store.insta360.com/. [Accessed: 07-Jul-2023].

[19] R. Shafi, W. Shuai, and M. U. Younus, "360-Degree Video Streaming: A Survey of the State of the Art," *Symmetry (Basel).*, vol. 12, no. 9, pp. 1–31, Sep. 2020.

[20] D. Gabor, “A New Microscopic Principle,” *Nature*, vol. 161, no. 4098, pp. 777–778, Dec. 1948.

[21] “Microsoft Mixed Reality - Healthcare.” [Online]. Available: https://www.microsoft.com/en-us/hololens/industry-healthcare. [Accessed: 10-Jul-2023].

[22] “Leia Display System.” [Online]. Available: http://leiadisplay.com/. [Accessed: 15-Jul-2023].

[23] A. Haleem, M. Javaid, and I. Khan, “Holography Applications Toward Medical Field: An Overview,” *Indian J. Radiol. Imaging*, vol. 30, no. 3, pp. 354–361, Jul. 2020.

[24] C. R. Ramachandiran, M. M. Chong, and P. Subramanian, “3D Hologram in Futuristic Classroom: A Review,” *Period. Eng. Nat. Sci.*, vol. 7, no. 2, pp. 580–586, Jul. 2019.

[25] “Mixed Reality Capture Studios for Volumetric Video | Microsoft.” [Online]. Available: https://www.microsoft.com/en-us/mixed-reality/capture-studios. [Accessed: 08-Jul-2023].

[26] “MagiScan AI — 3D scanner app tool for iOS and Android.” [Online]. Available: https://magiscan.app/. [Accessed: 08-Jul-2023].

[27] “Omniverse Platform for OpenUSD Development and Collaboration | NVIDIA.” [Online]. Available: https://www.nvidia.com/en-us/omniverse/. [Accessed: 08-Jul-2023].

[28] B. Mildenhall, P. P. Srinivasan, M. Tancik, J. T. Barron, R. Ramamoorthi, and R. Ng, “NeRF: Representing Scenes as Neural Radiance Fields for View Synthesis,” *Commun. ACM*, vol. 65, no. 1, pp. 99–106, Dec. 2021.

[29] K. Gao, Y. Gao, H. He, D. Lu, L. Xu, and J. Li, “NeRF: Neural Radiance Field in 3D Vision, A Comprehensive Review,” *arXiv Prepr. arXiv2210.00379*, Oct. 2022.

[30] M. Debbagh, “Neural Radiance Fields (NeRFs): A Review and Some Recent Developments,” *arXiv Prepr. arXiv2305.00375*, Apr. 2023.

[31] A. Smolic, “3D Video and Free Viewpoint Video—From Capture to Display,” *Pattern Recognit.*, vol. 44, no. 9, pp. 1958–1968, Sep. 2011.

[32] C. Galea and C. Guillemot, “Denoising of 3D Point Clouds Constructed from Light Fields,” in *IEEE Int. Conf. Acoust., Speech Signal Process. (ICASSP)*, Brighton, UK, 2019, pp. 1882–1886.

[33] H. Farhood, S. Perry, E. Cheng, and J. Kim, “3D Point Cloud Reconstruction From a Single 4D Light Field Image,” in *Opt., Photon. Digit. Technol. Imag. Appl. VI*, Online Only, 2020, p. 1–9.

[34] M. Gond, E. Zerman, S. Knorr, and M. Sjöström, “LFSphereNet: Real Time Spherical Light Field Reconstruction from a Single Omnidirectional Image,” in *the 20th ACM SIGGRAPH Eur. Conf. Vis. Media Prod. (CVMP)*, New York, NY, USA, 2023, pp. 1–10.

[35] A. Gershun, “The Light Field,” *J. Math. Phys.*, vol. 18, no. 1–4, pp. 51–151, Apr. 1939.

[36] M. Levoy and P. Hanrahan, “Light Field Rendering,” in *23rd annu. conf. Comput. graph. interactive tech.*, New York, NY, USA, 1996, pp. 31–42.

[37] S. J. Gortler, R. Grzeszczuk, R. Szeliski, and M. F. Cohen, “The Lumigraph,” in *23rd Annu. Conf. Comput. Graph. Interactive Techn.*, New Orleans, LA, USA, 1996, pp. 43–54.

[38] 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. 7–55, Mar. 1987.

[39] V. Vaish and A. Adams, “The (New) Stanford Light Field Archive,” *Stanford*, 2008. [Online]. Available: http://lightfield.stanford.edu/acq.html. [Accessed: 24-Jun-2020].

[40] B. Wilburn *et al.*, “High Performance Imaging using Large Camera Arrays,” in *ACM SIGGRAPH*, New

York, NY, USA, 2005, pp. 765–776.

[41] N. Sabater *et al.*, “Dataset and Pipeline for Multi-view Light-Field Video,” in *IEEE Conf. Comput. Vis. Pattern Recog. (CVPR)*, Honolulu, HI, USA, 2017, pp. 1743–1753.

[42] J. Trottnow *et al.*, “The Potential of Light Fields in Media Productions,” in *SIGGRAPH Asia Tech. Briefs*, New York, NY, USA, 2019, pp. 71–74.

[43] M. Broxton *et al.*, “Immersive Light Field Video with a Layered Mesh Representation,” *ACM Trans. Graph.*, vol. 39, no. 4, pp. 1–15, Jul. 2020.

[44] R. S. Overbeck, D. Erickson, D. Evangelakos, M. Pharr, and P. Debevec, “A System for Acquiring, Processing, and Rendering Panoramic Light Field Stills for Virtual Reality,” *ACM Trans. Graph.*, vol. 37, no. 6, pp. 1–15, Dec. 2018.

[45] K. Venkataraman *et al.*, “PiCam: An Ultra-thin High Performance Monolithic Camera Array,” *ACM Trans. Graph.*, vol. 32, no. 6, pp. 1–13, Nov. 2013.

[46] G. Lippmann, “Épreuves Réversibles Donnant la Sensation du Relief,” *J. Phys. Théorique Appliquée*, vol. 7, no. 1, pp. 821–825, Jan. 1908.

[47] E. H. Adelson and J. Y. A. Wang, “Single Lens Stereo with a Plenoptic Camera,” *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 14, no. 2, pp. 99–106, Feb. 1992.

[48] R. Ng, M. Levoy, M. Brédif, G. Duval, M. Horowitz, and P. Hanrahan, “Light Field Photography with a Hand-held Plenoptic Camera,” Ph.D. diss., Stanford university, 2005.

[49] “Lytro Archive | LightField Forum.” [Online]. Available: http://lightfield-forum.com/lytro/lytro-archive/. [Accessed: 29-May-2023].

[50] “Raytrix | light field cameras.” [Online]. Available: https://raytrix.de/products/. [Accessed: 29-May-2023].

[51] C. Hahne, “The Standard Plenoptic Camera: Applications of a Geometrical Light Field Model,” Ph.D. diss., University of Bedfordshire, 2016.

[52] A. Lumsdaine and T. Georgiev, “The Focused Plenoptic Camera,” in *IEEE Int. Conf. Comput. Photogr. (ICCP)*, San Francisco, CA, USA, 2009, pp. 1–8.

[53] T. Georgiev, “New Results on the Plenoptic 2.0 Camera,” in *2009 Conf. Signals, Syst. Comput.*, Pacific Grove, CA, USA, 2009, pp. 1243–1247.

[54] T. Georgiev and A. Lumsdaine, “Depth of Field in Plenoptic Cameras,” in *Eurographics 2009*, Munich, German, 2009, pp. 1–4.

[55] T. Georgiev and A. Lumsdaine, “The Multifocus Plenoptic Camera,” in *Digit. Photogr. VIII*, Burlingame, California, USA, 2012, p. 1–11.

[56] C. Perwass and L. Wietzke, “Single Lens 3D-camera with Extended Depth-of-field,” in *Human Vis. Electron. Imag. XVII*, Burlingame, CA, USA, 2012, p. 1–15.

[57] “Machine Vision - K|Lens.” [Online]. Available: https://www.k-lens.de/. [Accessed: 15-Dec-2023].

[58] “K|Lens One (Canceled) by K|Lens » Project cancelation — Kickstarter.” [Online]. Available: https://www.kickstarter.com/projects/k-lens-one/k-lens-one/posts/3416668. [Accessed: 14-Jul-2023].

[59] “blender.org - Home of the Blender project - Free and Open 3D Creation Software,” *blender.org*, 2018. [Online]. Available: https://www.blender.org/. [Accessed: 24-Sep-2021].

[60] K. Honauer, O. Johannsen, D. Kondermann, and B. Goldluecke, “A Dataset and Evaluation Methodology

for Depth Estimation on 4D Light Fields," in *13th Asian Conf. Comput. Vis. (ACCV)*, Taipei, Taiwan, 2016, pp. 19–34.

[61] H. Sheng, R. Cong, D. Yang, R. Chen, S. Wang, and Z. Cui, "UrbanLF: A Comprehensive Light Field Dataset for Semantic Segmentation of Urban Scenes," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 32, no. 11, pp. 7880–7893, Nov. 2022.

[62] J. Cao *et al.*, "Real-Time Neural Light Field on Mobile Devices," in *2023 IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Vancouver, BC, Canada, 2023, pp. 8328–8337.

[63] Z. Chen, T. Funkhouser, P. Hedman, and A. Tagliasacchi, "MobileNeRF: Exploiting the Polygon Rasterization Pipeline for Efficient Neural Field Rendering on Mobile Architectures," *arXiv:2208.00277*, pp. 16569–16578, Jul. 2022.

[64] Z. Jia, B. Wang, and C. Chen, "Drone-NeRF: Efficient NeRF based 3D Scene Reconstruction for Large-Scale Drone Survey," *Image Vis. Comput.*, vol. 143, pp. 1–15, Mar. 2024.

[65] Y. Wang *et al.*, "Disentangling Light Fields for Super-Resolution and Disparity Estimation," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 45, no. 1, pp. 425–443, Jan. 2023.

[66] D. Bonatto *et al.*, "Multiview from Micro-lens Image of Multi-focused Plenoptic Camera," in *2021 Int.Conf. 3D Immersion (IC3D)*, Brussels, Belgium, 2021, pp. 1–8.

[67] S. Shen, S. Xing, X. Sang, B. Yan, and Y. Chen, "Virtual Stereo Content Rendering Technology Review for Light-field Display," *Displays*, vol. 76, p. 102320, Jan. 2023.

[68] F. Zhou, W. Qiao, and L. Chen, "Fabrication Technology for Light Field Reconstruction in Glasses-free 3D Display," *J. Inf. Disp.*, vol. 24, no. 1, pp. 13–29, Jan. 2023.

[69] S. Sawada and H. Kakeya, "Integral Volumetric Imaging using Decentered Elemental Lenses," *Opt. Express*, vol. 20, no. 23, pp. 25902–25913, Nov. 2012.

[70] H. Watanabe, T. Omura, N. Okaichi, M. Kano, H. Sasaki, and J. Arai, "Full-parallax Three-dimensional Display based on Light Field Reproduction," *Opt. Rev.*, vol. 29, no. 4, pp. 366–374, Aug. 2022.

[71] "FOVi3D Shows Light Field Displays – Display Daily." [Online]. Available: https://displaydaily.com/fovi3d-shows-light-field-displays/. [Accessed: 19-Feb-2024].

[72] "FOVI3D | 3D Collaboration." [Online]. Available: https://www.fovi3d.com/. [Accessed: 15-Jul-2023].

[73] "Looking Glass Display." [Online]. Available: https://lookingglassfactory.com/product-overview. [Accessed: 15-Jul-2023].

[74] "Apple Vision Pro - Apple." [Online]. Available: https://www.apple.com/apple-vision-pro/. [Accessed: 09-May-2024].

[75] "CREAL light-field display." [Online]. Available: https://creal.com/technology/. [Accessed: 14-Jul-2023].

[76] "Magic Leap | Create innovative solutions for complex problems with the most immersive enterprise AR device." [Online]. Available: https://www.magicleap.com/magic-leap-2. [Accessed: 06-Feb-2024].

[77] D. Lanman and D. Luebke, "Near-eye Light Field Displays," *ACM Trans. Graph.*, vol. 32, no. 6, pp. 1–10, Nov. 2013.

[78] K. Bang, Y. Jo, M. Chae, and B. Lee, "LensIet VR: Thin, Flat and Wide-FOV Virtual Reality Display Using Fresnel Lens and LensIet Array," *IEEE Trans. Vis. Comput. Graph.*, vol. 27, no. 5, pp. 2545–2554, May 2021.

[79] T. Wang *et al.*, “Light Field Depth Estimation: A Comprehensive Survey from Principles to Future,” *High-Confidence Comput.*, vol. 4, no. 1, p. 1–16, Mar. 2024.

[80] D. Yang, T. Zhu, S. Wang, S. Wang, and Z. Xiong, “LFRSNet: A Robust Light Field Semantic Segmentation Network Combining Contextual and Geometric Features,” *Front. Environ. Sci.*, vol. 10, p. 1–14, Oct. 2022.

[81] R. Cong, D. Yang, R. Chen, S. Wang, Z. Cui, and H. Sheng, “Combining Implicit-Explicit View Correlation for Light Field Semantic Segmentation,” in *IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Vancouver, Canada, 2023, pp. 9172–9181.

[82] R. Cong *et al.*, “Multimodal Perception Integrating Point Cloud and Light Field for Ship Autonomous Driving,” *IEEE Trans. Intell. Transp. Syst.*, early access, pp. 1–13, Apr. 2024.

[83] M. Zhao *et al.*, “A Survey for Light Field Super-resolution,” *High-Confidence Comput.*, vol. 4, no. 1, p. 1–12, Mar. 2024.

[84] Z. Cheng, Z. Xiong, C. Chen, and D. Liu, “Light Field Super-resolution: A Benchmark,” in *IEEE Conf. Comput Vis. Pattern Recognit. (CVPR)*, Long Beach, CA, USA, 2019, pp. 1804–1813.

[85] A. Wafa and P. Nasiopoulos, “Light Field GAN-based View Synthesis using Full 4D Information,” in *ACM SIGGRAPH Eur. Conf. Vis. Media Prod (CVMP)*, New York, NY, USA, 2022, pp. 1–7.

[86] M. Ziegler *et al.*, “Multi-camera System for Depth Based Visual Effects and Compositing,” in *ACM SIGGRAPH Eur. Conf. Vis. Media Prod (CVMP)*, New York, NY, USA, 2015, pp. 1–10.

[87] Lytro, “cinamatog | LightField Forum.” [Online]. Available: http://lightfield-forum.com/lytro/lytro-timeline/. [Accessed: 29-Apr-2021].

[88] M. Suhail, C. Esteves, L. Sigal, and A. Makadia, “Light Field Neural Rendering,” in *IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, New Orleans, LA, USA, 2022, pp. 8259–8269.

[89] M. Levoy, R. Ng, A. Adams, M. Footer, and M. Horowitz, “Light Field Microscopy,” *ACM Trans. Graph.*, vol. 25, no. 3, pp. 924–934, Jul. 2006.

[90] E. Longo *et al.*, “Flexible Plenoptic X-ray Microscopy,” *Photonics*, vol. 9, no. 2, p. 1–13, Feb. 2022.

[91] O. Bimber and D. Schedl, “Light-Field Microscopy: A Review,” *J. Neurol. Neuromedicine*, vol. 4, no. 1, pp. 1–6, Jan. 2019.

[92] C. Yi, L. Zhu, D. Li, and P. Fei, “Light Field Microscopy in Biological Imaging,” *J. Innov. Opt. Health Sci.*, vol. 16, no. 1, pp. 1–15, Jan. 2023.

[93] B. S. R. Santos and D. Trevisan, “Overview and Tendencies of Augmented Reality Applications in Medicine,” in *Symp. Virtual Augment. Real.,* New York, NY, USA, 2023, pp. 26–37.

[94] R. Raghavendra, K. B. Raja, and C. Busch, “Exploring the Usefulness of Light Field Cameras for Biometrics: An Empirical Study on Face and Iris Recognition,” *IEEE Trans. Inf. Forensics Secur.*, vol. 11, no. 5, pp. 922–936, May 2016.

[95] K. Fu, Y. Jiang, G. P. Ji, T. Zhou, Q. Zhao, and D. P. Fan, “Light Field Salient Object Detection: A review and Benchmark,” *Comput. Vis. Media*, vol. 8, no. 4, pp. 509–534, 2022.

[96] M. Wang *et al.*, “Visual Object Tracking Based on Light-Field Imaging in the Presence of Similar Distractors,” *IEEE Trans. Ind. Informatics*, vol. 19, no. 3, pp. 2705–2716, Mar. 2023.

[97] R. S. Higa, Y. Iano, R. B. Leite, R. F. L. Chavez, and R. Arthur, “Employing Light Field Cameras in Surveillance: An Analysis of Light Field Cameras in a Surveillance Scenario,” *3D Res.*, vol. 5, no. 1, pp.

1–11, Mar. 2014.

[98] “Project Starline: Feel like you’re there, together.” [Online]. Available: https://blog.google/technology/research/project-starline/. [Accessed: 28-Jul-2023].

[99] H. Wang *et al.*, “A Survey on the Metaverse: The State-of-the-Art, Technologies, Applications, and Challenges,” *IEEE Internet Things J.*, vol. 10, no. 16, pp. 14671–14688, Aug. 2023.

[100] P. A. Kara, R. R. Tamboli, V. K. Adhikarla, T. Balogh, M. Guindy, and A. Simon, “Connected without Disconnection: Overview of Light Field Metaverse Applications and Their Quality of Experience,” *Displays*, vol. 78, p. 1–12, Jul. 2023.

[101] C. Conti, L. D. Soares, and P. Nunes, “Dense Light Field Coding: A Survey,” *IEEE Access*, vol. 8, pp. 49244–49284, Mar. 2020.

[102] “ISO/IEC 21794-2:2021 - Information technology — Plenoptic image coding system (JPEG Pleno) — Part 2: Light field coding.”

[103] H. Amirpour, C. Guillemot, M. Ghanbari, and C. Timmerer, “Advanced Scalability for Light Field Image Coding,” *IEEE Trans. Image Process.*, vol. 31, pp. 7435–7448, Nov. 2022.

[104] N. Khan, M. H. Kim, and J. Tompkin, “View-consistent 4D Light Field Depth Estimation,” in *Proc. Brit. Mach. Vis. Conf. (BMVC),* Sep. 2020, pp. 1–13.

[105] 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.

[106] B. Sasmal and K. G. Dhal, “A Survey on the Utilization of Superpixel Image for Clustering Based Image Segmentation,” *Multimed. Tools Appl.*, vol. 82, no. 23, pp. 35493–35555, Sept. 2023.

[107] I. Borlido *et al.*, “Superpixel Segmentation: From Theory to Applications*,” in *36th SIBGRAPI Conf. Graph., Patterns Imag. (SIBGRAPI)*, Rio Grande, Brazil, 2023, pp. 258–263.

[108] I. B. Barcelos, F. D. C. Belém, L. D. M. João, Z. K. G. Do Patrocínio, A. X. Falcão, and S. J. F. Guimarães, “A Comprehensive Review and New Taxonomy on Superpixel Segmentation,” *ACM Comput. Surv.*, vol. 56, no. 8, pp. 1–39, Apr. 2024.

[109] M. Hamad, C. Conti, P. Nunes, and L. D. Soares, “ALFO: Adaptive Light Field Over-Segmentation,” *IEEE Access*, vol. 9, pp. 131147–131165, Sep. 2021.

[110] M. Hamad, C. Conti, P. Nunes, and L. D. Soares, “Hyperpixels: Flexible 4D Over-Segmentation for Dense and Sparse Light Fields,” *IEEE Trans. Image Process.*, vol. 32, pp. 3790–3805, Jul. 2023.

[111] M. Hamad, C. Conti, A. M. De Almeida, P. Nunes, and L. D. Soares, “SLFS: Semi-supervised Light-field Foreground-background Segmentation,” in *2021 Telecoms Conference, ConfTELE 2021*, Leiria, Portugal, 2021, pp. 1–6.

[112] M. Hamad, C. Conti, P. Nunes, and L. D. Soares, “View-consistent 4D Light Field Style Transfer using Neural Networks and Over-segmentation,” in *IEEE Imag., Video, Multidimensional Signal Process. (IVMSP)*, Nafplio, Greece, 2022, pp. 1–5.

[113] “Official Adobe Photoshop - Photo & design software.” [Online]. Available: https://www.adobe.com/il_en/products/photoshop.html. [Accessed: 16-Sep-2023].