Scene analysis and view rendering from light fields

Pierre David

UNIVERSITE BRETAGNE LOIRE / MATHSTIC

UNIVERSITÉ DE RENNES 1

THÈSE DE DOCTORAT DE

L'UNIVERSITÉ DE RENNES 1
COMUE UNIVERSITÉ BRETAGNE LOIRE

ÉCOLE DOCTORALE N° 601
Mathématiques et Sciences et Technologies de l'Information et de la Communication
Spécialité : *Signal, Image, Vision*

Par
Pierre DAVID

Scene analysis and view rendering from light fields

Thèse présentée et soutenue à Rennes, le 30 juin 2020
Unité de recherche : Sirocco, Inria Rennes - Bretagne Atlantique

Rapporteurs avant soutenance :

Peter SCHELKENS — Professeur, Vrije Universiteit Brussel
Mårten SJÖSTRÖM — Professeur, Mittuniversitetet

Composition du Jury :

Présidente :	Luce MORIN	Professeur, INSA Rennes
Examinateur :	Frédéric DUFAUX	Directeur de recherche, CNRS, CentraleSupélec
Dir. de thèse :	Christine GUILLEMOT	Directrice de recherche, Inria Rennes - Bretagne Atlantique
Co-dir. de thèse :	Mikaël LE PENDU	Chercheur, Trinity College Dublin

SCENE ANALYSIS AND VIEW RENDERING FROM LIGHT FIELDS

PIERRE DAVID

2020

I knew exactly what to do, but in a much more real sense I had no idea what to do.

— Michael Scott, *The Office*

ACKNOWLEDGMENTS

I would first like to thank the members of the jury for accepting to be part of it: Mårten Sjöström, Peter Schelkens, Luce Morin and Frédéric Dufaux. Your feedback on this manuscript has been very helpful and the discussion during the defense was very interesting.

I would also like to thank my PhD supervisor, Christine, for her guidance and patience during these three and a half years. Many thanks to Mikaël for is many insightful advices. You two helped me a lot during my research while also giving me a lot of freedom. I am very proud of the work we did together.

In addition, I want to thank my friends and colleagues at Inria: Elian, Navid, Simon, Fatma, Xiaoran, Jinglei, Maja, Thomas, Aline, Ehsan, Nam, Pierre, Maï, Laurent, Christian, Matthieu, Thierry, Fatemeh, Lara and everyone else. I was very happy to part of this team and to talk with everyone of you during coffee break and afterwork events.

Finally, I must express my very profound gratitude to my family. I would not have been able to complete my thesis without your support and kindness. Last, but not least, I would like to thank Mira for being part of my life. I love you.

CONTENTS

Résumé en français 15

Opening

Introduction 23
- Context . . . 23
- Motivations and Contributions . . . 24
- Thesis Structure . . . 25
- List of Publications . . . 26

I LIGHT FIELD IMAGING

1 BACKGROUND ON LIGHT FIELD IMAGING 29
- 1.1 History . . . 29
- 1.2 Light field definition . . . 30
- 1.3 Capturing light fields . . . 37
- 1.4 Decoding light fields . . . 40

2 FROM RAW DATA TO LIGHT FIELDS 45
- 2.1 Introduction . . . 45
- 2.2 Description of the decoding pipeline . . . 46
- 2.3 Flaws and improvements of the pipeline . . . 48
- 2.4 Experiments on real lenslet images . . . 54
- 2.5 Summary . . . 54

II SCENE FLOW ESTIMATION AND TEMPORAL INTERPOLATION

3 STATE OF THE ART 61
- 3.1 Introduction . . . 61
- 3.2 Scene depth estimation from light fields . . . 62
- 3.3 Optical flow estimation from videos . . . 63
- 3.4 Scene flow estimation . . . 65
- 3.5 Video frame interpolation . . . 66

4 LOCAL 4D AFFINE MODEL FOR SCENE FLOW 69
- 4.1 Introduction . . . 69
- 4.2 4D Affine Model . . . 70
- 4.3 Estimating the model parameters . . . 74
- 4.4 Applications . . . 78
- 4.5 Evaluation . . . 82
- 4.6 Summary . . . 97

5 ANGULARLY CONSISTENT FRAME INTERPOLATION 99
- 5.1 Introduction . . . 99
- 5.2 Light field frame interpolation method . . . 100

5.3 Experiments 103
5.4 Summary 107

Closing
General Conclusion 113
Thesis Summary 113
Future Work and Perspectives 114

Bibliography 117

LIST OF FIGURES

Figure 1.1 Sketch from Lippmann's presentation 29
Figure 1.2 Geometrical representation of the plenoptic function 31
Figure 1.3 Geometrical representation of the light field function 32
Figure 1.4 Alternative representation of the light field function 33
Figure 1.5 Alternative representation of the light field function 33
Figure 1.6 View and epipolar plane images of a light field taken from [20] 34
Figure 1.7 Sketch of a light field display taken from [23] 35
Figure 1.8 Example of post-capture image rendering 36
Figure 1.9 Illustration of the depth estimation principle using slopes in EPI 36
Figure 1.10 Optical configuration of an array of cameras 37
Figure 1.11 Raw images taken by an array of cameras 37
Figure 1.12 Optical configuration of a plenoptic camera 1.0 38
Figure 1.13 Raw image taken by a plenoptic camera 1.0 39
Figure 1.14 Optical configuration of a plenoptic camera 2.0 39
Figure 1.15 Raw image taken by a plenoptic camera 2.0 39
Figure 1.16 Pinhole model 40
Figure 2.1 Pipeline for extracting views from a raw lenslet image [36] . . 46
Figure 2.2 Kernels in [42] 47
Figure 2.3 Misalignment between the sensor and the lenslets 47
Figure 2.4 Extraction of a given view from a lenslet image 48
Figure 2.5 From a synthetic light field to a lenslet image 49
Figure 2.6 Lenslet image of Butterfly [27] 50
Figure 2.7 Assessment of our demosaicing method on a lenslet image . . 50
Figure 2.8 White lenslet image guided demosaicing 51
Figure 2.9 Assessment of our alignment method on a synthetic lenslet image 53
Figure 2.10 Visual comparisons on synthetic light fields 55
Figure 2.11 Visual comparisons on Lytro 1 light fields 56
Figure 2.12 Visual comparisons on Lytro Illum light fields 57
Figure 3.1 Different ways of estimating depth with a light field 62
Figure 3.2 Optical flow definition 64
Figure 3.3 Different ways of doing temporal interpolation 66
Figure 4.1 Block diagram of our scene flow method 70
Figure 4.2 Projections of a 3D scene point on a light field 71
Figure 4.3 Sparse estimation of scene flow. 79
Figure 4.4 Initialization of the scene flow using a deep optical flow method 81
Figure 4.5 Grid search for the sparse-to-dense interpolation 84
Figure 4.6 Grid search for the scene flow regularization 84
Figure 4.7 Cost evolution of the RANSAC model for 10 iterations 85

Figure 4.8 Visual comparison on *Bamboo2 clean* 88
Figure 4.9 Visual comparison on *Temple1 final* 89
Figure 4.10 Visual comparison on a real light field 90
Figure 4.11 Visual comparison of the model validation on *Bamboo2 final* . 94
Figure 4.12 Visual comparison of the model validation on *Temple1 final* . . 94
Figure 5.1 Block diagram of our interpolation method. 100
Figure 5.2 Visual comparison on Sintel dataset 106
Figure 5.3 Backwarp variance map σ^2 for each method 107
Figure 5.4 Visual comparison of our interpolation method with [86, 88, 91] 108

LIST OF TABLES

Table 4.1 EPE of estimated optical flow for all pixels 86
Table 4.2 MAE of estimated disparity for all pixels 87
Table 4.3 MAE of estimated disparity variation for all unoccluded pixels 87
Table 4.4 EPE of estimated optical flow for all pixels 91
Table 4.5 RMSE of estimated disparities for all pixels 92
Table 4.6 Validation of the affine model according to the scene 93
Table 4.7 Influence of occlusions on the affine model 95
Table 4.8 Influence of the motion amplitude on the affine model 95
Table 4.9 Influence of the disparity on the affine model 95
Table 4.10 MSE of estimated optical flow and disparity 97
Table 5.1 PSNR of synthesized light field for all views 104
Table 5.2 SSIM of synthesized light field for all views 105
Table 5.3 LFEC of synthesized light field 105

RÉSUMÉ EN FRANÇAIS

CONTEXTE

Depuis son invention en 1826 par Nicéphore Niépce, la photographie s'est réinventée à de nombreuses reprises en termes d'applications, mais aussi de procédés. Du bitume de Judée au capteur photographique, en passant par les halogénures d'argent, le support d'enregistrement a beaucoup évolué au cours du XXe siècle. Cependant, la photographie classique ne permet de capturer la scène que d'un seul point de vue. Cela peut être contraignant car cela limite l'immersion du spectateur, qui est privé de plusieurs indices visuels qui sont souvent utilisés pour percevoir la profondeur dans les scènes réelles : parallaxe, accommodation, stéréopsie et convergence.

Pour remédier à cela, de nouvelles modalités d'imagerie sont apparues ces dernières années. L'une des modalités les plus prometteuses est le *champ de lumière*, qui décrit la lumière qui circule dans une scène comme un champ de rayons lumineux. On peut aussi le considérer comme un ensemble de photographies d'une scène prises simultanément à partir de différents points de vue. En acquérant et en affichant des champs de lumière, un observateur est capable de percevoir la scène comme s'il y était. Dans le domaine de la vision par ordinateur, les différents points de vue permettent généralement d'estimer la disparité et donc la profondeur. Enfin, des photographies classiques peuvent être générées à partir de champs de lumière où le point de vue, l'ouverture numérique et le plan de mise au point peuvent être modifiés.

L'acquisition de champs de lumière se fait généralement avec deux types d'appareils : (a) avec une matrice de caméras régulièrement disposées sur un même plan, (b) avec une caméra plénoptique qui est composée d'un objectif principal, d'une grille de micro-lentilles et d'un capteur. Au début, les dispositifs d'acquisition de champs de lumière mis au point enregistraient des champs de lumière fixes. Cependant, étant donné que les capteurs utilisés dans ces dispositifs sont les mêmes que ceux des caméras classiques (c'est-à-dire CMOS ou CCD) et si les bandes passantes des appareils sont suffisamment larges, il est possible d'acquérir des champs de lumière d'une scène à différents moments. À cet égard, plusieurs jeux de données vidéos ont été récemment publiés, offrant une grande diversité de champs de lumière : synthétiques (générés avec des logiciels de synthèse d'images), enregistrés avec une matrice de caméras ou avec une caméra plénoptique.

L'extraction d'un champ de lumière utilisable à partir de ces appareils peut cependant s'avérer difficile car les deux types de dispositif nécessitent un traitement post-acquisition lourd : démosaïquage, dévignettage, étalonnage, etc. Une partie importante de la littérature sur les champs de lumière est donc consacrée à cette question. Une autre partie de la recherche se concentre sur l'estimation de la profondeur et l'interpolation

de points de vue. Ces tâches sont des étapes clés pour tout schéma de compression ou rendu post-acquisition. Développer des méthodes efficaces de compression est en outre critique pour les champs de lumière car ce sont des objets très redondants et de grande dimension, d'autant plus lorsque l'on considère des vidéos de champs de lumières.

MOTIVATIONS ET CONTRIBUTIONS

L'extraction d'un champ de lumière à partir d'une image plénoptique brute avec la meilleure qualité possible est cruciale pour de nombreuses applications. Un champ de lumière mal extrait aura un impact négatif sur les performances en matière d'estimation de la profondeur, d'interpolation des vues et de traitements post-acquisition. Par conséquent, chaque étape de l'extraction doit être soigneusement étudiée. Il faut notamment tenir compte de la structure sous-jacente donnée à l'image brute par le réseau de micro-lentilles d'une caméra plénoptique. Ceci nous amène à la première problématique de cette thèse :

1. Comment extraire les vues d'un champ de lumière à partir des données du capteur d'une caméra plénoptique, en tenant compte de la structure lenticulaire de l'image brute ?

Pour répondre à cette question, nous avons étudié les défauts du processus d'extraction le plus utilisé, étape par étape. Nous avons vu comment certaines étapes ne prennent pas en compte la structure lenticulaire de l'image brute et comment cela produit un champ de lumière avec des artefacts fantômes (c'est-à-dire que les vues périphériques du champ de lumière sont mélangées avec d'autres vues). Sur la base de ce constat, nous proposons quelques améliorations qui tiennent compte de la structure lenticulaire en utilisant une image blanche plénoptique pour guider les étapes d'interpolation. Nous montrons comment la méthode proposée permet d'éliminer les artefacts fantômes.

Une fois que les champs de lumière sont extraits des caméras plénoptiques ou des matrices de caméras, une utilisation courante consiste à récupérer des informations géométriques. De nombreuses méthodes ont déjà été proposées pour estimer des cartes de profondeur à partir de champs de lumière fixes, mais très peu de recherches ont été menées sur l'analyse du mouvement à partir de vidéos de champs de lumière. Dans la littérature de la vision par ordinateur, les concepts de *flux optique* et de *flux de scène* sont souvent utilisés pour décrire le mouvement apparent dans une scène. Le flux optique est un champ vectoriel qui décrit le mouvement apparent en 2D de chaque pixel du capteur entre deux instants. Le flux de scène est une extension du flux optique dans laquelle des informations de profondeur sont ajoutées, i. e., l'estimation de la profondeur et de la variation de la profondeur. Intuitivement, les vidéos de champ de lumière doivent être appropriées pour estimer un flux de scène : chaque vue de la vidéo de champs de lumière peut être considérée comme une vidéo classique à laquelle on peut appliquer des méthodes de flux optique standard. Pour l'estimation de la profondeur, nous pourrions utiliser des paires de vues comme des images stéréoscopiques et appliquer des méthodes classiques d'estimation de disparité. Cependant, cela soulèverait deux problèmes : pre-

mièrement, la distance entre chaque point de vue du champ de lumière pourrait être trop petite pour utiliser des méthodes stéréoscopiques classiques, deuxièmement les estimations du flux de scène pourraient être globalement incohérentes pour l'ensemble du champ de lumière. C'est ce qui nous amène à notre deuxième point :

2. Comment estimer le flux de scène d'une vidéo de champs de lumière qui soit à la fois précis et cohérent entre chaque vue (i. e., *angulairement cohérent*) ?

Nous proposons un modèle affine 4D local pour décrire le flux de scène d'une vidéo de champs de lumière. Pour l'appliquer, nous divisons le champ de lumière en sous-ensembles et estimons les paramètres du modèle pour chaque sous-ensemble avec des estimations initiales du flux de scène. Le modèle renforce alors la cohérence du flux de scène dans le champ de lumière. Nous montrons deux applications de ce modèle : une méthode d'interpolation "épars-vers-dense" où le flux de scène initial est estimé sur un sous-ensemble limité de rayons, une méthode de régularisation où l'estimation du flux de scène initial est dense mais incohérent au sein du champ de lumière. Cette dernière application nous permet d'utiliser des méthodes d'apprentissage profond sur chaque vue de champ de lumière, puis de régulariser les estimations afin d'éliminer les valeurs aberrantes et les incohérences angulaires.

Une application habituelle des méthodes d'estimation du mouvement consiste à interpoler une image intermédiaire à partir de deux images consécutives dans une vidéo. Cette interpolation peut être utile pour la vidéo de champs de lumière, car certaines caméras plénoptiques vidéos ont une très faible fréquence d'images. De plus, comme la compression est cruciale pour le traitement des champs de lumière, la capacité à générer une vidéo de champs de lumière entière à partir de quelques champs de lumière est particulièrement intéressante. C'est de là que découle notre dernière problématique :

3. En utilisant notre estimation du flux de scène, comment pouvons-nous générer un champ de lumière intermédiaire cohérent à partir de deux champs de lumière consécutifs ?

Dans ce contexte, nous proposons une nouvelle méthode d'interpolation de vidéo de champs de lumière qui soit cohérente angulairement et temporellement. La cohérence angulaire est contrainte par la méthode d'estimation du flux de scène susmentionnée et la cohérence temporelle est renforcée par un réseau de neurones entraîné sur des vues individuelles.

STRUCTURE DE LA THÈSE

La thèse est organisée de la manière suivante :

Le Chapitre 1 définit formellement le concept de champ de lumière. Tout d'abord, l'histoire des champs de lumière est résumée. Ensuite, nous montrons comment le champ de lumière est dérivé de la fonction plénoptique. Différentes façons de visualiser les champs de lumière sont présentées et des utilisations habituelles des champs de lumière sont données. Ensuite, nous détaillons les différentes façons d'acquérir les champs de lumière : matrice de caméras et caméra plénoptique.

Enfin, nous présentons les différentes méthodes qui ont été proposées pour extraire un champ de lumière à partir de matrices de caméras et de caméras plénoptiques.

Le Chapitre 2 propose des améliorations sur l'un des processus d'extraction les plus utilisés pour les caméras plénoptiques. Tout d'abord, nous analysons et décrivons les défauts de ce processus. Pour mieux identifier les différentes sources d'artefacts, notre analyse est effectuée en générant des images plénoptiques brutes idéales à partir de champs de lumière synthétique et en les utilisant comme entrées du processus de décodage. Ensuite, nous détaillons une nouvelle méthode de démosaïquage basée sur les images blanches plénoptiques fournies qui servent de guide. En outre, nous montrons que ce type d'interpolation guidée peut être utile à d'autres étapes du pipeline de décodage. Enfin, nous évaluons la qualité de vues extraites de champs de lumière synthétiques et réels via des comparaisons visuelles ainsi que des mesures objectives.

Le Chapitre 3 donne une vue d'ensemble sur plusieurs questions de vision par ordinateur liées à l'estimation du flux de scène. Plus précisément, nous présentons d'abord les différents types de méthodes qui ont été proposées pour estimer la profondeur à partir de champs de lumière. Ensuite, nous passons en revue une partie de la littérature sur l'estimation de flux optique. Ensuite, différentes publications sur l'estimation du flux de scène sont discutées. Enfin, nous détaillons plusieurs méthodes pour l'interpolation temporelle des vidéos.

Le Chapitre 4 propose une nouvelle méthode d'estimation du flux de scène à partir de vidéos de champs de lumière. Nous introduisons d'abord un modèle local, affine et 4D pour représenter les flux de scène, en tenant compte de la géométrie épipolaire des champs de lumière. Les paramètres du modèle sont estimés par sous-ensembles de rayons du champs de lumière. Ils sont obtenus en ajustant le modèle sur des estimations initiales de mouvement et de disparité obtenues par des techniques d'estimation de flux optique 2D. Nous montrons d'abord que le modèle peut être utilisé pour densifier des estimations de flux calculés sur un sous-ensemble restreint de rayons. Nous démontrons ensuite que le modèle est également très efficace pour l'estimation des flux de scène à partir de flux optiques 2D. Le modèle régularise les flux optiques et les cartes de disparité, et interpole les valeurs de variation de disparités dans les régions occultées. Le modèle proposé nous permet de bénéficier des méthodes d'estimation des flux optiques 2D basées sur la théorie de l'apprentissage profond tout en assurant la cohérence de la géométrie épipolaire des flux de scène dans les quatre dimensions du champ de lumière.

Le Chapitre 5 aborde le problème de l'interpolation temporelle des vidéos de champ de lumière en utilisant des flux de scènes denses. Étant donné des champs de lumière à deux instants, l'objectif est d'interpoler un champ de lumière intermédiaire pour former une séquence vidéo de champs de lumière cohérentes spatialement, angulairement et temporellement. Nous commençons par calculer les flux de scène bidirectionnels angulairement cohérents entre les deux champs de lumière d'entrée. Nous utilisons ensuite les flux optiques et les deux champs de lumière

comme entrées d'un réseau de neurones convolutifs qui synthétise indépendamment les vues du champ de lumière à un moment intermédiaire. Afin de mesurer la cohérence angulaire d'un champ de lumière, nous proposons une nouvelle métrique basée sur la géométrie épipolaire. Les résultats expérimentaux montrent que la méthode proposée produit des champs de lumière qui sont angulairement cohérents tout en conservant une cohérence temporelle et spatiale similaire à celle des méthodes d'interpolation temporelle de vidéos les plus avancées, qui elles ne permettent pas de s'assurer de la cohérence angulaire.

LISTE DES PUBLICATIONS

La plupart des contributions présentées ont été publiées dans les communications de conférences et revues internationales suivantes :

- Pierre David, Mikaël Le Pendu, Christine Guillemot. « White Lenslet Image Guided Demosaicing for Plenoptic Cameras, » *IEEE International Workshop on Multimedia Signal Processing*. IEEE. 2017, DOI : 10.1109/MMSP.2017.8122234. **Prix du meilleur papier.**
- Pierre David, Mikaël Le Pendu, Christine Guillemot. « Sparse to Dense Scene Flow Estimation From Light Fields, » *IEEE International Conference on Image Processing*. IEEE. 2019, p. 3736-3740, DOI : 10.1109/ICIP.2019.8803520. **Papier top 10% .**
- Pierre David, Mikaël Le Pendu, Christine Guillemot. « Scene Flow Estimation from Sparse Light Fields Using a Local 4D Affine Model, » *IEEE Transactions on Computational Imaging*. IEEE. 2020, vol. 6, p. 791-805, DOI : 10.1109/TCI.2020.2981200.
- Pierre David, Mikaël Le Pendu, Christine Guillemot. « Angularly Consistent Light Field Video Interpolation, » *IEEE International Conference on Multimedia and Expo.* IEEE. 2020, DOI : 10.1109/ICME46284.2020.9102968.

OPENING

INTRODUCTION

CONTEXT

Since its invention in 1826 by Nicéphore Niépce, Photography reinvented itself many times in terms of applications but also in terms of process. From bitumen to electronic photodetector, trough silver halide, the recording medium changed a lot during the 20th century. However, standard photography remains trapped in recording a unique point of view of the scene at once. This unique point of view can be restricting as it limits the immersion of the observer who is deprived of several depth cues that are often used to perceive depth in real scenes: parallax, accommodation, stereopsis and convergence.

To cope with this, new imaging modalities have emerged these last years. One of the most promising modality is *light field*, which describes light flowing in a scene as a field of light rays. Alternatively, it can be seen as a collection of photographs of a scene simultaneously recorded from different points of view. By capturing and displaying light fields, an observer is able to perceive the scene as if they were in it. From a computer vision perspective, different points of view typically allow for disparity estimation and thus depth estimation. Last, standard photographs can be generated from light fields where the point of view, the aperture and the focal plane are changed after the acquisition.

Capturing light fields is usually done with two types of devices: (a) with an array of cameras regularly placed on one plane, (b) with a plenoptic camera which is composed of a main lens, a lenslet array and a sensor. At first, the developed light field capturing devices were recording still light fields. However, given that the sensors used in these arrays of cameras and plenoptic cameras are the same as in standard cameras (i. e., CMOS or CCD) and provided that the bandwidths of the devices are large enough, recording light fields of a scene at different time instants is feasible. In this regard, several video datasets have been recently published offering an important diversity of light fields: synthetic (generated with computer graphics softwares), captured with an array of cameras or with a plenoptic camera.

Retrieving a usable light field from those devices can however be challenging as both types require heavy post-capture processing, e. g., demosaicing, devignetting, calibration. So, an important part of literature on light fields is dedicated to this issue. Another major part of research focus on depth estimation and view interpolation, which are key steps for any compression scheme or post-capture rendering. Having efficient methods to perform compression is especially needed for light fields as they are 4D and highly redundant, even more when we consider light field videos.

MOTIVATIONS AND CONTRIBUTIONS

Extracting a light field from a plenoptic raw image with the best possible quality is crucial for a lot of applications. A badly extracted light field will impact performance in depth estimation, view interpolation and post-capture rendering tasks. Therefore, every step of the extraction must be carefully investigated. Particularly, the underlying structure of the raw image given by the lenslet array of a plenoptic camera must be taken into account. This leads us to the first question tackled in this thesis:

1. How to retrieve the views of the light field from the sensor data of a plenoptic camera, taking into account the lenticular structure of the raw image?

To answer this question, we will investigate the flaws of the most used extracting pipeline, step by step. We will see how some steps do not take into account the lenslet structure of the raw image and how it produces light field with ghost artifacts (i. e., peripheral views of the light field are blended with other views). Based on this assessment, we propose some improvements that take into account the lenslet structure by using a plenoptic white image to guide the interpolation steps. We show how our proposed method removes the ghost artifacts.

Once light fields are extracted from plenoptic cameras or arrays of camera, a common use of them is to retrieve geometry information. A lot of methods have already been proposed to estimate depth maps from still light fields but very few research has been conducted on motion analysis from light field videos. In the computer vision litterature, the concepts of *optical flow* and *scene flow* are often used to describe the apparent motion in a scene. The optical flow is a vector field that gives the apparent 2D motion of every pixel on the sensor between two time instants. The scene flow is an extension of the optical flow where some depth information is added, i. e., depth and depth variation estimation. Intuitively, light field videos should be well suited for estimating scene flow: we can treat every view of the light field video as a classical frame and apply standard optical flow methods. For depth estimation, we could use pairs of views as stereo images and apply common disparity estimation methods. However, this would raise two major issues: first, the baseline between two different points of view of the light field might be too small to use regular stereo methods, second the scene flow estimates can be inconsistent among views. This gives us our second point:

2. How to estimate the scene flow of any light field video that is both accurate and consistent between every view (i. e., *angularly consistent*)?

We propose a local 4D affine model to describe the scene flow of a light field video. We split the light field into clusters and estimate the model parameters for each cluster with initial scene flow estimates. The model enforces the consistency of the scene flow in every 4D cluster. We show two applications of this model: a sparse-to-dense interpolation method where the initial scene flow is estimated on a limited subset of rays, and a regularization method where the scene flow estimates are dense but inconsistent in the light field volume. This last application allows us to use state-of-the-art deep

learning methods on each light field view and then to regularize the estimates to remove the outliers and the angular inconsistencies.

A usual application of motion estimation methods is to perform frame interpolation which consists in generating intermediate frames between two given consecutive frames. This is especially helpful for light field video since some plenoptic cameras have a very low framerate. Furthermore, as compression is crucial for light field processing, being able to generate a whole light field video from a few light field frames is a meaningful goal to aim for. From this arises our final question:

3. Using our scene flow estimation, how can we generate consistent intermediate light field frames from two consecutive ones?

In this context, we propose a new method for interpolating light field frames from a light field video that are angularly and temporally consistent. The angular consistency is constrained by the aforementioned scene flow estimation method and the temporal consistency is enforced with a neural network trained on individual views.

THESIS STRUCTURE

The rest of the thesis is organized in the following manner:

Chapter 1 provides background on light field imaging. First, the history of light field is summarized. Then the light field function is formally defined and we show how it is derived from the plenoptic function. We exhibit different ways of visualizing light fields and we give usual applications of light fields. Next, we detail the different ways of capturing light fields: arrays of cameras and plenoptic camera. Finally, we review different methods that have been proposed to extract a light field from arrays of cameras and plenoptic cameras.

Chapter 2 proposes improvements on one of the most used extracting pipeline for plenoptic cameras. First, we analyze and describe the flaws of the state-of-the-art pipeline. To better identify the different sources of artifacts, our analysis is performed by generating ideal lenslet images from synthetic light fields and use them as input of the decoding pipeline. Then, we detail a new method of demosaicing based on the provided white lenslet images serving as guide. Furthermore, we show that this kind of guided interpolation can be useful on other steps of the decoding pipeline. Finally, the quality of the resulting views is assessed for both synthetic and real light fields using visual comparisons as well as objective metrics.

Chapter 3 gives an overview on several computer vision issues related to scene flow estimation. Namely, we first exhibit the different types of methods that have been proposed to estimate depth from light fields. Then, we review some literature about optical flow estimation. Next, publications about scene flow estimation are discussed. Finally, we detail numerous state-of-the-art methods for frame interpolation.

Chapter 4 proposes a new method of scene flow estimation from light field videos. We first introduce a local 4D affine model to represent scene flows, taking into account light field epipolar geometry. The model parameters are estimated per cluster in the 4D ray space. They are derived by fitting the model on initial motion and disparity estimates obtained by 2D sparse or dense optical flow estimation techniques. The model is first shown to enable dense scene flow estimation from sparse correspondences between pairs of views. We then demonstrate that the model is also very effective for estimating scene flows from 2D optical flows. The model regularizes the optical flows and disparity maps, and interpolates disparity variation values in occluded regions. The proposed model allows us to benefit from deep learning-based 2D optical flow estimation methods while ensuring scene flow geometry consistency in the 4 dimensions of the light field.

Chapter 5 addresses the problem of temporal interpolation of light field videos using dense scene flows. Given light fields at two time instants, the goal is to interpolate an intermediate light field to form a spatially, angularly and temporally coherent light field video sequence. We first compute angularly coherent bidirectional scene flows between the two input light fields. We then use the optical flows and the two light fields as inputs to a convolutional neural network that synthesizes independently the views of the light field at an intermediate time. In order to measure the angular consistency of a light field, we propose a new metric based on epipolar geometry. Experimental results show that the proposed method produces light fields that are angularly coherent while keeping similar temporal and spatial consistency as state-of-the-art video frame interpolation methods.

LIST OF PUBLICATIONS

Most of the described findings and proposals are published in the following proceedings of international conferences and journals:

- Pierre David, Mikaël Le Pendu, Christine Guillemot. "White Lenslet Image Guided Demosaicing for Plenoptic Cameras," In: *IEEE International Workshop on Multimedia Signal Processing*. IEEE. 2017, doi: 10.1109/MMSP.2017.8122234. **Best Paper Award.**
- Pierre David, Mikaël Le Pendu, Christine Guillemot. "Sparse to Dense Scene Flow Estimation From Light Fields," In: *IEEE International Conference on Image Processing*. IEEE. 2019, pp. 3736-3740, doi: 10.1109/ICIP.2019.8803520. **Top 10% Paper.**
- Pierre David, Mikaël Le Pendu, Christine Guillemot. "Scene Flow Estimation from Sparse Light Fields Using a Local 4D Affine Model," In: *IEEE Transactions on Computational Imaging*. IEEE. 2020, vol. 6, pp. 791-805, doi: 10.1109/TCI.2020.2981200.
- Pierre David, Mikaël Le Pendu, Christine Guillemot. "Angularly Consistent Light Field Video Interpolation," In: *IEEE International Conference on Multimedia and Expo*. IEEE. 2020, doi: 10.1109/ICME46284.2020.9102968.

Part I

LIGHT FIELD IMAGING

1

BACKGROUND ON LIGHT FIELD IMAGING

1.1 HISTORY

Gabriel Lippmann, inventor of color photography and 1908 Nobel Prize winner is the first scientist who proposed a system design which could capture a light field. Back then, the expression *light field* did not exist, Lippman used the expression "photographie intégrale", in English integral photography. In March 1908, Lippman showed his theory in front of the French Academy of Science in a report named "Épreuve réversible, Photographie Intégrale" [1]. However at that time, he did not have the technical means to manufacture his invention.

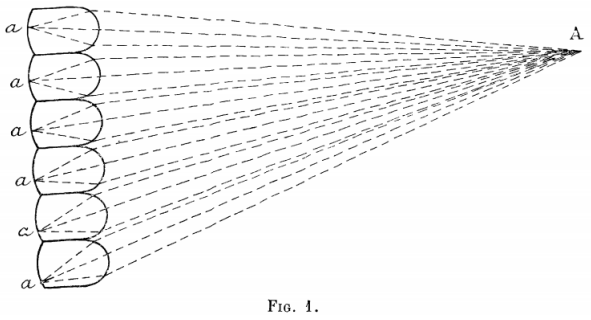


Figure 1.1: Sketch from Lippmann's presentation

In 1930, Herbert E. Ives invented a method to make parallax panoramagrams, using a large diameter lens [2]. He filed two patents related to this invention [3, 4]. Panoramagrams are auto-stereoscopic photographs which use binocular disparity and parallax to give the illusion of depth.

In 1932, Gerd Heymer filed a patent [5] which described how to print a lens array onto a photosensitive film. The original invention of Lippmann could then be realized. From 1935, Winnek Coffey invented a device close to the already existing ones (main lens and lenticular film) but unlike the others, it showed the need to match the aperture of the main lens with the aperture of the lenslet array [6]. Progressively, the devices were perfected and in 1952, Gruetzner filed the patent of the first light field camera with commercial purpose [7].

From a theoritical point of view, Gershun is the first scientist who used the term *light field* in 1936. He formalized this notion in a vectorial function in a monograph named "The light field" [8].

Gradually, from the 50s, the scientific interest for light field began to fall. Some scientist like Chutjian (in the 60s) and Dudnikov (in the 70s) however carried on some research. Rebirth of light field occurred with the development of computer graphics and digital camera. Adelson published two papers in 1991 and 1992 respectively focused on the plenoptic function [9] and the plenoptic camera [10].

In 1996, Marc Levoy and Pat Hanrahan, researchers at Stanford University, published a seminal paper named "Light Field Rendering" [11], in which they described a new representation of light fields, which they called light slab. The article also showed how to extract new views from a limited number of recorded views. They proved it could be useful for many applications requiring interaction with 3D scenes and that the huge volume of data it produces could be compressed. This article truly boosted the trend around light fields in the scientific community as it showed a practical use of light fields.

At the same time, there were also innovations in capturing light fields from real scenes. In 2004, Stanford Computer Graphics Laboratory (the laboratory where Levoy and Hanrahan worked) built a multi-camera array. One of its application was to capture light fields [12]. A few years later, in 2005, Ren Ng, PhD student of Marc Levoy and Pat Hanrahan, published two important articles [13, 14]. In the first one, he developed a device to capture light fields without using a multi-camera array and in the other one, he demonstrated how to refocus a light field image, using the Fourier domain. In 2006, Ren Ng created a company called Lytro, which developed and sold plenoptic cameras. In 2009, Christian Perwaß founded Raytrix, which develops another type of hand-held light field cameras for industry and research. This type of light field cameras is based on what Todor Georgiev called *focused plenoptic camera* in his eponymous article [15], he sooner had demonstrated that it was beneficial to trade angular resolution for spatial resolution [16].

1.2 LIGHT FIELD DEFINITION

1.2.1 *Plenoptic function*

One of the simplest way to describe light and its interaction with matter is to model its propagation in terms of rays. In this model, a light source emits a discrete amount of rays in different directions. Each ray carries a part of the emitted energy and travels in straight lines in homogeneous medium. The rays can also be bend, reflected or absorbed when they encounter a new medium. This description of light has the advantage of being simple to compute and accurate when it comes to describe light interactions with objects that are much larger than the light ray wavelength ($\approx 10^{-3}$ mm).

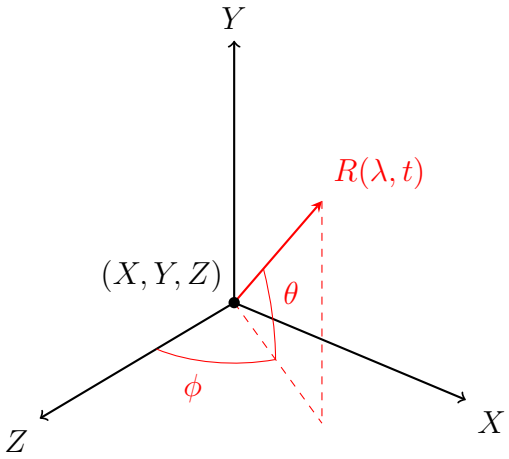


Figure 1.2: Geometrical representation of the plenoptic function

In this context, when trying to characterize human and computer vision, Adelson and Bergen [9] proposed the plenoptic function P:

$$R = P(X, Y, Z, \theta, \phi, \lambda, t) \tag{1.1}$$

This function describes the radiance R perceived by an observer located at point (X, Y, Z) at a given time t, looking in the direction given by the spherical angular coordinates (θ, ϕ) for a wavelength λ. It is a 7 dimensional function with 5 geometrical dimensions (X, Y, Z, θ, ϕ) whose parametrization is shown in Figure 1.2.

The plenoptic function can be used to describe any observable scene with any light capturing device. For example, a pinhole camera records a sampling of the plenoptic function at a fixed position (X, Y, Z) at time t. The focal length of the lens and the size of the sensor determine the range of (θ, ϕ) and the color filter array placed over the sensor samples the visible wavelengths λ into three colors: *red*, *green*, *blue*. In practice, the plenoptic function is often simplified: instead of using the radiance of the light rays for every wavelength, [17] directly uses the signal intensity of the sensor pixels I for three color channels (*red*, *green*, *blue*). Furthermore, the temporal dimension is also often omitted.

1.2.2 *Light field function*

The plenoptic function is a complicated function that we cannot measure for now. However, for some scenes with non concave objects and without bulk attenuation, it is possible to reduce the function to four degrees of freedom. As the radiance remains the same along a light ray, a plenoptic function is very redundant, the five geometrical dimensions of the plenoptic function can be reduced to four. In 1996, [11] and [18] simultaneously proposed a new 4-dimensional parametrization of the reduced plenoptic function, that they respectively called *light field* and *lumigraph*. In the rest of this thesis,

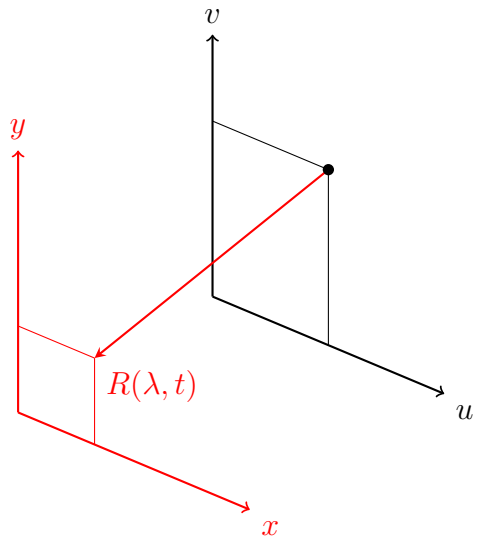


Figure 1.3: Geometrical representation of the light field function

we will refer to this reduced plenoptic function as *light field* as it is currently the most commonly used in the literature. The parametrization described in [11] and [18] uses two parallel planes to define the light field as we can see in Figure 1.3. Note that this parametrization implies that the light field function is only defined in a half-space.

$$I = L(u, v, x, y) \tag{1.2}$$

This parametrization has the advantage of eliminating angular coordinates and getting closer to the parameterization of a conventional 2D image. Thus, if we take the same example of a pinhole camera shooting a scene, the camera samples the light field function at a given position (u, v) for a range of (x, y) determined by the optical configuration of the camera. The (x, y) coordinates define a 2D image and (u, v) coordinates a point of view. By convention, we respectively call the (u, v) and (x, y) coordinates *angular* and *spatial* dimensions.

Other parametrizations were proposed by [19]. They are shown in Figures 1.4 and 1.5. In Figure 1.4, a ray is defined by two distinct points on a sphere. In the parametrization on Figure 1.5, a ray is defined by a point on any surface and an orientation.

Similarly to the plenoptic function, we can also define a *light field video* function where a temporal dimension is added to the *light field* function. By definition, each frame of a light field video is a light field.

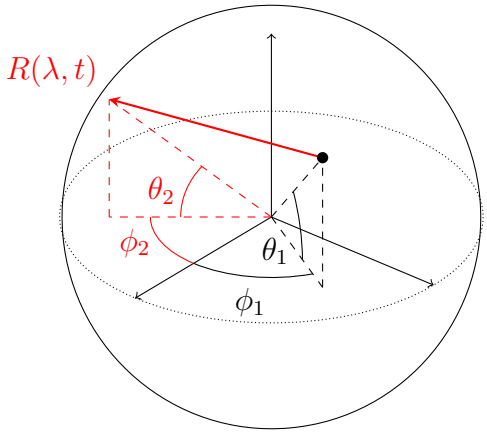


Figure 1.4: Alternative representation of the light field function

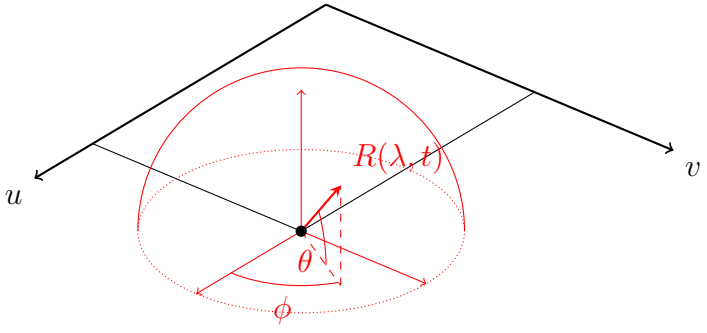


Figure 1.5: Alternative representation of the light field function

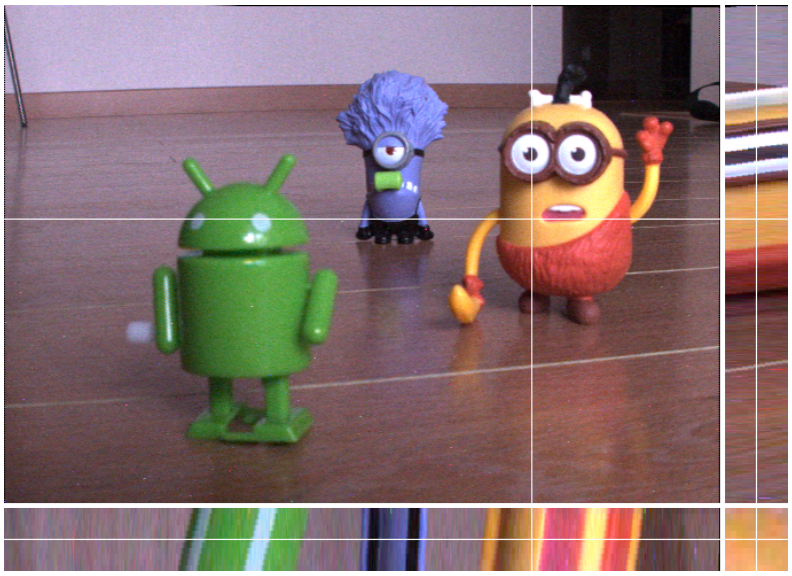


Figure 1.6: View and epipolar plane images of a light field taken from [20]

1.2.3 *Views and epipolar plane images*

Because a light field is a 4D function, it is not possible to visualize it. Instead, we often use partial function representations to display a part of the light field. We call *view* or *subaperture image* a sample of the light field at a fixed angular position (u_0, v_0):

$$V_{u_0 v_0}(x, y) = L(u_0, v_0, x, y) \tag{1.3}$$

Likewise we respectively denote *horizontal* and *vertical epipolar plane images* the following partial functions:

$$E^h_{v_0 y_0}(u, x) = L(u, v_0, x, y_0) \tag{1.4}$$
$$E^v_{u_0 x_0}(v, y) = L(u_0, v, x_0, y) \tag{1.5}$$

These representations allow us to visualize the epipolar lines in the light field and thus the geometry of the scene. The slopes of the lines vary with the depth of the 3D point they come from. The different presentations are shown in Figure 1.6.

1.2.4 *Possibilities offered by light fields*

Virtual and augmented reality

Once a light field is captured or synthesized, the first application is to display it. Research on light field displays has been going on since 2000 when [21] proposed new methods for dynamic rendering and autostereoscopic viewing of light fields. Then, an autostereoscopic viewing system was developped in [22], using a lenslet array in front

of a projector. Another possibility to view a light field is to use multilayer displays, also know as tensor displays, as it is the case in [23, 24].

Compared to standard 2D displays, light field displays provide more depth cues: parallax, stereopsis, accommodation and potentially convergence. By projecting the individual light rays of the light field with their original radiance, the observer theoretically has the same viewing experience as if the object displayed was really in the same room. This enhances the immersion of the observer and therefore opens a lot of opportunities for virtual [24] and augmented [25] reality applications.

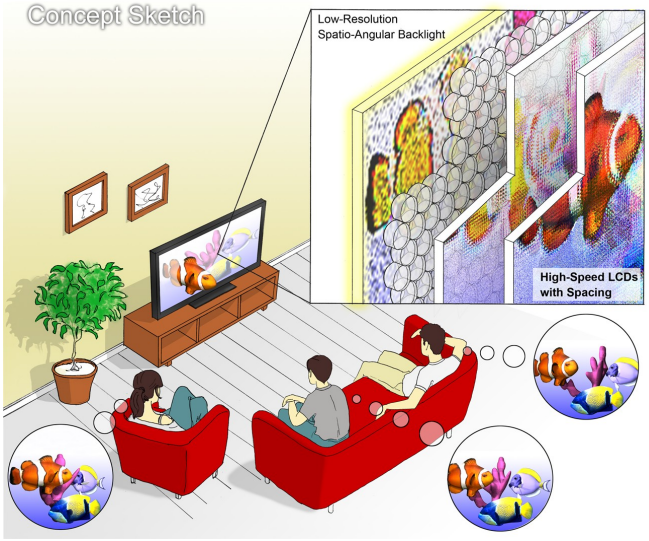


Figure 1.7: Sketch of a light field display taken from [23]

Post-capture image rendering

By having captured a light field, a conventional image I can be produced from it, where we can change the point of view (u_0, v_0), the aperture and the focus. A very basic way of doing so is via the shift-and-add procedure:

$$I^{\lambda,\kappa}_{u_0,v_0}(x,y) = \frac{1}{(2\kappa+1)^2} \sum\sum_{(\Delta u,\Delta v)\in\mathcal{W}_\kappa} L(u_0+\Delta u, v_0+\Delta v, x+\lambda\Delta u, y+\lambda\Delta v) \quad (1.6)$$

where $\mathcal{W}_\kappa = [-\kappa,\kappa]^2$. The parameters λ and κ respectively control the depth of the virtual focal plane and the size of the aperture. Note that more sophisticated methods based on splatting kernels [26] or Fourier transforms [14] exist. They either produce higher quality images or have a shorter running time. This is just a very basic example of image rendering methods offered by light fields.

Figure 1.8: Example of post-capture image rendering: Left image is focused on a near plane with a wide aperture, right one has a slightly different point of view with a focus on a far plane and a narrow aperture

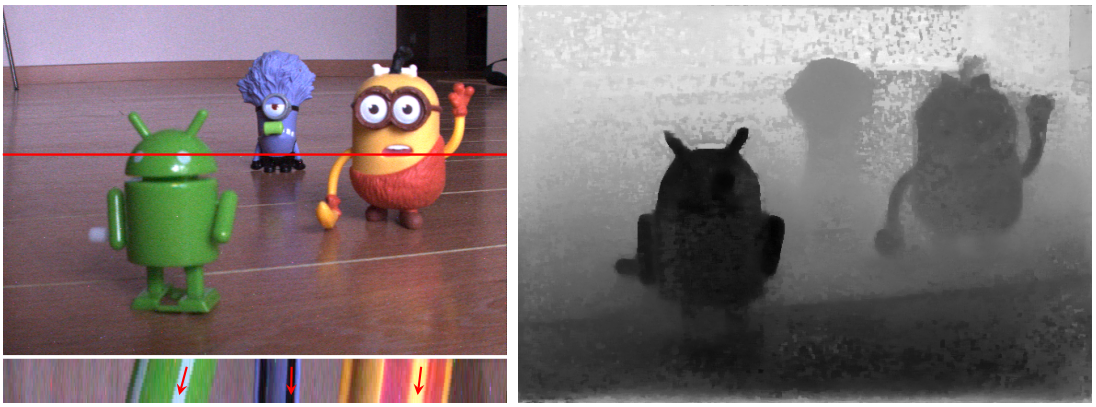


Figure 1.9: Illustration of the depth estimation principle using slopes in EPI

Depth estimation

As mentioned before, a light field describes the radiance of a collection of light rays sampled in position and direction. If we consider that the light rays emitted in every direction by a point of an object have the same radiance (we then call the object *lambertian*), then it is possible to use this redundancy to estimate the distance between the light field capturing device and the object. By applying this principle on the whole scene, we can estimate a depth map. Classically this redundancy has been exploited in three different ways: (1) by finding correspondences between the different views of a light field, similarly to stereo matching, (2) by estimating the slopes in the EPI, (3) by generating a collection of images with different focus (or *focal stack*) and analyzing the defocus cues. In practice, these methods do not directly give the actual depth of the scene but a representation of the depth, e. g., disparity, slope coefficient, defocus values. Other methods have been proposed, including learning-based approaches and a more thorough litterature review will be made in Chapter 3.

1.3 CAPTURING LIGHT FIELDS

1.3.1 *Camera arrays*

The most straightforward way of capturing light fields is to use multiple standard cameras to capture different points of view. As mentioned before, a standard camera captures spatial information at fixed angular position. Capturing a light field with standard cameras can be done in two different ways: either by moving a camera on a gantry like [11, 27] — this technique can only be used to capture a light field of a still scene — or by placing multiple cameras on a regular grid like [28–31]. The latter requires the cameras to be synchronized when shooting dynamic scenes. In both cases, a calibration is needed to compute the relative position of each point of view.

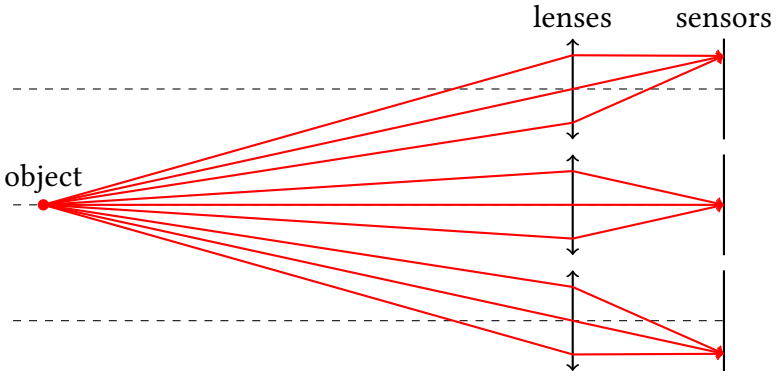


Figure 1.10: Optical configuration of an array of cameras

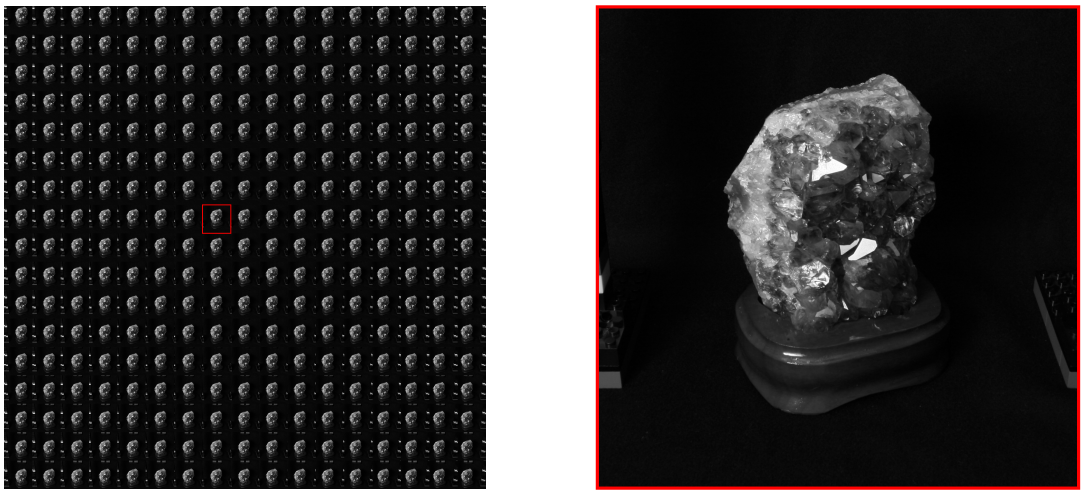


Figure 1.11: Raw images taken by an array of cameras

1.3.2 *Plenoptic cameras*

The recent development of plenoptic cameras enables the instantaneous capture of light fields with both spatial and angular information of the scene, as opposed to traditional cameras which can only capture spatial information from a singular point of view. It is more compact than the array of cameras and does not require an alignment and calibration step. The optical design of a plenoptic camera is actually similar to the optical design of a traditional camera, with a main lens and a sensor (CCD or CMOS). A lenslet array is then precisely introduced between the last lens and the sensor. Depending on the distance between the lenslet array and the sensor, the imaging of light field is completely different and defines two types of plenoptic cameras, called *1.0* and *2.0*.

Plenoptic 1.0

The first model, originally described in [1] and modernized by [13], was popularized by its implementation in the *Lytro* cameras. In this design, the main lens focuses the objects on the lenslet array which separates the converging rays on the sensor (see Figure 1.12). In this configuration, the spatial resolution of the captured light field is given by the number of lenslets and the angular resolution is given by the number of pixels behind every lenslet. For example, the *Lytro 1* camera has approximately 370×370 lenslets and 9×9 pixels behind each lenslet.

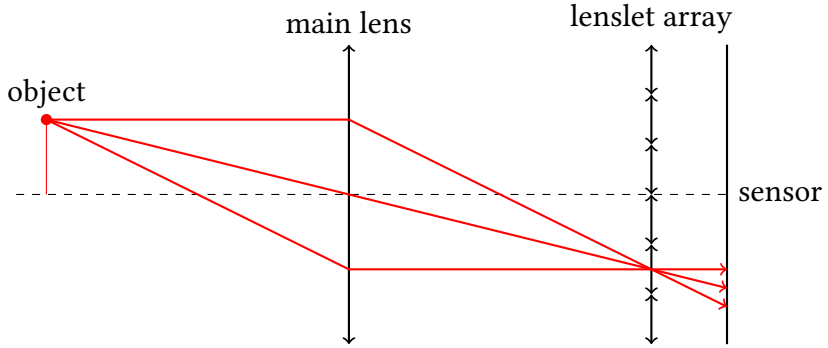


Figure 1.12: Optical configuration of a plenoptic camera 1.0

Plenoptic 2.0

The *plenoptic 2.0* design, also called *focused plenoptic*, was proposed in [15]. In this configuration, the image plane of the main lens is the object plane of the lenslet array (see Figure 1.14). So, the lenslet images on the sensor are in focus. For example, in Figure 1.15, the wheel trim of the car is in focus. This design is used in *Raytrix* cameras and enables the users to control the trade-off between spatial and angular resolution. It is similar to the camera array configuration where each lenslet captures a different point of view, the main difference is that one lenslet only captures a fraction of the scene.

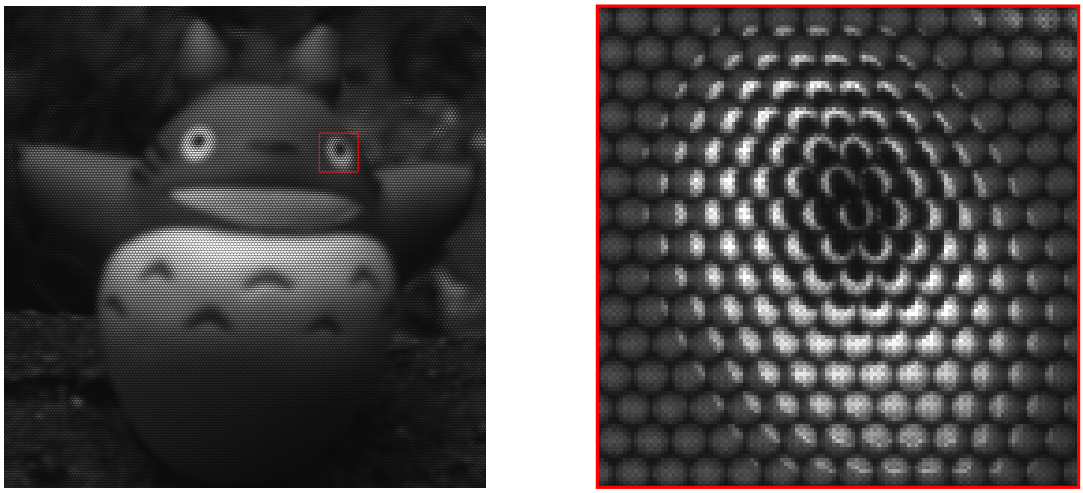


Figure 1.13: Raw image taken by a plenoptic camera 1.0

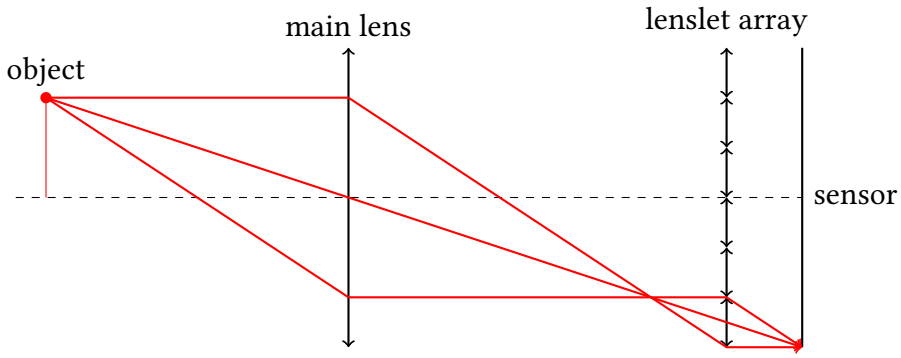


Figure 1.14: Optical configuration of a plenoptic camera 2.0

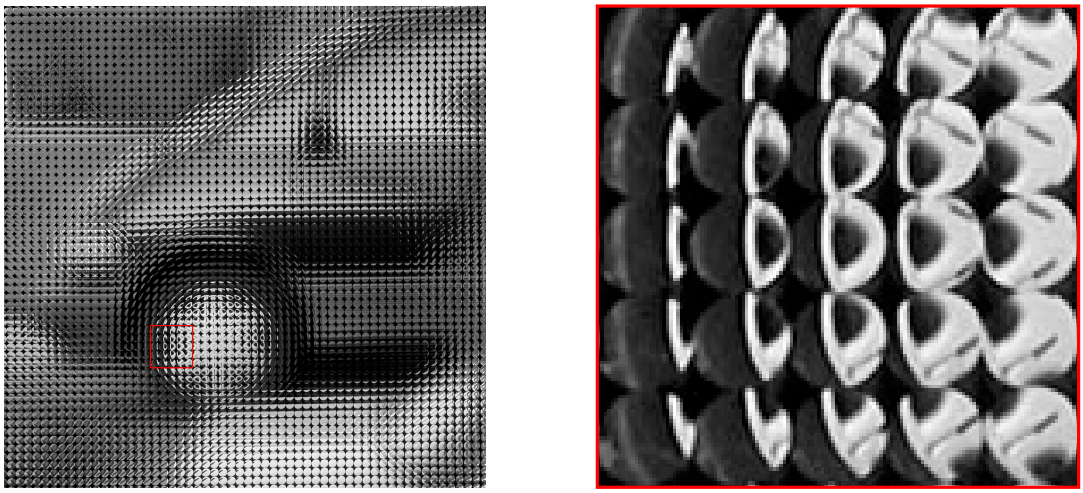


Figure 1.15: Raw image taken by a plenoptic camera 2.0

1.4 DECODING LIGHT FIELDS

1.4.1 *Calibrating camera arrays*

When assembling a camera array to capture light field, we want the camera to be coplanar and regularly spaced from each other. However there are always small mechanical misalignment between the cameras. This produces light field with an irregular sampling of the angular coordinates which are complex to handle properly, even more so when the exact sampling is unknown. Furthermore, the different cameras always have different sensor responses which lead to inconsistent colors among every view. For these reasons, raw views captured with camera arrays are often rectified to produce light fields with a regular angular sampling and homogeneous colors. In order to interpolate the new rectified view, a geometrical and calibration step is needed to know the relative positions of the cameras. This can be done independently on every camera so, before explaining the different methods for calibrating a whole camera array, we will explain how one performs the geometrical calibration of one camera.

Background on classical geometrical camera calibration

In computer vision, a geometrical camera calibration describes a method to estimate the parameters of a lens and image sensor model. These parameters can then be used to correct for lens distortion, measure the actual size of an object, or determine the location of the camera in the scene.

PINHOLE MODEL One of the most used models in camera calibration is the pinhole model. The camera is approximated by a perfect pinhole, which has five intrinsic parameters. We call intrinsic the camera parameters which are independent of the place of the camera in the world.

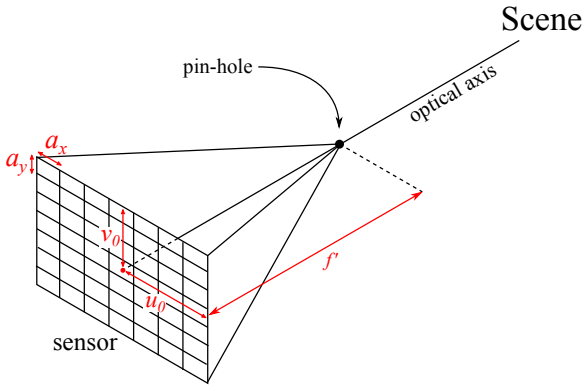


Figure 1.16: Pinhole model

These five intrinsic parameters can be put in a 3×3 matrix K:

$$K = \begin{pmatrix} \alpha_x & \gamma & u_0 \\ 0 & \alpha_y & v_0 \\ 0 & 0 & 1 \end{pmatrix} \tag{1.7}$$

Where $\alpha_x = f'/a_x$ and $\alpha_y = f'/a_y$, with f' being the effective focal length of the lens, a_x and a_y being the size of a pixel along the x- and y-axis. The parameters u_0 and v_0 are the coordinates of the optical center, and γ is the skew coefficient between the x- and y-axis (often equal to 0).

In a real lens, the pinhole plane of the model corresponds to the pupil plane and the rays passing through the center of the pupil/pinhole are the chief rays. So, when computing the intrinsic parameters α_x, α_y, we actually compute the distance between the sensor and the pupil in terms of pixels.

DISTORTION COEFFICIENTS The pin-hole model does not take the distortion induced by the lenses into account. For narrow angle lenses, it is not much of a problem since the distortion is normally very small but for wide angle lenses, distortion cannot be ignored. Such distortions are well modeled using polynomial regression. The order of the regression depends on the precision we want to have. There are two types of coefficients: radial ones and tangential ones. For example, the third order distortion model is parameterized with five coefficients three radial distortion coefficients k_1, k_2, k_3 and two tangential distortion coefficients p_1, p_2, such as:

$$\begin{cases} x_d = x_u(1 + k_1 r^2 + k_2 r^4 + k_3 r^6) + p_2(r^2 + 2{x_u}^2) + 2p_1 x_u y_u \\ y_d = y_u(1 + k_1 r^2 + k_2 r^4 + k_3 r^6) + p_1(r^2 + 2{y_u}^2) + 2p_2 x_u y_u \end{cases} \tag{1.8}$$

Where (x_d, y_d) is the distorted image point as projected on image plane using specified lens, (x_u, y_u) is the undistorted image point as projected by an ideal pinhole camera and $r^2 = (x_u - u_0)^2 + (y_u - v_0)^2$

POSITION OF THE CAMERA After having computed the intrinsic parameters and possibly the distortion coefficients, the third part of camera calibration is called extrinsic calibration: it computes the 3×3 rotation matrix R and the 3D translation vector T, which is the position of the origin of the world coordinate system (X, Y, Z) expressed in coordinates of the camera-centered coordinate system of the camera. Summarizing the whole development, we have the following relation:

$$w \begin{pmatrix} x_u \\ y_u \\ 1 \end{pmatrix} = K \begin{bmatrix} R & T \end{bmatrix} \begin{pmatrix} X \\ Y \\ Z \\ 1 \end{pmatrix} \tag{1.9}$$

Camera array calibration

Most modern sensors use a color filter array to capture color images. This array is composed of red, green and blue filters and so, each pixel on the sensor lacks two color channels which can be retrieved by a process called demosaicing. In [32], the views captured from the different cameras are first demosaiced. Then a white-balance is applied using a medium gray area visible from every camera. This step is performed to have views with homogeneous colors. Next, based on the aforementioned distorted pinhole model, a standard intrinsic/extrinsic calibration is performed on every camera in order to apply a mapping homography on the views.

The authors in [12] and [33] have a more complex and global approach on calibration. First, for color calibration, inspired by [30], they calculate the slopes and offsets of each camera sensor responses for each color channel by taking images of a white target under several different exposure levels. Then, after the color homogenization, they demosaic each raw image. For the geometric calibration, while having different approaches, both assume that the cameras are coplanar. In [12], they compute the relative position of every camera on the array by measuring the parallax on several points. On the other hand, the authors in [33] perform an intrinsic/extrinsic calibration of every camera using [34] as bundle adjustment methods yield good results according to [35]. Then, the authors interpolate pseudo-rectified views.

1.4.2 *Demultiplexing plenoptic cameras*

In every design of plenoptic cameras, the lenslet image captured by a plenoptic camera is not directly interpretable and needs to be decoded. Because information is captured and arranged differently on plenoptic 1.0 and 2.0 designs, their respective decoding pipelines are different. However, every plenoptic camera shares common flaws that need to be handled during the decoding. First, the raw lenslet image has to be demosaiced. Then, images captured with a plenoptic camera have the same pattern as the lenslet array: each lenslet creates a small image on the sensor. These images suffer from heavy vignetting, which means that the brightness at the periphery of the lenslets is reduced compared to the lenslet centers. Finally, there is always a slight misalignment between the sensor and the lenslet and this misalignment changes for every manufactured plenoptic cameras. This misalignment should be taken into account for the demultiplexing and therefore a calibration step is needed to determine the positions of the lenslet centers on the sensor.

Plenoptic 1.0

Several demultiplexing pipelines have been proposed for plenoptic 1.0 designs. The pipeline proposed in [36] will be detailed in the next chapter, as we will propose improvements.

The authors in [37] first apply a gamma correction on their raw image. Then, using a white raw image, they perform a white-balance and then demosaic the captured

image. The next step is the lenslet array calibration: they compute the rotation of the lenslet array in the frequency domain using the Fourier Theorem, then they compute the centers of the lenslet array on the spatial domain using a triangular mesh (or Delaunay triangulation). Next, for every position around the lenslet centers, they extract the corresponding view by using a triangular mesh and performing a barycentric interpolation. Finally, they super-resolve their views using dictionary-learning with sparse coding.

In [38], the parameters of the grid model of the lenslet array are first optimized by solving a global optimization problem using a coarse-to-fine brute force search. Then, 4D demosaicing is performed: to demosaic each pixel of a lenslet image, the authors use the neighboring pixel color information as well as the pixels of the neighbor lenslet images. For the remaining steps, i. e., slicing the views from the raw image and color correction, the authors use the same methods as in [36].

Plenoptic 2.0

Plenoptic 2.0 cameras have different optical designs than plenoptic 1.0. Therefore the way of extracting information from their raw sensor is also different. As [39] showed, although it is possible to extract views and EPI from raw plenoptic 2.0 images, it is not the most natural way since it produces a lot of artifacts and we first need to estimate a depth map. Instead, a focal stack is often extracted, e. g., in [15]. A focal stack is a collection of pictures of the same scene but with different focal planes. With a plenoptic 2.0 camera, a focal stack can be generated by projecting each pixel (x, y) of the sensor associated with the lenslet (i, j) into a 2D image according to the following equation:

$$\begin{pmatrix} X \\ Y \end{pmatrix} = s \left(g \begin{pmatrix} x \\ y \end{pmatrix} - C_{ij} \right) + C_{ij} \tag{1.10}$$

Where (X, Y) are the coordinates of the projected pixel on the 2D refocused image. Different (x, y) coordinates can be projected on the same (X, Y) coordinates. In this case, the pixel value at location (X, Y) is often averaged using splatting kernels. C_{ij} denotes the coordinate of the center of the lenslet (i, j), $s < 1$ controls the size of the 2D refocused image, and g controls the plane which is in focus. The output image is s^2 times the sensor image size. In this formulation the size of the re-focus image is independent to the parameter g, and the small images are zoomed by sg. In order to generate a focal stack, a calibration is needed to estimate the positions of the lenslet centers C_{ij} on the raw image as well as a demosaicing step.

For the calibration step, multiple methods have been proposed. They are not specific to a certain design of plenoptic cameras as they rely on the lenslet structure of the raw image which is common to both designs. While proposing a whole decoding pipeline for plenoptic 1.0 cameras as mentioned in the last section, a significant contribution of [36] is the calibration step in which the positions of the lenslet centers are individually estimated on a white lenslet image before being used to fit grid parameters. This calibration method is strictly done in the spatial domain but several methods proposed

to compute the grid parameters in the frequency domain. The authors in [26, 37] use both spatial and frequency domains to estimate the grid parameters while the authors in [40] only use the frequency domain.

2

FROM RAW DATA TO LIGHT FIELDS

2.1 INTRODUCTION

In this chapter, we propose to improve the decoding pipeline for plenoptic 1.0 cameras described in [36], by designing a new demosaicing and a new alignment method. The captured raw images have a particular lenslet structure which must be taken into account to retrieve the views which compose the light field. We choose this pipeline [36], referred to as Dansereau pipeline in the rest of the chapter, because it is widely used in the research community. For instance, it was chosen as part of the common test conditions for the JPEG-Pleno compression standard as well as the ICME 2016 Grand Challenge on Light Field Image Compression. Lastly, the code in [36] is available. Our contributions are as follows: first we analyze the flaws of the pipeline and identify the link between the lenslet structure of the image and demosaicing artifacts, then we propose a demosaicing method guided by a white lenslet image, finally we propose an application of the same principle for the alignment step in the decoding pipeline. The proposed methods are assessed within Dansereau pipeline but would still be theoretically valid for any pipeline that performs these operations on the raw lenslet image.

As evidenced by the theoretical analysis in [41], classical demosaicing methods are not suitable for images captured with plenoptic cameras. Three approaches have been studied in the demosaicing litterature for "plenoptic 1.0 cameras". First, we can directly demosaic the raw sensor image (also called lenslet image). The authors in [38] use a 4D kernel regression method exploiting the lenslet structure. However, it performs independent interpolation of the color channels, which results in a loss of detail compared to a classical 2D demosaicing such as [42]. In [43], the pixels are projected in a depth-layered object space using a depth map computed on the raw image. The missing color values are then interpolated using nearest neighbors inside a layer. The second approach is demosaicing the views after having demultiplexed them. The authors in [44] chose a dictionary learning based method to retrieve the missing colors, while [45] used a disparity map to find the missing colors in other decoded views. This second type of demosaicing presents a major drawback: every interpolation before extracting the views must be done in a nearest neighbor way. The resulting views have therefore some strong aliasing. Finally, the third approach consists in demosaicing the focal stack as in [46]. While computing a focal stack from a set of views is straigtforward, retrieving high quality views from a focal stack is a more challenging problem [47]. For these

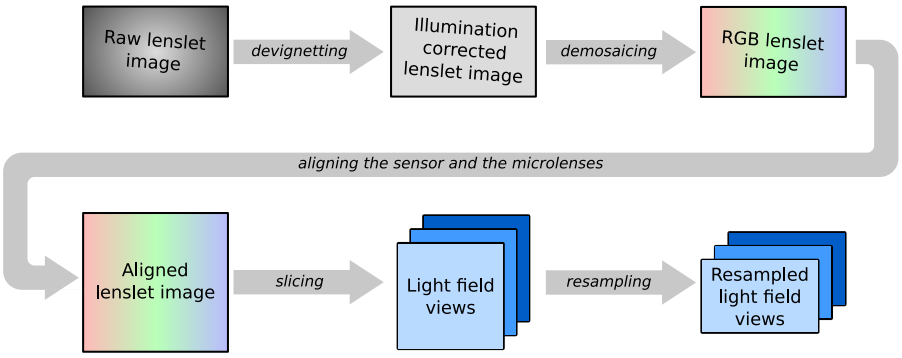


Figure 2.1: Pipeline for extracting views from a raw lenslet image [36]

reasons, we believe that the first approach, taking into account the lenslet structure of the images, could lead to a better decoding of the light field.

2.2 DESCRIPTION OF THE DECODING PIPELINE

This section explains how Dansereau pipeline retrieves the light field views from the raw image of a plenoptic camera and the associated metadata. This algorithm is called "demultiplexing" or "decoding". It proceeds by the following steps (illustrated in Figure 2.1):

DEVIGNETTING As mentionned in section 1.4, lenslet images suffer from heavy vignetting. The devignetting step consists in dividing the sensor data by a calibration image (or white lenslet image) that is chosen, depending on the zoom and focus settings, among a set of RAW pictures of a uniformly illuminated lambertian white surface. These calibration images exhibit the vignetting pattern of the lenslet array. Note that they are also used in the calibration phase to determine the positions of the lenslet centers on the sensor.

DEMOSAICING After correcting the vignetting of the raw image, the pipeline proceeds by demosaicing the lenslet images. It uses the gradient corrected interpolation method proposed in [42]. This method computes the values of the missing RGB channels by bilinearly interpolating the neighbor values and correcting the output with the computed gradient of the present channel. The image is convolved with various kernels depending on the color filters of the pixels. The kernels are given in Figure 2.2. The coefficients α, β and γ represent the strength of the gradient correction. They are chosen to be respectively equal to $1/2$, $5/8$ and $3/4$. These values were selected empirically by the authors to minimize the mean square error on their test dataset.

ALIGNING THE SENSOR WITH THE LENSLETS Knowing the positions of the lenslet centers on the sensor (determined in the calibration phase), rotation, translation and

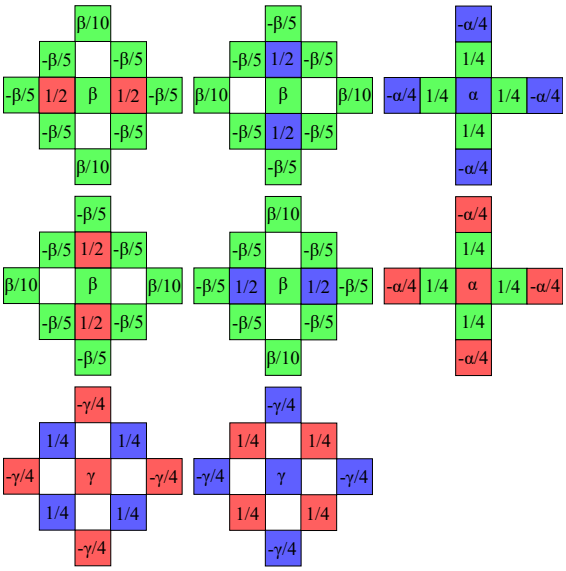


Figure 2.2: Kernels in [42]

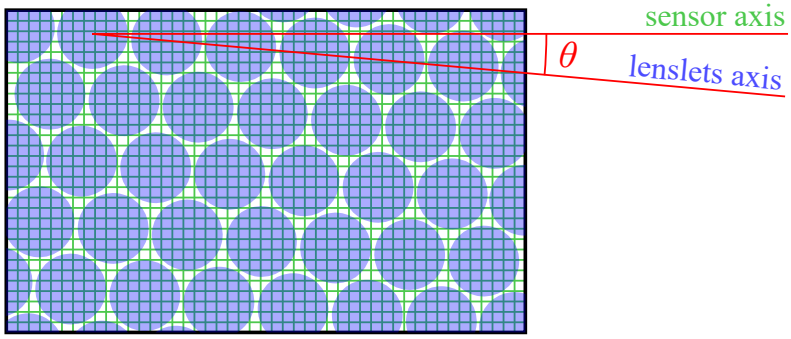


Figure 2.3: Misalignment between the sensor and the lenslets

scaling are applied to the image to compensate for the misalignements between the lenslet array and the pixel grid (illustrated in Figure 2.3).

SLICING THE LENSLET IMAGE The next step is to slice the lenslet image, i. e., demultiplex the lenslet image to extract the views. Now that the diameter D of each lenslet image is scaled to be an integer, we just have to pick a pixel every D pixels to form a view (see Figure 2.4).

RESAMPLING THE VIEWS As the lenslet array is hexagonal, the views suffer from an hexagonal aliasing (see the two missing pixels in black in the extracted view in Fig. 2.4: every line over two is shifted by half a pixel). The pipeline removes this aliasing by resampling the views into a rectangular grid.

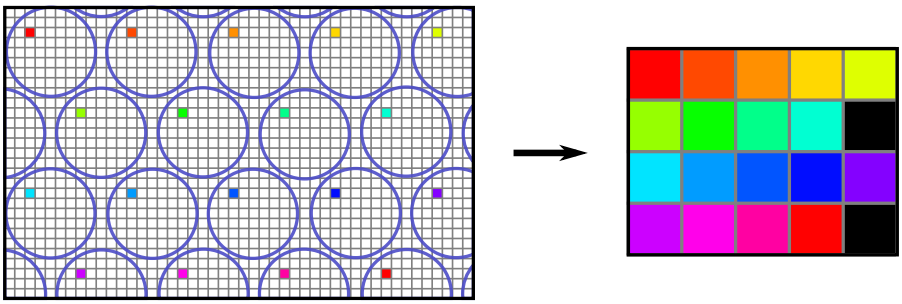


Figure 2.4: Extraction of a given view from a lenslet image: note how the hexagonal sampling of the raw lenslet image on the left results in missing pixels (in black) in the view on the right.

2.3 FLAWS AND IMPROVEMENTS OF THE PIPELINE

2.3.1 *Synthetic lenslet image generation*

In order to better analyze the pipeline, we developed a method to make a synthetic lenslet image from a synthetic light field[27] (see Fig. 2.5). We first remove some peripheral views (in the corners), in order to have circular angular patches (as the captured plenoptic light field are). Then we put the different angular patches near one another along a hexagonal grid. This operation is the exact inverse of the slicing or demultiplexing step in the decoding pipeline. Note that the diameter of the generated lenslet image is only 9 (as we originally have 9x9 synthetic views) and that for a real plenoptic camera, the diameter is likely to be higher (11 for the Lytro 1, 15 for the Lytro Illum for example). Then we remove two channels from each pixel according to a Bayer pattern to generate ground truth demosaiced lenslet images.

Figure 2.6 shows the ground truth lenslet image and the right picture in Figure 2.7 shows the same image demosaiced with Malvar's method [42]. We can see that this method is not ideal as it creates color fringes on the borders of the lenslets. The fringes are due to the pixels that are out of the lenslet. As they get no signal, their value remains the same (if we ignore noise) between the raw image and a white image. So when we divide the raw image by the white image, these pixel values are 1. When interpolating the missing channels for a pixel at the border of a lenslet, the bilinear interpolation and the gradient corrections are both disturbed by these white pixels.

2.3.2 *Proposed demosaicing method*

Following this observation, we propose to discard the pixels that are out of the lenslets from the demosaicing step. To do this, we adapted the gradient corrected interpolation method by weighting the bilinear interpolation and gradient correction:

- First with a mask b : we do not want to interpolate data from different lenslets as this creates crosstalk artifacts. Knowing the lenslet grid parameters, we know the

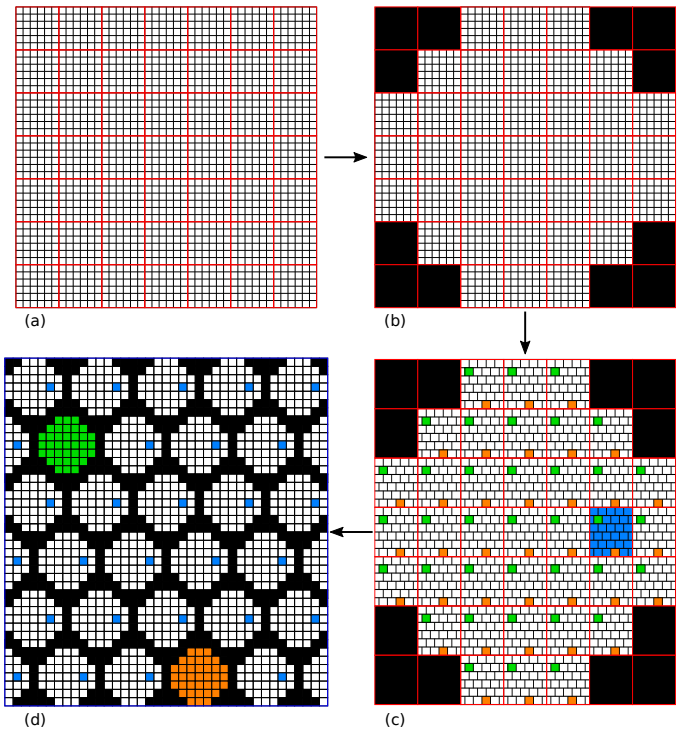


Figure 2.5: From a synthetic light field to a lenslet image: (a) is a 7x7 synthetic light field, the red squares are representing the views and the black squares the pixels of a view; in (b) the views in the corners are removed; in (c) the light field is resampled in order for each view to have an hexagonal grid of pixels; (d) is the final lenslet image.

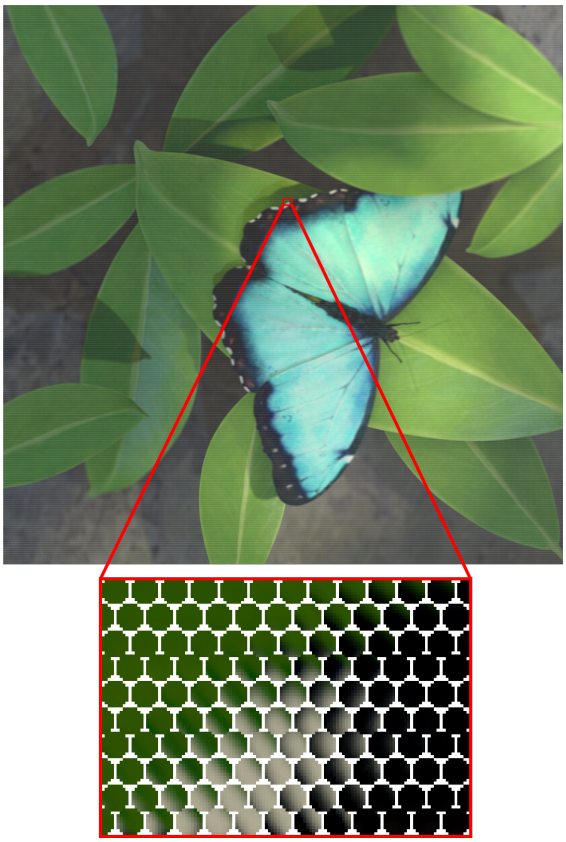


Figure 2.6: Lenslet image of Butterfly [27]

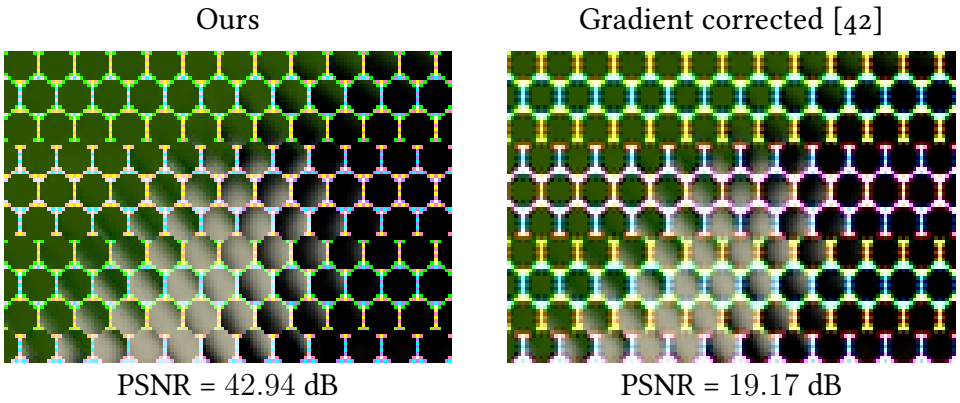
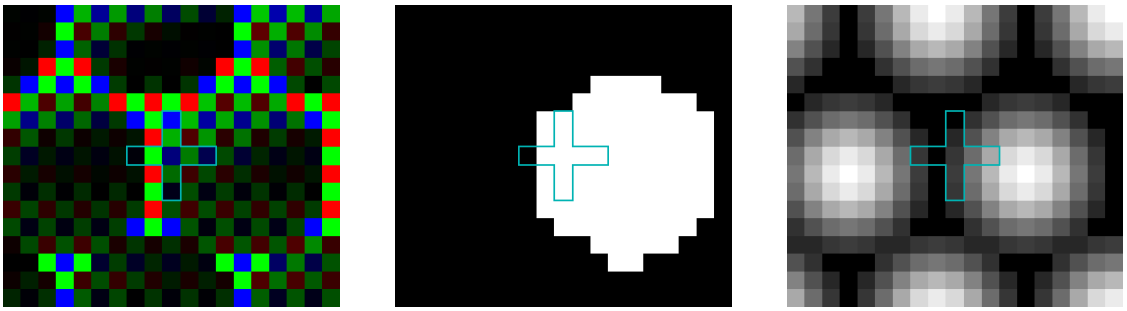


Figure 2.7: Assessment of our demosaicing method on a synthetic lenslet image in comparison with [42]. The pixels which are outside the lenslets are ignored in the PSNR computation.



(a) (b) (c)

Figure 2.8: White lenslet image guided demosaicing: (a) is the raw image we want to demosaic, (b) is a mask which holds every pixel belonging to the same lenslet, (c) is a white image. We use the combination of (b) and (c) as additional weights in the bilinear interpolation and gradient correction described in [42].

position of the lenslets centers and we can identify the pixels which belong to the same lenslets.

- Then with a white image c : we have full confidence in the pixels that have the maximum values on the white image whereas we have less confidence in the pixels that are darker on the white image, as they are noisier.

Let us take the example shown in Figure 2.8 where we want to compute the green value of the blue pixel located in the center of the cross (Image **(a)**). Let (i, j) be the coordinates of the blue pixel, G_{ij} the green value to compute, B_{ij} the measured blue value, (b_{ij}) the mask, (c_{ij}) the white image and (d_{ij}) the coefficients of the kernels mentioned in [42] (see Figure 2.2).

Adapting the equations in [42], the green value is computed as follows:

$$G_{ij} = \sum_{(k,l)\in N} w_{kl}^{\text{bil}} G_{kl} + \alpha\Big(\sum_{\substack{(k,l)\in M \\ d_{kl}>0}} w_{kl}^{\text{grad+}} B_{kl} - \sum_{\substack{(k,l)\in M \\ d_{kl}<0}} w_{kl}^{\text{grad-}} B_{kl}\Big) \tag{2.1}$$

where:

$$\begin{aligned} N &= \{(i-1,j),(i+1,j),(i,j-1),(i,j+1)\} \\ M &= \{(i,j),(i-2,j),(i+2,j),(i,j-2),(i,j+2)\} \end{aligned} \tag{2.2}$$

and:

$$\begin{aligned} w_{kl}^{\text{bil}} &= \frac{e_{kl}}{\sum_{(m,n)\in N} e_{mn}} \\ w_{kl}^{\text{grad+}} &= \frac{f_{kl}}{\sum_{\substack{(m,n)\in M \\ d_{mn}>0}} f_{mn}} \quad , \quad w_{kl}^{\text{grad-}} = \frac{f_{kl}}{\sum_{\substack{(m,n)\in M \\ d_{mn}<0}} f_{mn}} \\ e_{kl} &= b_{kl} \times c_{kl} \\ f_{kl} &= b_{kl} \times c_{kl} \times d_{kl} \end{aligned} \tag{2.3}$$

Every other missing values of each pixel is computed in the same way, using (b_{ij}) and (c_{ij}) as weights (respectively illustrated by (b) and (c) in Figure 2.8). Figure 2.7 (a) and (b) show the results respectively with and without the contribution of the weights (b_{ij}) and (c_{ij}). In this experiment, the PSNR can be computed since the ground truth lenslet image is known. Note that the pixels out of the lenslets are ignored in the PSNR computation since they do not hold any signal and they are not necessary for the view reconstruction. In these conditions, a large PSNR gain of 23.77 dB is observed with our method.

2.3.3 *Proposed alignment method*

As explained above, before extracting the views, the sensor and the lenslet array need to be aligned. This alignment is performed by applying an affine transformation on the pixel array followed by interpolation. As in the case of demosaicing, it is important to account for the borders of the micro-lenses in the interpolation process. Hence, instead of using a a simple bilinear interpolation, we propose a bilinear interpolation weighted by the white image and a mask which eliminates the pixels which do not belong to the same lenslet

To assess the benefit of our method, we first generate a synthetic lenslet image where the lenslets are not perfectly aligned with the pixels. The grid defined by the lenslets is slightly tilted with respect to the pixel grid (the angle is approximately the same as the Lytro 1, $0.05°$). Then, as it is done in the pipeline, we demosaic and apply an affine transformation to the lenslet image in order for the lenslets to be aligned with the pixels, using either a simple bilinear interpolation or our method (see Equations 2.4-2.5) to compute the pixel value I at non-integer position (x,y).

$$I(x,y) = \sum_{(k,l)\in K} w_{kl}^{\text{aff}} I_{kl} \tag{2.4}$$

$$w_{kl}^{\text{aff}} = \frac{g_{kl}}{\sum_{(m,n)\in K} g_{mn}} \qquad g_{kl} = b_{kl} \times c_{kl} \times h_{kl} \tag{2.5}$$

Where K comprises the four nearest neighbor integer positions, b_{kl} and c_{kl} are defined as in the proposed demosaicing method and h_{kl} are the simple bilinear interpolation coefficients.

As the slicing step is just a reorganization of the data to have light field views, it introduces no further error. So, we apply the slicing step to better visualize the impact of the previous rotation and to compare it with the ground truth light field.

As we can see in Figure 2.9, the peripheral views are greatly degraded by the bilinear interpolation. On the other hand, using our white image guided interpolation successfully removes the line artifacts and doubles the global PSNR, going from 19.45 to 38.51 dB.

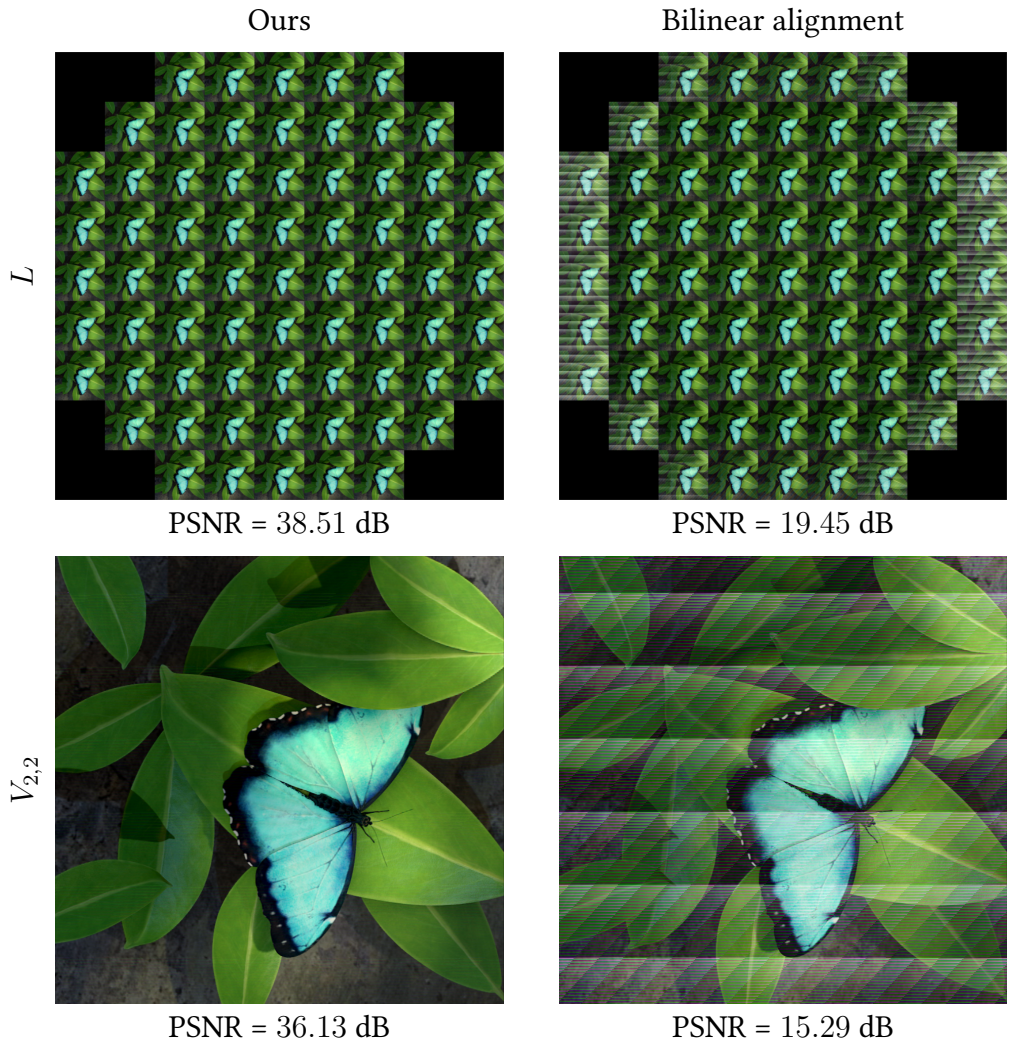


Figure 2.9: Assessment of our aligment method on a synthetic lenslet image in comparison with a simple bilinear interpolation. Top images show the views of the light field, bottom ones show view $V_{2,2}$.

If we use a perfectly demosaiced lenslet image (using the ground truth lenslet image), Dansereau's alignment method gives us a global PSNR of 23.12 dB whereas our method gives a PSNR of 39.84 dB. So, our interpolation method is shown to significantly improve the decoding pipeline for the synthetic lenslet images.

More experimental results on synthetic light fields are shown in Figure 2.10. Our method provides peripheral views with less artifacts than [36].

2.4 EXPERIMENTS ON REAL LENSLET IMAGES

We now test our method on real lenslet images. We use Lytro 1 and Illum images [20]. For a real lenslet image, the pixels at the border of the lenslets are not enough penalized when using the pixel values (c_{ij}) of the white image as coefficients. So we use the pixel values of the white image raised to the power of ten instead, which reduces the weights of the pixels at the border of the lenslets relatively to those at the centers. In Figures 2.11 and 2.12, we can see that the peripheral views are sharper with the proposed demosaicing and alignment methods. Futhermore, thanks to the mask that separates each lenslet, the proposed approach avoids the crosstalk, which visibly reduces the ghosting artifacts of external views generated with the pipeline in [36]. Indeed, we can see on the bottle in the foreground in Figure 2.11 and on the rose petals in Figure 2.12 that our method produces less ghosting artifacts than [36]. Finally, the colors of the peripheral views are closer to the colors of the central view, as raw pixels at the center of the lenslets have more weights than those at the borders that have attenuated colors.

2.5 SUMMARY

We developed a method to create lenslet images from synthetic light fields. With this synthetic lenslet images, we were able to analyze the light field decoding pipeline chosen by JPEG-Pleno. We particularly evaluated the demosaicing and alignment steps. We noticed that the initial demosaicing and interpolation methods used in Dansereau's pipeline were not adapted to the lenslet images and that they created crosstalk artifacts and color patterns. In order to improve these steps, we developed methods of demosaicing and alignment guided by a white lenslet image. These proposed methods showed improvements on the synthetic and on the real lenslet images. The results are shown to reduce crosstalk artifacts and produce better colors on the decoded light fields views. Since the publication that was issued from this work, the proposed demosaicing and alignment methods have been used for a new view extraction pipeline in [48]. They showed that the combination of our contributions with theirs significantly improves the quality of extracted views by performing objective and subjective tests. They also demonstrated the positive impact of the improved views on a number of applications such as compression, editing (e. g., recolorization impainting), or light field rendering.

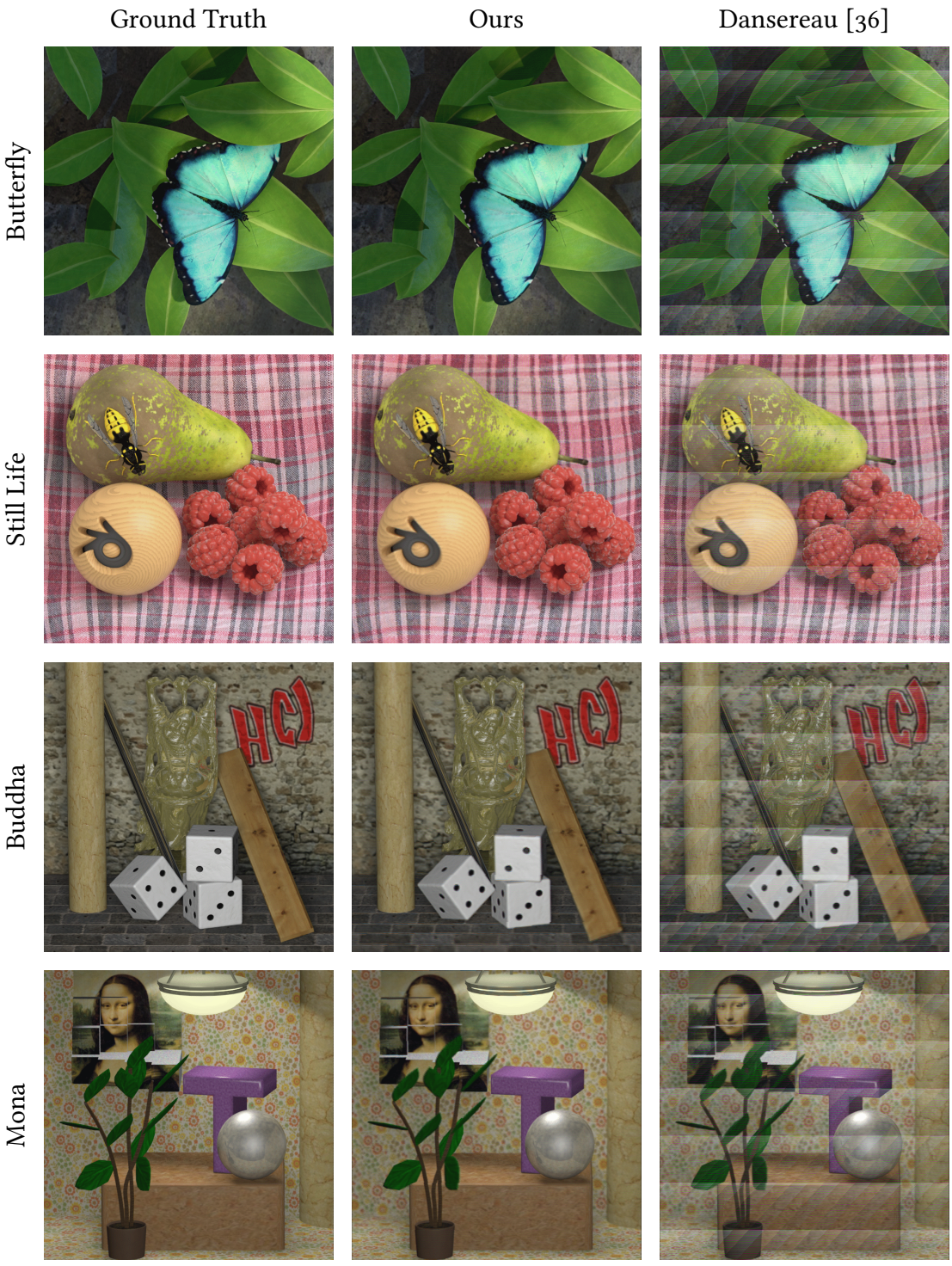


Figure 2.10: Visual comparisons of our improved pipeline with [36] on peripheral view $V_{2,2}$ of synthetic light fields taken from [27].

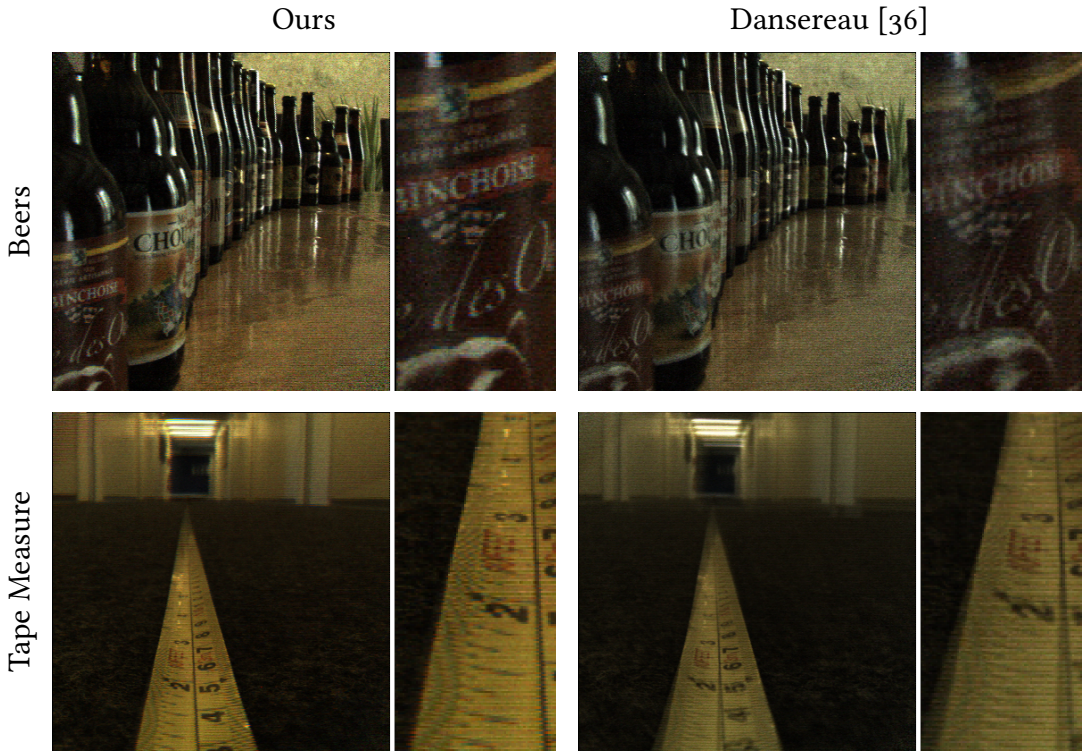


Figure 2.11: Visual comparisons of our improved pipeline with [36] on peripheral view $V_{2,2}$ of Lytro 1 light fields taken from [20].

Figure 2.12: Visual comparisons of our improved pipeline with [36] on peripheral view $V_{2,2}$ of Lytro Illum light fields taken from [20].

Part II

SCENE FLOW ESTIMATION AND TEMPORAL INTERPOLATION

3

STATE OF THE ART

3.1 INTRODUCTION

Light fields, by capturing light rays emitted by a scene along different orientations, enable a variety of computer vision applications, and in particular 3D scene modeling. While the problem of depth estimation for 3D scene modeling has already been widely investigated [49–53], the possibility to estimate the motion in a 3D scene from light fields remains widely open, despite the numerous applications, e. g., for robot navigation, augmented and virtual reality.

The measured displacement of each point in the 3D scene is referred to as a dense scene flow, concept that has first been defined in [54]. Considering a multi-view set-up, the scene flow is estimated using an optical flow estimator for each view. The 3D scene flow is then computed by fitting its projection on each view to the estimated optical flows, hence it is defined by the real 3D motion $(\Delta X, \Delta Y, \Delta Z)$ of each 3D point. However, in the recent literature (e. g., [55–58]), the scene flow is instead defined as a direct extension of the optical flow, where the depth (or disparity) d and the depth variation Δd of objects along time is represented in addition to the apparent 2D motion $(\Delta x, \Delta y)$.

The problem of scene flow analysis has first been addressed for stereo video sequences. The authors in [55–57] estimate a scene flow $(\Delta x, \Delta y, \Delta d, d)$ assuming that the scene can be decomposed into rigidly moving objects and using discrete-continuous optimization techniques. Several methods based on RGB-D videos have also been developed [58–60]. The first methods for scene flow analysis from light fields have been proposed in [61] and [62], based on variational models. The authors in [63] propose oriented light field windows to estimate the scene flow from a dense light field. All these methods rely on epipolar plane images, and hence, are only applicable to densely sampled light fields (as those captured with plenoptic cameras). They are not suitable for sparse light fields (i. e., with large baselines), as for example those captured by rigs of cameras. Before reviewing prior work on scene flow estimation from multi-view captures and from light fields, we will give a quick overview of recent methods proposed for solving two strongly related problems, i. e., scene depth estimation from light fields but also optical flow estimation from videos.

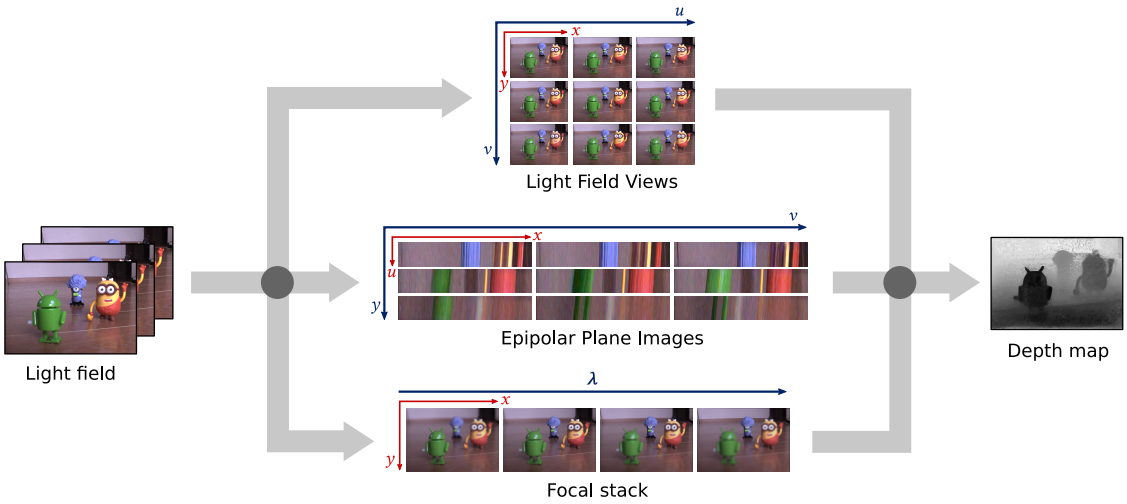


Figure 3.1: Different ways of estimating depth with a light field

Finally, a classical application of motion estimation is frame interpolation. So, the last part of this chapter will be dedicated to the temporal interpolation problem and will give several examples of methods proposed to tackle this issue.

3.2 SCENE DEPTH ESTIMATION FROM LIGHT FIELDS

We can classify depth estimation methods from light fields into three categories (see Figure 3.1) depending on whether they are based on light field views, on epipolar plane images or on focal stacks.

In dense light fields with small baselines, pixels in the different views corresponding to the same 3D point form a line in the EPI, whose slope is proportional to the disparity between the views [64]. This observation naturally led to estimating scene depth (related to the disparity or parallax between the views) by analyzing the EPI of dense light fields. The authors in [49] use structure tensors to locally estimate these slopes, this local estimation being then placed in a global optimization framework using a variational approach. The authors in [50] propose a spinning parallelogram operator for disparity estimation from EPIs, accompanied with a confidence measure to handle ambiguities and occlusions.

While the above methods are well suited for dense light fields, they fail in the case of light fields with large baselines for which stereo matching and optical flow estimation techniques yield more accurate estimates. As a consequence, several methods based on views have been proposed. To give a few examples, the authors in [51] estimate disparity by computing a matching cost volume between the central sub-aperture image and sub-aperture images warped using the phase shift theorem. The approach in [53] consists in estimating disparities between the four corner views, then propagating them to the target viewpoint. The authors in [52] employ an empirical Bayesian framework to estimate scene-dependent parameters for inferring scene disparity.

Numerous depth estimation based on focal stack have been proposed as focal stack are easier to extract from *plenoptic 2.0* cameras than views. A first method was proposed in [65] where the different planes of focus are detected. The proposed method does not however produce a depth map, simply a list of focal values where the captured scene appears sharp in the focal stack. In [66], a focus map is estimated by searching for the strongest gradient in the focal stack, while [67] exploits the symmetry of the focal stack in the focus direction. Last, [40] and [68] combine defocus cues from the focal stack and correspondence cues from views. Note that focal stack based method are often used for densely sampled light fields, since sparsely sampled light fields produce focal stacks with strong artifacts.

Finally, we have recently seen the emergence of deep learning solutions, using in particular convolutional network architectures, for scene depth estimation from light fields. The architectures proposed in [69], [70] operate on EPIs, and hence are well suited for dense light fields only. A deep neural network, called Dispnet, is proposed in [71] based on the optical flow estimation network Flownet2 [72]. It computes 1D correlation instead of 2D correlation to be better suited for disparity estimation. The authors in [73] propose a learning based depth estimation framework suitable for both densely and sparsely sampled light fields, that can learn depth maps for every viewpoint from any subset of input views.

3.3 OPTICAL FLOW ESTIMATION FROM VIDEOS

Since the seminal work of [74] and [75], optical flow estimation has been a prominent issue in computer vision. Optical flow describes the apparent motion in a captured scene (see Figure 3.2). Formally it is often defined as the combination of a horizontal and vertical displacement, respectively denoted Δx and Δy, sometimes the optical flow is also described in the polar coordinates system with an angle θ and a magnitude r.

In order to compare the different methods, benchmark data sets have been proposed. The two most popular datasets are the MPI Sintel Dataset [76] and the KITTI Benchmark [77]. The first one consists in synthetic sequences taken from the movie Sintel. The second one consists in video sequences captured from a moving car, and is therefore better suited for autonomous driving applications.

When looking at the top ranking optical flow estimation methods with the two datasets, we can see that they are almost exclusively using a deep learning approach. To only cite a few methods, FlowNet[78] was the first end-to-end neural network to compute an optical flow from images. It is a trainable encoder-decoder network. The authors in [72] further improve the network by stacking multiple encoder-decoder networks. However, the final network is much bigger than the original one and needs to be trained sequentially for each encoder-decoder part to avoid over-fitting. To reduce the size of the network and make it easier to train, a coarse-to-fine strategy, and the corresponding network called SpyNet, were proposed in [79]. Finally, the authors in [80], as in [79], take advantage of coarse-to-fine approaches, and add a partial cost

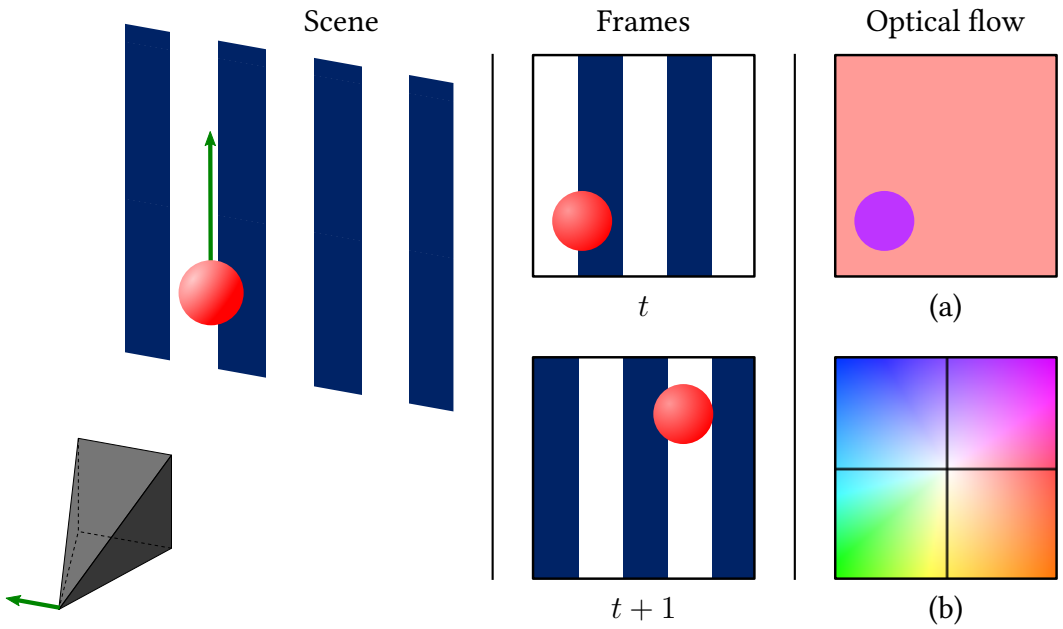


Figure 3.2: The optical flow describes the apparent motion of an object. If a camera moves left while filming a sphere moving up, the direction of the optical flow observed between frames t and $t+1$ will be up right. The optical flow (a) is visualized with the Middlebury color code (b), where the hue and saturation respectively represent the direction and amplitude of the optical flow.

volume computation in their network, named PWC-Net. It is currently one of the top ranking optical flow methods in the MPI Sintel benchmark.

3.4 SCENE FLOW ESTIMATION

The most common way of estimating the scene flow is by using stereo images. The authors in [55] propose a slanted-plane scene flow model for objects in a 3D scene, within the context of autonomous driving. They assume that the scene is composed of a small number of rigidly moving objects and perform a joint segmentation and scene flow estimation. In order to estimate the scene flow model, a discrete-continuous conditional random field is optimized with particle belief propagation [81]. A scene flow model representing the scene with piecewise planar and rigidly moving regions is proposed in [56]. The authors in [57] propose a conditional random field (CRF) based model for robust 3D scene flow estimation. The approach estimates so called *instance scene flows*, i. e., scene flows of 3D points that are geometrically and semantically grouped into instances, using a CNN.

While the models used in these methods are not completely specific for autonomous driving applications, they are however optimized and tested on the KITTI Benchmark [77] which essentially consists of driving scenes. Using any of these methods on other types of scenes may require some changes in the parameters of the models.

Another way to estimate a scene flow is by using RGB-D images. In [59], local and global constraints are combined in a variational framework to estimate a scene flow, assuming a locally rigid motion. The authors use the depth map to regularize the final scene flow with an adaptive total variation formulation. Similarly to [55], the authors in [58] jointly perform segmentation and 3D motion estimation. The scene is decomposed into depth layers to handle occlusions and a scene flow model is computed for each layer. The method in [60] first performs geometric segmentation and then jointly estimates odometry and scene flow by isolating the static clusters.

Scene flow estimation from densely sampled light fields was first tackled in [61]. The authors jointly estimate the disparity and the optical flow assuming piecewise smoothness of the scene flow. A preconditioned primal-dual algorithm is used to solve a convex global energy functional, which also enforces consistency between the multiple views. On the other hand, the authors in [62], [63] and [82] first estimate the geometry of the scene by computing a disparity map and then estimate the apparent motion in the scene. In [62], the disparity value in each point of the EPIs is derived by analyzing the structure tensor. The optical flow is estimated by minimizing an energy function that assumes spatio-angular smoothness and and takes into account occlusions between objects in the scene. The authors in [63] also use an EPI-based method [68] to compute the disparity maps at time t and $t + 1$. Then, they use oriented light-field windows along with a coarse-to-fine strategy to minimize an energy function derived from SimpleFlow [83]. A confidence measure is computed and used to regularize the scene flow in the coarse-to-fine iterations. The authors in [82] fist estimate a disparity map

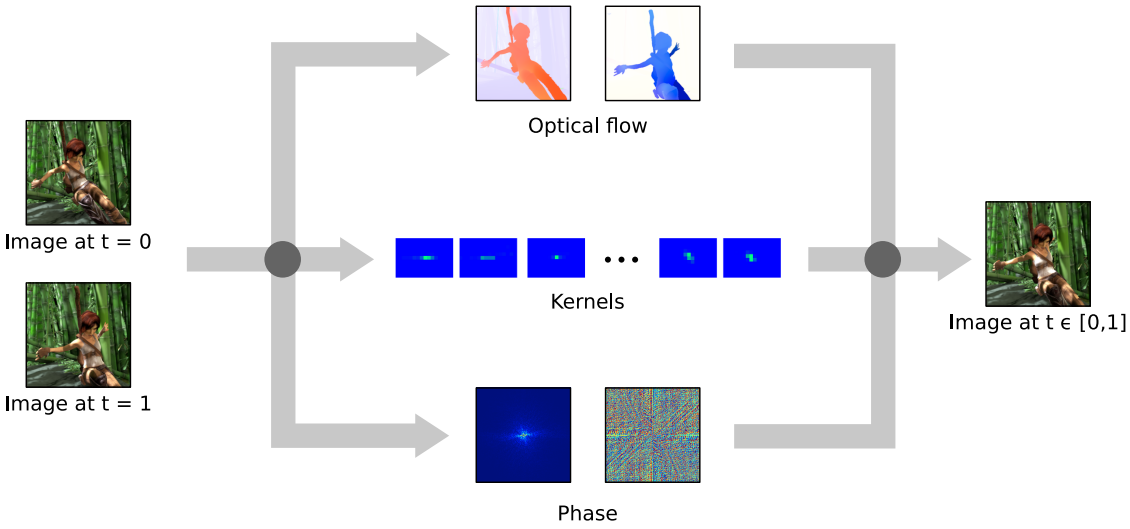


Figure 3.3: Different ways of doing temporal interpolation

using [49] and then recover a 3D scene flow solving a linear flow equation for each ray. This equation, which relies on 4D light field gradients, is under-constrained, so a global and a local approach are combined in order to solve it. The local one is derived from Lucas-Kanade [74] and the global one from Horn-Schunck [75]. The authors in [84] estimate a scene flow to construct a 4D spatio-temporally coherent representation of dynamic scenes from sparse light fields. First, a 3D point cloud is estimated, then every point is back-projected to a more densely sampled virtual light field, and the resulting EPIs are used to compute the scene flow using the oriented window approach [63]. Finally, the authors in [85] use light field super-pixels and their slanted-planes representation in 3D space to propagate and optimize an optical flow and a disparity map from the central view to every other view. The method is mostly fit for dense light fields because accurately computing the normal of the 3D slanted-plane for every super-pixels requires to have a dense set of views.

3.5 VIDEO FRAME INTERPOLATION

Frame interpolation or *temporal interpolation* refers to the interpolation of one or several frames from at least two consecutive frames of a video. It can be used to increase the framerate of a video (i. e., give a *slow-motion* effect to the video) or in the context of video compression, in this case we talk of *motion compensation*.

Let I^0, I^1 be two consecutive frames of a video sequence, the aim of frame interpolation is to generate an intermediate frame I^t, where $t \in [0, 1]$ that is consistent with the two others. Some methods were also developed for frame extrapolation [86], the only difference is that $t > 1$. Last, several methods like [87] use more than two frames as inputs.

With the recent development of deep learning, this field of research has profoundly changed. Today, almost every method that occupies the top of the frame interpolation benchmarks uses deep learning. As showed in Figure 3.3, we can classify interpolation methods in three groups. First, the majority of frame interpolation methods are based on optical flow or at least on motion estimation. Different deep learning models have been proposed that estimate the motion. The authors in [86] use a simple U-Net architecture which is an autoencoder with skipped connections between the encoder and decoder layers. Their network estimates a flow that is used to back-warp and merge the two input images at the intermediate time instant. In [88] and [89], the network is divided into two sub-networks: a first one estimates the optical flow while the second one uses the optical flow and the input frames to produce an intermediate frame.

The second group of frame interpolation methods is based on kernels. For each pixels of the output frame, a kernel is computed to convolve and merge the input frames. In [90, 91], the authors use a U-Net architecture to achieve that. Moreover, some methods use both kernels and motion estimation: the authors in [92, 93] use multiple parallel U-Nets to interpolate the same frame based on either kernels or motion estimation, alongside with occlusion and context extraction. They merge the parallel outputs with a final network to generate the intermediate frame.

Finally, a few methods based on phase shift have been proposed [94, 95]. The authors exploit the idea that a spatial displacement in an image results in a phase shift in the frequency domain. Therefore, these articles use the Fourier transform of the input images, interpolate a new phase image and then apply an inverse Fourier transform to retrieve the intermediate frame.

To the best of our knowledge, only one method has been proposed to temporally interpolate a light field sequence [96]. This approach considers a hybrid capture system composed of a plenoptic camera with a low frame-rate (3 fps) and a classical camera with a standard frame-rate (30 fps). Thanks to a neural network, the frames captured by the classical camera are warped to the different views of the the plenoptic camera.

4

LOCAL 4D AFFINE MODEL FOR SCENE FLOW

4.1 INTRODUCTION

In this chapter, we focus on the problem of scene flow analysis from large baseline light field videos. This problem is made difficult due to the large temporal and angular occlusions. Recent work (detailed in Chapter 3) has shown the important benefits of using deep learning for estimating optical flows, disparity maps or scene flows from stereo images or videos. However, extending and training network architectures that would take light fields as inputs is challenging. First, using light fields as inputs would increase the complexity of the architecture. Then, training a deep neural network typically requires very large datasets, particularly for high dimensional data such as light fields. Only a few light field video datasets are available, which is insufficient for performing unsupervised learning. Furthermore, none of these datasets contain ground truth optical flows, disparity maps or scene flows which would be necessary in the context of supervised learning. To cope with the above difficulties, we propose a 4D local affine model for scene flow estimation. The model is defined in the ray space and incorporates epipolar constraints to ensure consistency of the scene flow on all light field views. We show how the proposed model can be used for (a) interpolating a scene flow from sparse estimates, (b) regularizing initial and independently computed optical flows and disparity maps in order to derive a coherent scene flow. In both applications, to estimate the model, we first perform a 4D over-segmentation of the light field at time t, then we compute initial optical flows, disparity maps and disparity variation estimates between the light field at time t and $t+1$. For each 4D cluster, the parameters of the affine model are estimated by fitting the model on the initial optical flow and disparity estimates. The approach is summarized in Figure 4.1. The proposed regularization method and the corresponding 4D affine model allow us to benefit from state-of-the-art deep learning-based optical flow estimation methods while ensuring scene flow geometry consistency in the 4 dimensions of the light field.

In order to validate the proposed model on sparse light fields, we have created synthetic light field videos based on the Sintel movie (used in the optical flow benchmark [76, 97]). The light field views are provided with the corresponding ground truth scene flow (optical flow, disparity and disparity variation).

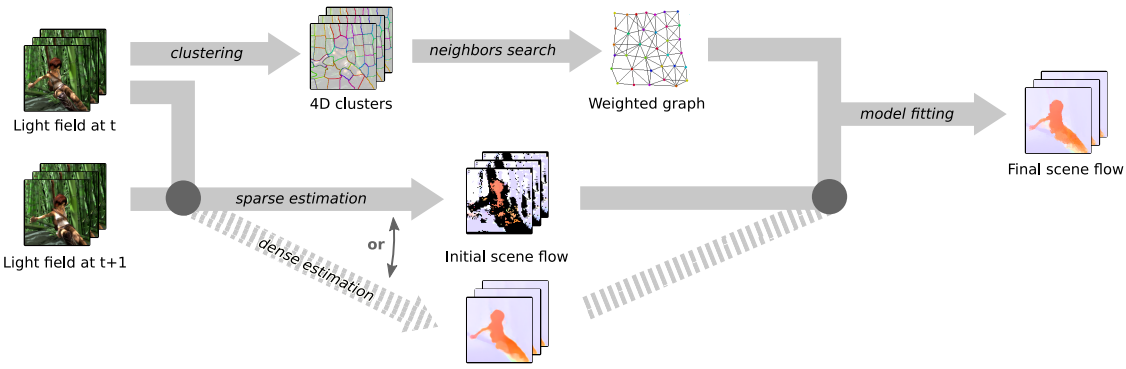


Figure 4.1: Block diagram of our method

Although the proposed scene flow method is designed for sparse light fields, we also assess our method on a dense light field video dataset provided by the authors of [85]. For the sparse dataset, the obtained scene flows are compared against those computed with the oriented window method in [63] and with various stereo scene flow methods [55, 56]. We also compared the estimated optical flow, disparity maps and disparity variation with the one given by a state-of-the-art optical flow estimation technique based on a deep learning architecture called PWC-Net [80]. For the dense dataset, we compared our results with the full view method in [85], with the aforementioned stereo scene flow methods [55, 56] and with various light field depth estimation methods like [51, 98, 99].

While our sparse-to-dense interpolation method yields comparable results on optical flow and disparity variation as the state-of-the-art methods, our regularized scene flow estimation outperforms any other tested method in terms of accuracy of the estimated optical flow, disparity, and disparity variation for the sparse dataset, and achieves comparable results to state-of-the-art methods for the dense dataset.

4.2 4D AFFINE MODEL

Let us consider a light field video. We denote L^t the light field frame at time instant t. A view (u, v) of this frame is written V^t_{uv}. In the rest of the chapter, we assume that the vertical and horizontal baselines are the same. The scene flow can be divided in the following components:

- the optical flow: $F = \begin{pmatrix} \Delta x & \Delta y \end{pmatrix}^\top$,
- the disparity at time t: d^t,
- the disparity variation between t and $t+1$: Δd.

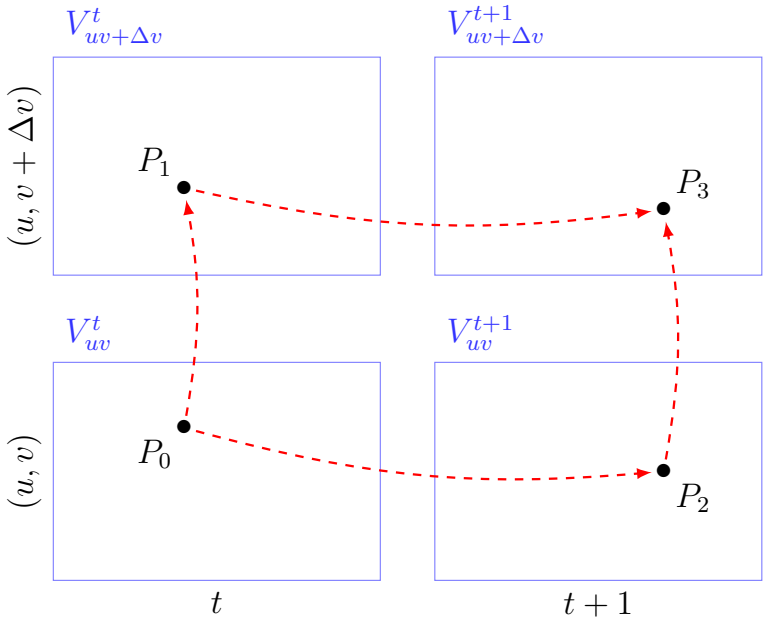


Figure 4.2: Projections of one 3D scene point on 2 views of the light field at time instants t and $t+1$

In this chapter, we propose a local 4D affine model to represent a scene flow in a light field. The fundamental affine model can be defined as follows:

$$\Delta x(u,v,x,y) = \theta^0_1 u + \theta^0_2 v + \theta^0_3 x + \theta^0_4 y + \theta^0_5, \tag{4.1}$$
$$\Delta y(u,v,x,y) = \theta^0_6 u + \theta^0_7 v + \theta^0_8 x + \theta^0_9 y + \theta^0_{10}, \tag{4.2}$$
$$d^t(u,v,x,y) = \theta^0_{11} u + \theta^0_{12} v + \theta^0_{13} x + \theta^0_{14} y + \theta^0_{15}, \tag{4.3}$$
$$\Delta d(u,v,x,y) = \theta^0_{16} u + \theta^0_{17} v + \theta^0_{18} x + \theta^0_{19} y + \theta^0_{20}, \tag{4.4}$$

where $\boldsymbol{\theta^0} = \begin{pmatrix}\theta^0_1 & \dots & \theta^0_{20}\end{pmatrix}^\top$ are the parameters of the model.

However, this model does not take into account epipolar geometry of light fields, i. e.the fact that a 3D point in a scene is projected on 1D lines in EPIs, the slope of these lines being directly related to inter-view disparity. Hence, we derive in this section the equations for a reduced affine model that also satisfies the epipolar constraints.

4.2.1 *Constraints on the Optical Flow*

First, let us consider the vertical epipolar constraints. Given a non-occluded point in the 3D scene, we denote by P_0, P_1, P_2 and P_3 its respective projections on the views (u, v) and $(u, v+\Delta v)$ at the time instants t and $t+1$, as shown in Figure 4.2.

The coordinates of the points P_0, P_1, P_2 and P_3 in the 4-dimensional space (u, v, x, y) are then related with the following equation:

$$\begin{cases} P_1 = P_0 + \Delta v \cdot \boldsymbol{\nu}(P_0, t), \\ P_2 = P_0 + \boldsymbol{\phi}(P_0), \\ P_3 = P_2 + \Delta v \cdot \boldsymbol{\nu}(P_2, t+1), \\ P_3 = P_1 + \boldsymbol{\phi}(P_1), \end{cases} \tag{4.5}$$

where $\boldsymbol{\nu}(P, t)$ and $\boldsymbol{\phi}(P)$ are 4-dimensional vectors representing respectively the orientation of the vertical epipolar line passing by a point P at time t, and the optical flow of P from time t to $t+1$. These vectors are expressed as:

$$\boldsymbol{\nu}(P, t) = \begin{pmatrix} 0 \\ 1 \\ 0 \\ d^t(P) \end{pmatrix} \quad \text{and} \quad \boldsymbol{\phi}(P) = \begin{pmatrix} 0 \\ 0 \\ \Delta x(P) \\ \Delta y(P) \end{pmatrix}. \tag{4.6}$$

We can derive from Equation (4.5) that the optical flow vectors $\boldsymbol{\phi}(P_0)$ and $\boldsymbol{\phi}(P_1)$ must satisfy the following equality to be angularly consistent:

$$\boldsymbol{\phi}(P_1) - \boldsymbol{\phi}(P_0) = \Delta v \cdot \left[\boldsymbol{\nu}(P_2, t+1) - \boldsymbol{\nu}(P_0, t)\right]. \tag{4.7}$$

From the definition of $\boldsymbol{\nu}$ and $\boldsymbol{\phi}$ in Equation (4.6), this equality can be rewritten:

$$\begin{cases} \Delta x(P_1) - \Delta x(P_0) = 0, \\ \Delta y(P_1) - \Delta y(P_0) = \Delta v \cdot \Delta d(P_0), \end{cases} \tag{4.8}$$

where we define $\Delta d(P_0) = d^{t+1}(P_2) - d^t(P_0)$.

Let us now reintegrate these constraints into the affine model. Knowing the relationship between the coordinates of P_0 and P_1 in Equation (4.5) and the expressions of Δx and Δy in Equations (4.1) and (4.2), we can express the variation of optical flow between P_0 and P_1 (i. e.along a vertical epipolar line) as a function of the model's parameters:

$$\begin{cases} \Delta x(P_1) - \Delta x(P_0) = \Delta v \left(\theta_2^0 + \theta_4^0 \times d^t(P_0)\right), \\ \Delta y(P_1) - \Delta y(P_0) = \Delta v \left(\theta_7^0 + \theta_9^0 \times d^t(P_0)\right). \end{cases} \tag{4.9}$$

By combining Equations (4.8) and (4.9), we obtain the following constraints on the model's parameters:

$$\theta_2^0 + \theta_4^0 \times d^t(P_0) = 0, \tag{4.10}$$

$$\theta_7^0 + \theta_9^0 \times d^t(P_0) = \Delta d(P_0). \tag{4.11}$$

Similarly, horizontal epipolar constraints give:

$$\theta_1^0 + \theta_3^0 \times d^t(P_0) = \Delta d(P_0), \tag{4.12}$$
$$\theta_6^0 + \theta_8^0 \times d^t(P_0) = 0. \tag{4.13}$$

Note that one could directly replace $d^t(P_0)$ and $\Delta d(P_0)$ by their expressions in Equations (4.3) and (4.4). However, the model would lose its linearity and become more complex to solve. Instead, we choose to approximate the disparity $d^t(P_0)$ by a pre-estimated disparity value $\overline{d}$. The derivation of $\overline{d}$ is detailed in Section 4.3 (see Equation (4.36)). We also eliminate $\Delta d(P_0)$ by taking the difference between Equations (4.11) and (4.12). We can then simplify our model and reduce the number of parameters as

$$\begin{aligned} \theta_2^0 &= -\theta_4^0 \times \overline{d}, \\ \theta_7^0 &= \theta_1^0 + \theta_3^0 \times \overline{d} - \theta_9^0 \times \overline{d}, \\ \theta_6^0 &= -\theta_8^0 \times \overline{d}. \end{aligned} \tag{4.14}$$

So, the optical flow model becomes

$$\begin{aligned} \Delta x(P_0) &= \theta_1^0 u + \theta_3^0 x + \theta_4^0 \times (y - \overline{d}v) + \theta_5^0, \\ \Delta y(P_0) &= \theta_1^0 v + \theta_3^0 \overline{d} v + \theta_8^0 (x - \overline{d}u) + \theta_9^0 (y - \overline{d}v) + \theta_{10}^0. \end{aligned} \tag{4.15}$$

4.2.2 *Constraints on the Disparity and Disparity Variation*

Furthermore, we can also reduce the number of parameters of the disparity and the disparity variation models. The epipolar geometry of a light field requires that the disparity remains constant along a vertical or horizontal epipolar line. This constraint gives the following equations:

$$\begin{aligned} d^t(P_0 + \Delta v \cdot \boldsymbol{\nu}(P_0, t)) - d^t(P_0) &= 0, \\ d^t(P_0 + \Delta u \cdot \boldsymbol{\mu}(P_0, t)) - d^t(P_0) &= 0, \\ \Delta d(P_0 + \Delta v \cdot \boldsymbol{\nu}(P_0, t)) - \Delta d(P_0) &= 0, \\ \Delta d(P_0 + \Delta u \cdot \boldsymbol{\mu}(P_0, t)) - \Delta d(P_0) &= 0. \end{aligned} \tag{4.16}$$

By replacing the terms in Equation (4.16) by the expression of their model in Equations (4.3)-(4.4), we obtain the additional constraints:

$$\begin{aligned} \theta_{12}^0 &= -\theta_{14}^0 \times \overline{d}, \\ \theta_{11}^0 &= -\theta_{13}^0 \times \overline{d}, \\ \theta_{17}^0 &= -\theta_{19}^0 \times \overline{d}, \\ \theta_{16}^0 &= -\theta_{18}^0 \times \overline{d}. \end{aligned} \tag{4.17}$$

The disparity and disparity variation models thus become:

$$d^t(P_0) = \theta^0_{13}(x - \overline{d}u) + \theta^0_{14}(y - \overline{d}v) + \theta^0_{15}, \tag{4.18}$$
$$\Delta d(P_0) = \theta^0_{18}(x - \overline{d}u) + \theta^0_{19}(y - \overline{d}v) + \theta^0_{20}. \tag{4.19}$$

This allows us to reduce again the number of parameters from 17 to 13. We denote $\boldsymbol{\theta} = \begin{pmatrix}\theta_1 & \dots & \theta_{13}\end{pmatrix}^\top$ the new parameters. The final scene flow model is the following:

$$\begin{aligned}
\Delta x(P_0) &= \theta_1 u + \theta_2 x + \theta_3(y - \overline{d}v) + \theta_4, \\
\Delta y(P_0) &= \theta_1 v + \theta_2 \overline{d}v + \theta_5(x - \overline{d}u) + \theta_6(y - \overline{d}v) + \theta_7, \\
d^t(P_0) &= \theta_8(x - \overline{d}u) + \theta_9(y - \overline{d}v) + \theta_{10}, \\
\Delta d(P_0) &= \theta_{11}(x - \overline{d}u) + \theta_{12}(y - \overline{d}v) + \theta_{13}.
\end{aligned} \tag{4.20}$$

4.3 ESTIMATING THE MODEL PARAMETERS

4.3.1 *Clustering the light field*

The model previously described works under one assumption: our model is affine, so the scene flow should not have discontinuities. As a consequence, we can partition our light field into clusters that respect the assumption and fit one model for each cluster. If the clusters correspond to the same object in the scene, the assumption will be valid. We therefore group pixels of similar color across the views and corresponding to the same scene area in 4D clusters, using the method proposed in [100]. The method is inspired by the SLIC algorithm[101]. Centroids are first initialised on a reference view. Their disparity is then estimated and used to project the centroids to all the views. A k-means clustering is then simultaneously performed on all views. Using the centroid disparity, all the rays assigned to a cluster are projected back to the reference view to update the centroids colors and spatial positions. The approach is fast, free of any strong scene geometry prior and does not require a dense depth map estimation. For these reasons, we chose this method for our model.

To estimate the model parameters for each cluster, the model is fitted to the initial optical flow and disparity estimates available in each cluster. The number of estimates may however not be sufficient in some clusters. For this reason, we propose to build a graph connecting the different clusters. This graph enables us to look for the N nearest clusters of a given cluster, adding more estimates in the computation of the scene flow model of a given cluster.

4.3.2 *Connecting the clusters with a weighted graph*

In order to connect the different clusters, we build an undirected weighted graph $\mathcal{G} = \{\mathcal{V}, \mathcal{E}, w\}$. $\mathcal{V}$ is the set of the K clusters computed in the clustering step (see

Figure 4.1). A vertex i is connected to another one j if their corresponding clusters are adjacent to one another in at least one view of the light field or if they are in the same range of disparity, that is if

$$|d^t(C_i) - d^t(C_j)| < \beta \left(\max_{k \in \mathcal{V}} d^t(C_k) - \min_{k \in \mathcal{V}} d^t(C_k) \right), \tag{4.21}$$

when C_i and C_j denote the clusters centroids and $\beta \in [0, 1]$ a threshold coefficient. In the experiments, we fix $\beta = 0.1$.

The weight between two connected nodes i and j is defined as

$$w(i, j) = \min_{(u,v) \in \Omega_{ij}} \exp\left[-\alpha D\left(\mathcal{P}_{uv}(C_i), \mathcal{P}_{uv}(C_j)\right)\right], \tag{4.22}$$

where Ω_{ij} is the set of views where i and j are adjacent, $\mathcal{P}_{uv}(C)$ the projection of the centroid C on the view (u, v) and α a parameter that we empirically fix to 0.2. The distance D is based on spatial and color proximity and defined as in [100]:

$$D = \sqrt{{d_c}^2 + \frac{m^2}{S^2} {d_s}^2} \tag{4.23}$$

where $S = \sqrt{H \times W / K}$ with W and H the width and height of a view. The parameter m has the same value as for the clustering step, it is used in the clustering step to control the compactness of the clusters. d_c and d_s are the color and spatial distances respectively defined as euclidean distances in the CIELAB colorspace and the $[xy]$ space:

$$d_c(C_i, C_j) = \sqrt{(L_i - L_j)^2 + (a_i - a_j)^2 + (b_i - b_j)^2} \tag{4.24}$$

$$d_s(C_i, C_j) = \sqrt{(x_i - x_j)^2 + (y_i - y_j)^2} \tag{4.25}$$

Once the graph is computed, we can look for the N nearest neighbors of a given node i using Dijkstra's algorithm [102]. In the search, we discard every vertex whose corresponding cluster contains no scene flow estimate (which can happen when the initial scene flow is sparse). The set of N neighbors of i (including itself) is denoted $\mathcal{N}_i$. It is used to have more scene flow estimates than those inside the cluster and in particular when the cluster i has no estimate inside itself.

4.3.3 *Fitting a model with RANSAC*

The approach we use to fit the model described in Section 4.2 to the scene flow estimates that we have is inspired from the RANSAC method[103]. The general idea is to choose m scene flow estimates, to compute the parameters of the model and then to evaluate the cost of the model.

Let SF_i be the set of initial scene flow estimates contained in the cluster i, we have

$$\mathrm{SF}_i = \begin{pmatrix} u_1 & v_1 & x_1 & y_1 & \Delta x_1 & \Delta y_1 & d_1^t & \Delta d_1 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ u_{n_i} & v_{n_i} & x_{n_i} & y_{n_i} & \Delta x_{n_i} & \Delta y_{n_i} & d_{n_i}^t & \Delta d_{n_i} \end{pmatrix} \tag{4.26}$$

Equation (4.20) is linear in $\boldsymbol{\theta}$ so we build a block matrix A_i and vector b_i such that $\left\| A_i \hat{\boldsymbol{\theta}} - b_i \right\|_2$ represents the fidelity of a model $\hat{\boldsymbol{\theta}}$ to the initial scene estimates SF_i.

$$A_i = \begin{pmatrix} A_i^x & 0 & 0 \\ A_i^y & 0 & 0 \\ 0 & A_i^d & 0 \\ 0 & 0 & A_i^\Delta \end{pmatrix} \tag{4.27}$$

where the sub-matrices A_i^x, A_i^y, A_i^d and A_i^Δ are defined as

$$A_i^x = \begin{pmatrix} u_1 & x_1 & y_1 - \overline{d_i} v_1 & 1 & 0 & 0 & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ u_{n_i} & x_{n_i} & y_{n_i} - \overline{d_i} v_{n_i} & 1 & 0 & 0 & 0 \end{pmatrix} \tag{4.28}$$

$$A_i^y = \begin{pmatrix} v_1 & \overline{d_i} v_1 & 0 & 0 & x_1 - \overline{d_i} u_1 & y_1 - \overline{d_i} v_1 & 1 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ v_{n_i} & \overline{d_i} v_{n_i} & 0 & 0 & x_{n_i} - \overline{d_i} u_{n_i} & y_{n_i} - \overline{d_i} v_{n_i} & 1 \end{pmatrix} \tag{4.29}$$

$$A_i^d = A_i^\Delta = \begin{pmatrix} x_1 - \overline{d_i} u_1 & y_1 - \overline{d_i} v_1 & 1 \\ \vdots & \vdots & \vdots \\ x_{n_i} - \overline{d_i} u_{n_i} & y_{n_i} - \overline{d_i} v_{n_i} & 1 \end{pmatrix} \tag{4.30}$$

Let b_i be the corresponding vector to A_i:

$$b_i = \begin{pmatrix} b_i^x & b_i^y & b_i^d & b_i^\Delta \end{pmatrix}^\top \tag{4.31}$$

with:

$$b_i^x = \begin{pmatrix} \Delta x_1 & \cdots \Delta x_{n_i} \end{pmatrix} \tag{4.32}$$

$$b_i^y = \begin{pmatrix} \Delta y_1 & \cdots \Delta y_{n_i} \end{pmatrix} \tag{4.33}$$

$$b_i^d = \begin{pmatrix} d_1^t & \cdots d_{n_i}^t \end{pmatrix} \tag{4.34}$$

$$b_i^\Delta = \begin{pmatrix} \Delta d_1 & \cdots & \Delta d_{n_i} \end{pmatrix} \tag{4.35}$$

In order to make our model linear, we approximate the disparity in Equations (4.10), (4.11), (4.12) and (4.13) by a pre-estimated disparity value. We compute one disparity

estimate per cluster $\overline{d}_i$ by averaging the disparity estimates contained in the cluster i and in its neighbors as

$$\overline{d}_i = \frac{\sum\limits_{j \in \mathcal{N}_i} e^{-\lambda w(i,j)} \sum\limits_{k=1}^{n_j} b^d_{j,k}}{\sum\limits_{j \in \mathcal{N}_i} e^{-\lambda w(i,j)} n_j} \tag{4.36}$$

The parameter λ controls the weight of the neighboring clusters in the average computation and $b^d_{j,k}$ denotes the kth element of vector b^d_j.

The more constant the disparity is in a cluster, the more correct the approximation is. For each cluster i, we search for the parameters $\boldsymbol{\theta}$ of the model (4.20) that minimize the following cost function:

$$\mathcal{L}_i(\boldsymbol{\theta}) = \sum_{j \in \mathcal{N}_i} e^{-\lambda w(i,j)} \cdot f_j(\boldsymbol{\theta}) \tag{4.37}$$

where $f_j(\boldsymbol{\theta})$ is the number of outliers produced by the model $\boldsymbol{\theta}$ among the estimates which are inside the cluster j, that can be formally defined as:

$$f_j(\boldsymbol{\theta}) = \sum_{k=0}^{4n_j} [\![|A_{j,k}\boldsymbol{\theta} - b_{j,k}| > \tau]\!] \tag{4.38}$$

The symbols $[\![\cdot]\!]$ denote the Iverson brackets, which return 1 if the proposition inside the brackets is true and 0 otherwise. $A_{j,k}$ denotes the kth row of A_j. The hyperparameter τ is analogous to the threshold defined in the classical RANSAC algorithm. It is fixed to 5 in our experiments. As with RANSAC algorithm, we generate an hypothesis $\hat{\boldsymbol{\theta}}$ for our model, we compute its cost function $\mathcal{L}_i(\hat{\boldsymbol{\theta}})$ and compare it with the best candidate $\hat{\boldsymbol{\theta}}_{\text{min}}$ that we found so far (the one with the lowest cost function). We iterate N_{iter} times.

Before iterating, we initialize our model to a constant model: we set every coordinate of $\hat{\boldsymbol{\theta}}$ to 0 except for $\hat{\theta}_4$, $\hat{\theta}_7$, $\hat{\theta}_{10}$ and $\hat{\theta}_{13}$. This way, the scene flow inside a cluster is constant and equal to the weighted average scene flow estimate.

$$\begin{aligned}
\hat{\theta}_4 &= \overline{\Delta x_i} = \frac{\sum\limits_{j \in \mathcal{N}_i} e^{-\lambda w(i,j)} \sum\limits_{k=1}^{n_j} b^x_{j,k}}{\sum\limits_{j \in \mathcal{N}_i} e^{-\lambda w(i,j)} n_j} \\
\hat{\theta}_7 &= \overline{\Delta y_i} = \frac{\sum\limits_{j \in \mathcal{N}_i} e^{-\lambda w(i,j)} \sum\limits_{k=1}^{n_j} b^y_{j,k}}{\sum\limits_{j \in \mathcal{N}_i} e^{-\lambda w(i,j)} n_j} \\
\hat{\theta}_{10} &= \overline{d}_i \\
\hat{\theta}_{13} &= \overline{\Delta d_i} = \frac{\sum\limits_{j \in \mathcal{N}_i} e^{-\lambda w(i,j)} \sum\limits_{k=1}^{n_j} b^\Delta_{j,k}}{\sum\limits_{j \in \mathcal{N}_i} e^{-\lambda w(i,j)} n_j}
\end{aligned} \tag{4.39}$$

What differs from classical RANSAC is the hypothesis generation. Classically, we would randomly choose 13 rows from $\{A_j \mid j \in \mathcal{N}_i\}$ to form a matrix A_s and the corresponding vector b_s, and we would compute - if possible - $\hat{\boldsymbol{\theta}} = A_s^{-1} b_s$.

In our case, the process of selection of estimates is not totally random. At every iteration, we want to generate stable parameters $\hat{\boldsymbol{\theta}}$. So, we need to form a matrix A_s with a low condition number, which means with rows that are the most linearly independent from one another.

Inspired by the work in [104], we propose to choose the samples in a careful way in order to perform our hypothesis generation step. More precisely, given a cluster i, we first build the matrix and vector U_i and v_i such that

$$U_i = \bigoplus_{j \in \mathcal{N}_i} A_j \qquad \text{and} \qquad v_i = \bigoplus_{j \in \mathcal{N}_i} b_j, \tag{4.40}$$

where $\bigoplus_k X_k$ denotes the vertical concatenation of the matrices X_k.

Let M be the number of rows of U_i and v_i. Our goal is to find a set $\mathcal{S}$ of 13 linearly independent rows among the M rows of U_i. The general idea is to iteratively add samples to $\mathcal{S}$ taking into account the previous added samples.

We start with an empty set $\mathcal{S}$. The first sample which is added is randomly chosen. Then, for every iteration n from 2 to 13, we add a nth sample to the set $\mathcal{S}$. To do so, we build the matrix $U_i(\mathcal{S}, \mathcal{T})$ with $\mathcal{T}$ being the range $[1, n]$. The resulting matrix of size $(n, n-1)$ is of rank $n-1$ because every row is independent from one another. The nullspace of such matrix gives us the unique vector z that is orthogonal to all rows of this matrix. Then, in the normalized version of $U_i(\mathcal{R}, \mathcal{T})$ (with $\mathcal{R} = [1, M]$), we search for the row that is the most linearly dependent on the null vector, i. e., the most linearly independent of the rows of $U_i(\mathcal{S}, \mathcal{T})$. This constitutes the new sample added to our set. We continue until we can reach our 13 samples. Once the set $\mathcal{S}$ is complete, we can build a matrix and a vector $A_s = U_i(\mathcal{S}, \mathcal{T})$ and $b_s = v_i(\mathcal{S})$ with $\mathcal{T} = [1, 13]$. We finally generate an hypothesis $\hat{\boldsymbol{\theta}} = \arg\min_{\boldsymbol{\theta}} \|A_s \boldsymbol{\theta} - b_s\|_2$. The hypothesis generation is fully detailed in Algorithm 1.

Another change to the classic RANSAC algorithm is that for each iteration and for each cluster, we also evaluate the models given by the neighboring clusters. This allows us to propagate correct models among the clusters and to make the algorithm converge faster.

4.4 APPLICATIONS

4.4.1 *Sparse-to-dense interpolation*

The method takes a reference view V_{ref}^t (usually we choose the central view). We denote by $C : (I, I', \mathcal{S}) \to \mathcal{S}'$ the function that determines the set of points $\mathcal{S}'$ in an image I' that matches the set of seeds $\mathcal{S}$ in another image I.

ALGORITHM 1: Hypothesis generation for cluster i

Input: Matrix U_i and its corresponding vector v_i
Output: An hypothesis $\hat{\boldsymbol{\theta}}$ for the cluster i

$r_0 \leftarrow \text{selectRandomRow}(U_i)$;
$\mathcal{S} \leftarrow \{r_0\}$;
$\mathcal{R} \leftarrow [1, M]$;
for $n = 2 \rightarrow 13$ **do**
  $\mathcal{T} \leftarrow [1, n]$;
  $z \leftarrow \text{nullspace}(U_i(\mathcal{S}, \mathcal{T}))$;
  $\text{normalizeRows}(U_i(\mathcal{R}, \mathcal{T}))$;
  $b \leftarrow U_i(\mathcal{R}, \mathcal{T})z$;
  $r \leftarrow \arg\max_{k \in \mathcal{R}} |b_k|$;
  $\mathcal{S} \leftarrow \mathcal{S} \cup \{r\}$
end
$A_s \leftarrow U_i(\mathcal{S}, \mathcal{T})$;
$b_s \leftarrow v_i(\mathcal{S})$;
$\hat{\boldsymbol{\theta}} \leftarrow \arg\min_{\boldsymbol{\theta}} \|A_s\boldsymbol{\theta} - b_s\|_2$;

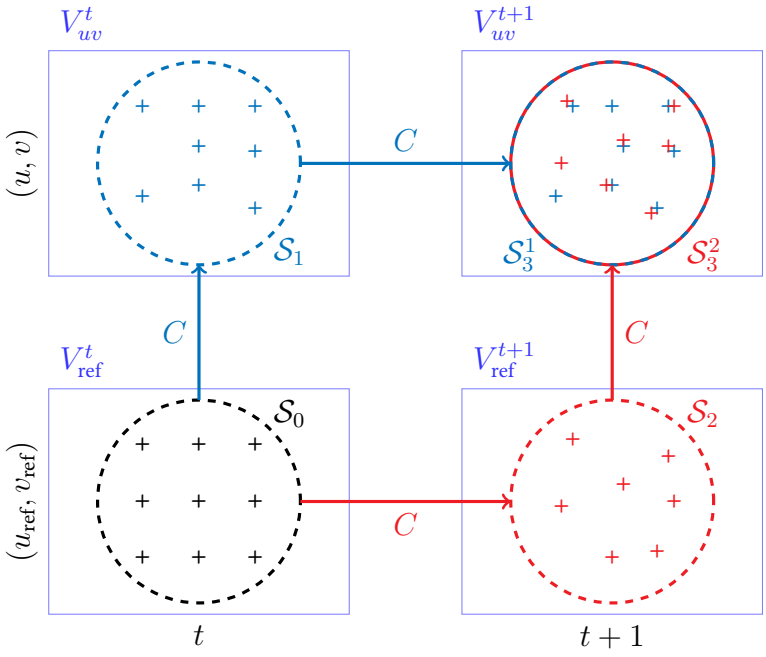


Figure 4.3: Sparse estimation of scene flow. A set of points $\mathcal{S}_0$ is initialized on a regular grid in $(u_{\text{ref}}, v_{\text{ref}})$ at time t. Then, a matching function C is used to respectively find the correspondences $\mathcal{S}_1$ and $\mathcal{S}_2$ of $\mathcal{S}_0$ in another view (u, v) at t and in the same view at $t + 1$. Again the matching function C is used to find the respective correspondences $\mathcal{S}_3^1$ and $\mathcal{S}_3^2$ of $\mathcal{S}_1$ and $\mathcal{S}_2$ at the view (u, v) at $t + 1$. The final step selects only the points that are both in $\mathcal{S}_3^1$ and $\mathcal{S}_3^2$ and estimates the scene flow at these points.

We initialize a set of points $\mathcal{S}_0$ regularly distributed on a grid in a reference view of the light field at t, denoted L^t_{ref}. Then, for every view that is not the reference view, we successively compute

$$\begin{aligned} \mathcal{S}_1 &= C(V^t_{\text{ref}}, V^t_{uv}, \mathcal{S}_0), & \mathcal{S}_2 &= C(V^t_{\text{ref}}, V^{t+1}_{\text{ref}}, \mathcal{S}_0) \\ \mathcal{S}^1_3 &= C(V^t_{uv}, V^{t+1}_{uv}, \mathcal{S}_1), & \mathcal{S}^2_3 &= C(V^{t+1}_{\text{ref}}, V^{t+1}_{uv}, \mathcal{S}_2). \end{aligned} \tag{4.41}$$

For each point $P_0 \in \mathcal{S}_0$, we check if the seed has correspondences in $\mathcal{S}^1_3$ and $\mathcal{S}^2_3$. Furthermore, we check if the correspondences are consistent with each other. That is if $D(P^1_3, P^2_3) < \tau$, where D is the distance described in Equation (4.23) and τ a distance threshold that is fixed to 10 in our experiments. The method is summarized in Figure 4.3.

Any remaining chain of points $\boldsymbol{g} = (P_0, P_1, P_2, P^1_3, P^2_3)$ is used to estimate a sparse scene flow. From $\boldsymbol{g}$, we can compute the scene flow of two points of the light field L^t: on the reference view V^t_{ref} at P_0 and on the view V^t_{uv} at P_1. With (x_0, y_0), (x_1, y_1), (x_2, y_2), (x^1_3, y^1_3) and (x^2_3, y^2_3) the respective spatial coordinates of P_0, P_1, P_2, P^1_3 and P^2_3, we have

$$F_{\text{init}}(P_0) = \begin{pmatrix} \Delta x_{\text{init}}(P_0) \\ \Delta y_{\text{init}}(P_0) \end{pmatrix} = \begin{pmatrix} x_2 - x_0 \\ y_2 - y_0 \end{pmatrix} \tag{4.42}$$

$$F_{\text{init}}(P_1) = \begin{pmatrix} \Delta x_{\text{init}}(P_1) \\ \Delta y_{\text{init}}(P_1) \end{pmatrix} = \begin{pmatrix} x^1_3 - x_1 \\ y^1_3 - y_1 \end{pmatrix} \tag{4.43}$$

Let d^t_{init}, d^{t+1}_{init} and Δd_{init} be the disparities at t and $t+1$ and the disparity variation of both P_0 and P_1. We have:

$$d^t_{\text{init}} = \frac{|\Delta u| \dfrac{x_1 - x_0}{\Delta u} + |\Delta v| \dfrac{y_1 - y_0}{\Delta v}}{|\Delta u| + |\Delta v|} \tag{4.44}$$

$$d^{t+1}_{\text{init}} = \frac{|\Delta u| \dfrac{x^2_3 - x_2}{\Delta u} + |\Delta v| \dfrac{y^2_3 - y_2}{\Delta v}}{|\Delta u| + |\Delta v|} \tag{4.45}$$

$$\Delta d_{\text{init}} = d^{t+1}_{\text{init}} - d^t_{\text{init}} \tag{4.46}$$

with $\Delta u = u_{\text{ref}} - u$ and $\Delta v = v_{\text{ref}} - v$

Once we have these sparse scene flow estimates, we can perform an over-segmentation of the light field, use the estimates and the clusters as inputs (see Figure 4.1) to estimate the dense 4D scene flow model in each cluster, as described in Section 4.2.

4.4.2 *Regularization*

Most recent methods to estimate optical flows or disparity maps use deep neural networks. However, these methods require a huge amount of data to train the models. Because of the limited number of light field video datasets with corresponding ground

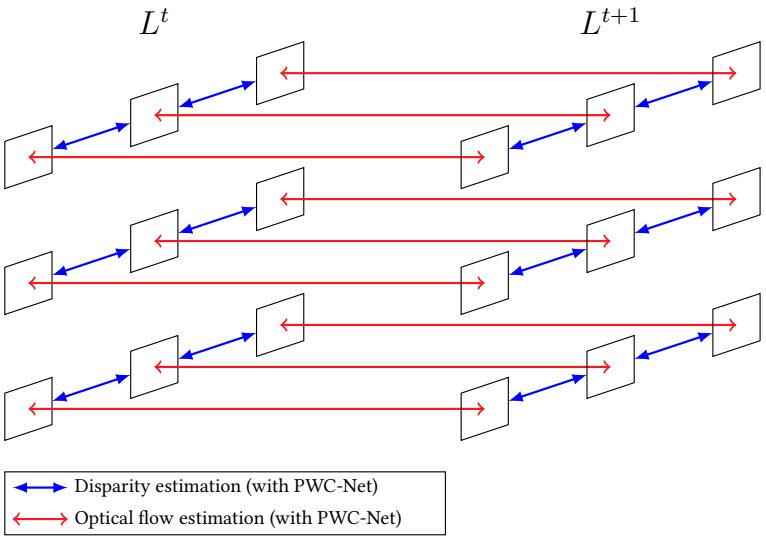


Figure 4.4: Initialization of the scene flow using a deep optical flow method (PWC-Net [80] in our case).

truth scene flows, extending deep scene flow estimation methods to light fields is very challenging.

Instead, we propose to take advantage of 2D optical flow methods and then use our model to regularize the different flows in the 4D ray space in order to compute a scene flow that would be consistent across all views. For each view of the light field, we estimate an optical flow independently. We also use the same optical flow method to estimate disparity maps at t and $t+1$. For the experiments, we consider a state-of-the-art technique based on a deep learning architecture called PWC-Net [80]. The initial disparity variation is estimated in regions where there is no temporal or angular occlusion by computing the difference of disparity along the optical flow, using the initial optical flow and disparity maps at t and $t+1$.

This approach requires to compute an occlusion mask in order to know where we can estimate reliable disparity variation. For that purpose, similarly to [53], we compute an energy value for every point $P(u,v,x,y)$ of L^t, as

$$E = E_c + \lambda_1 E_{\nabla c} + \lambda_2 E_f + \lambda_3 E_{\nabla f}. \tag{4.47}$$

The terms E_c and $E_{\nabla c}$ are respectively color and color gradient consistency terms computed between each view and the same projected view from $t+1$ to t, and are defined as

$$E_c(P) = \left\| L^{t+1}(P + \phi_{\text{init}}(P)) - L^t(P) \right\|_2, \tag{4.48}$$

$$\begin{aligned} E_{\nabla c}(P) = &\left\| \nabla_x L^{t+1}(P + \phi_{\text{init}}(P)) - \nabla_x L^t(P) \right\|_2 \\ &+ \left\| \nabla_y L^{t+1}(P + \phi_{\text{init}}(P)) - \nabla_y L^t(P) \right\|_2. \end{aligned} \tag{4.49}$$

The energy terms E_f and $E_{\nabla f}$ measure the consistency of the forward optical flow $\boldsymbol{\phi}_{\text{init}}$ and the backward optical flow $\boldsymbol{\phi}^{\text{b}}_{\text{init}}$, and are defined as

$$E_f(P) = \left\| \boldsymbol{\phi}_{\text{init}}(P) + \boldsymbol{\phi}^{\text{b}}_{\text{init}}(P + \boldsymbol{\phi}_{\text{init}}(P)) \right\|_2 , \tag{4.50}$$

$$\begin{aligned} E_{\nabla f}(P) = &\left\| \nabla_x \boldsymbol{\phi}_{\text{init}}(P) + \nabla_x \boldsymbol{\phi}^{\text{b}}_{\text{init}}(P + \boldsymbol{\phi}_{\text{init}}(P)) \right\|_2 \\ &+ \left\| \nabla_y \boldsymbol{\phi}_{\text{init}}(P) + \nabla_y \boldsymbol{\phi}^{\text{b}}_{\text{init}}(P + \boldsymbol{\phi}_{\text{init}}(P)) \right\|_2 \end{aligned} \tag{4.51}$$

From this energy value, we can compute a confidence measure C as

$$C(P) = \exp\left(-\frac{E(P)}{2\sigma_c^2} \right) \tag{4.52}$$

where σ_c controls the "width" of the distribution. Finally, in order to generate a binary mask B, we threshold the confidence map as

$$B(P) = \begin{cases} 1 & \text{if } C(P) > 0.5 \\ 0 & \text{otherwise} \end{cases} \tag{4.53}$$

For the experiments, we have chosen $\lambda_1 = 2$, $\lambda_2 = 10$, $\lambda_3 = 20$ and $\sigma_c = 0.5$.

Using this mask, we can compute an initial scene flow $F_{\text{init}}, d^t_{\text{init}}, \Delta d_{\text{init}}$ where the optical flow and the disparity map at t are completely dense and where the disparity variation is only available on non-occluded regions. We can then use our model to regularize the optical flow and disparity and to interpolate the disparity variation in occluded regions (while also regularizing it in non-occluded regions).

4.5 EVALUATION

4.5.1 *Scene Flow Datasets*

In order to be able to compute objective performance measures, we have generated a synthetic light field video dataset with the corresponding ground truth scene flow[1]. For that purpose, we have used the production files of the open source movie Sintel [105] and have modified them in the Blender 3D software [106] in order to render an array of 3x3 views. Similarly to the MPI Sintel flow dataset [76, 97], we have modified the scenes to generate not only the 'final' render, but also a 'clean' render without lighting effects, motion blur, or semi-transparent objects. Ground truth optical flow and disparity maps were also generated for each view. Since disparity variation maps could not be rendered within Blender, we have computed them using the disparity map and the optical flow. However, this process requires projecting the disparity map of a frame to the next frame using the optical flow, which results in unavailable disparity variation information in

1 http://clim.inria.fr/Datasets/SyntheticVideoLF

areas of temporal occlusion. We have processed two scenes of 3×3 views of 1024×436 pixels and 50 frames corresponding to the scenes 'Bamboo2' and 'Temple1' in [76]. The disparities (in pixels) between neighboring views are in the range $[-8, +52]$ for 'Bamboo2' and $[-22, +9]$ for 'Temple1'. We chose an angular configuration that is similar to the one of real light fields captured by rigs of cameras, such as in [32] and [33], which respectively provide 5×3 and 4×4 views. We also use the dataset of [32] to test our method on a real light field sequence: 'Bar'. Each frame is a 5×3 light field, in which each view has a spatial resolution of 1920×1080 pixels. The horizontal and vertical baselines of the camera setup are different, the ratio between the two is 0.625 and the horizontal disparity ranges from 22 to 75 pixels. Finally, we assess our method on a dense synthetic light field dataset provided by [85]. The light fields have an angular resolution of 9×9 views and their spatial resolution is either 1024×720 or 412×290, respectively referred to as 'Big' and 'Small' in the rest of the article. This configuration simulates light fields captured with plenoptic cameras as in [20] or captured with very dense camera arrays as in [30].

4.5.2 *Influence of hyperparameters*

We have various hyperparameters in our method: the most critical ones are the number of clusters K, the number of nearest neighbors N we select and the number of iterations to compute an affine model N_{iter}. The density of scene flow estimates in the light field is different for the sparse-to-dense interpolation and for the dense regularization. So, the optimal number of clusters K and of neighbors N may be different in each case.

For the experiments, we used three metrics to assess the scene flow estimations: the endpoint error for the optical flow (EPE OF), the mean absolute error for the disparity map at t (MAE d^t) and the mean absolute error for the disparity variation (MAE Δd). The latter is only computed for disoccluded pixels because there is no ground truth on occluded pixels.

In order to search for the best combination of (N, K) hyperparameters for the sparse-to-dense interpolation, we perform a grid search, using the aforementioned metrics for the whole Sintel dataset. We have tested 6 different values of $N = \{1, 2, 5, 10, 20, 40\}$ for 5 different values of $K = \{625, 1250, 2500, 5000, 10000\}$. The results of the grid search are shown in Figure 4.5.

We can observe that the errors of the optical flow, the disparity map and the disparity variation behave differently. Overall, the less clusters there are and the more neighbors we take into account, the lower the errors are. But for the disparity, there seems to be a limit where the clusters are too big. On the other hand, for the disparity variation and the optical flow, there is no such limit (although on the optical flow grid search, we can see that we have lower errors with 1250 clusters than with 625 clusters for $N > 5$). This difference of behavior is because an optical flow map is generally smoother than a disparity map. Furthermore, for the disparity variation, the sparse estimates are not estimated with direct correspondences but with chained correspondences (see Figure 4.3),

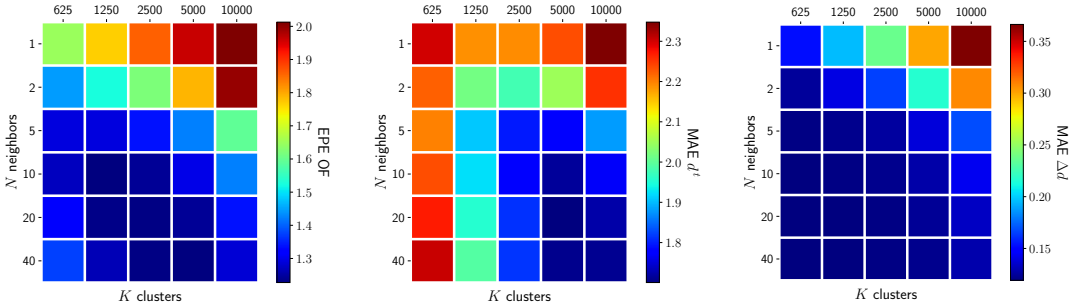


Figure 4.5: Grid search to find the optimal parameters K and N for the sparse-to-dense interpolation. Each image is an average error map for a set of (K, N) computed with the whole Sintel dataset. The combination of hyperparameters that gives the lowest errors is $K = 5000$ and $N = 40$.

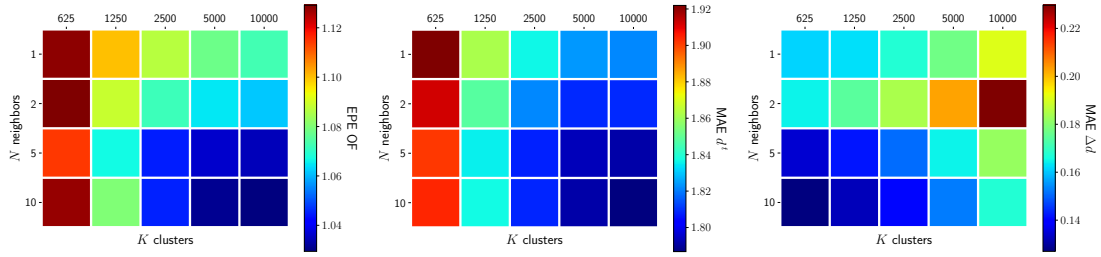


Figure 4.6: Grid search to find the optimal parameters K and N for the scene flow regularization. Each image is an average error map for a set of (K, N) computed with the whole Sintel dataset. The combination of hyperparameters that gives the lowest errors is $K = 10000$ and $N = 10$.

so the concentration of outliers is higher. As a consequence, bigger clusters with more neighbors provide lower errors. We choose to take $N = 40$ and $K = 5000$, which are the best hyperparameters in the grid search.

Similarly to the sparse-to-dense interpolation, we performed a grid search to find the combination of K and N that gives the lowest errors for the regularization. However, we limit our grid search to $N \leq 10$. The reason is that the concentration of scene flow estimates is much higher than in the sparse case, so it becomes computationally expensive to regularize the scene flow.

From Figure 4.6, we notice that the optical flow and the disparity grid search have approximately the same profile: the errors decrease when the number of neighbors N and the number of clusters K increase. We also notice that the optical flow and disparity errors increase drastically when the number of clusters is small. This is due to some underfitting of the model: there are too many estimates and our affine model is not complex enough to fit the data.

Contrary to the sparse-to-dense interpolation case, having $K = 10000$ does not increase the errors for the optical flow and disparity. This is because the concentration

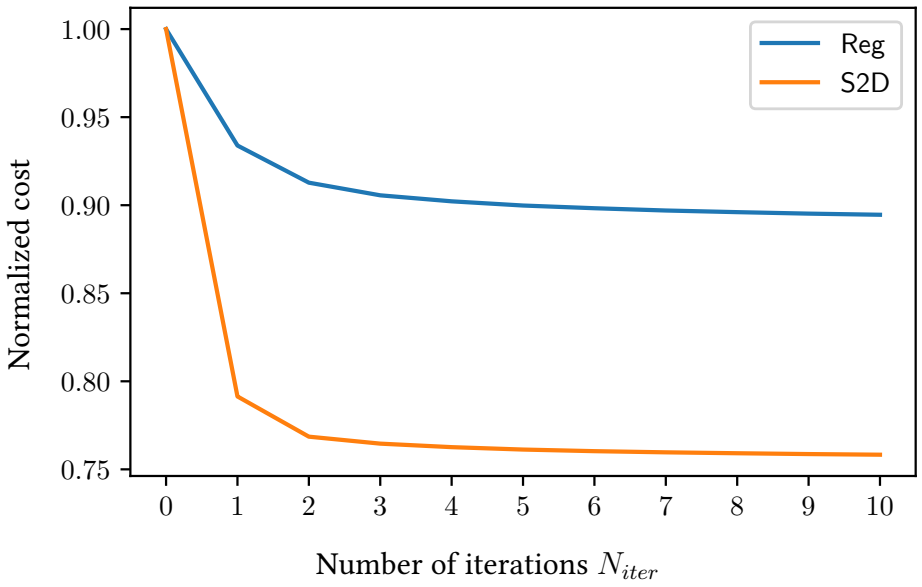


Figure 4.7: Evolution of the cost in the RANSAC model fitting for 10 iterations. The blue graph denotes the evolution of the cost in the case of dense regularization, the orange one denotes the sparse-to-dense interpolation cost. The costs are normalized by the cost of the initial model computed with Equation (4.39). In both cases, the cost becomes almost constant from the third iteration.

of the initial disparity and optical flow estimates is much higher, so the case where there is not enough scene flow estimates per cluster appears for a higher number of clusters.

On the other hand, we see that the disparity variation profile of the grid search is very similar to the one in the sparse-to-dense interpolation: the lowest error is obtained when the number of neighbors N is high and the number of clusters K is low. Again the difference with the optical flow and disparity behavior is caused by the way we compute the initial disparity variation estimates: we compute an occlusion mask to remove outliers. If there are inaccuracies in the occlusion mask, outliers will be taken into account for the model fitting. As a consequence, taking bigger clusters and more neighbors helps reducing the errors by decreasing the weight of an outlier among the estimates.

In our experiments, we choose $K = 10000$ and $N = 10$ for the scene flow regularization as it is the best combination for both optical flow and disparity estimations.

After selecting the optimal combination of N and K for the sparse-to-dense interpolation and the regularization methods, we need to determine the number of iterations N_{iter} that the model needs to converge towards a stable solution. The evolution of the cost functions (written in Equation (4.37)) of the sparse-to-dense interpolation and for the regularization are shown in Figure 4.7. They are respectively denoted 'S2D' and 'Reg' in the legend. In the figure, we normalized with the initial cost, that is the cost of the initial model as given in Equation (4.39).

Table 4.1: EPE of estimated optical flow for all pixels

		Bamboo2		Temple1	
		clean	final	clean	final
Central View	OSF[55]	1.943	1.901	6.400	4.797
	PRSM[56]	1.203	1.287	1.285	1.671
	OLFW[63]	1.421	1.462	2.061	2.374
	PWC-Net[80]	0.946	1.018	1.032	1.284
	Ours (S2D)	0.932	1.072	1.122	1.570
	Ours (Reg)	**0.883**	**0.946**	**0.959**	**1.242**
All Views	PWC-Net [80]	0.947	1.019	1.029	1.290
	Ours (S2D)	0.954	1.092	1.201	1.663
	Ours (Reg)	**0.889**	**0.952**	**0.968**	**1.253**

For both methods, the convergence rate is high and from the third iteration, the cost functions are quasi-constant. So, we have taken $N_{\text{iter}} = 3$ for the following evaluations.

4.5.3 *Comparison with state-of-the-art methods*

The proposed methods are first assessed using our sparse dataset. They are compared to the method in [63] referred to here as OLFW (Oriented Light Field Window). The OLFW method was designed for dense light fields captured with plenoptic cameras and is hardly applicable when the baseline is large. However, the optical flow searched via the oriented window can be combined with disparity maps estimated with methods suitable for sparse light fields. In the test reported here, we have used ground truth disparity maps for this method, thus showing the best results it can give for the estimated scene flow. We also compare the disparity maps that we estimate, as part of our scene flow model, with the ones obtained with the deep learning based disparity estimation method in [73], referred to here as FDE (Flexible Depth Estimation).

Besides, our methods are compared with the initial scene flow estimated as in 4.4.2, using PWC-Net [80]. The optical flow estimation technique [80] is used for separately estimating the optical flow in each view as well as the disparity between views. In order to have a dense disparity variation for this naive approach, we do not compute the occlusion mask. So, the disparity variation in occluded or disoccluded areas will never be consistent. Finally, we tested two stereo scene flow methods: [55, 56], respectively denoted as OSF (Object Scene Flow) and PRSM (Piecewise Rigid Scene Model), using the central view and its right neighbour as stereo pair.

The results are summarized in Tables 4.1, 4.2 and 4.3. For each successive light field frame of the four sequences (*Bamboo2* and *Temple1*, both rendered as *clean* and *final*), we compute the optical flow EPE, the disparity and disparity variation MAE on every ray of the light field and also on the central view only.

Table 4.2: MAE of estimated disparity for all pixels

		Bamboo2		Temple1	
		clean	final	clean	final
Central View	OSF[55]	2.578	2.611	19.307	16.990
	PRSM[56]	2.619	2.665	16.414	14.639
	FDE[73]	**1.598**	**1.663**	**0.250**	1.090
	PWC-Net[80]	1.888	1.985	0.384	0.689
	Ours (S2D)	2.043	2.041	0.680	1.115
	Ours (Reg)	1.738	1.819	0.332	**0.674**
All Views	FDE[73]	2.067	2.137	0.419	1.391
	PWC-Net [80]	1.972	2.055	0.378	0.710
	Ours (S2D)	2.445	2.418	0.753	1.220
	Ours (Reg)	**1.868**	**1.932**	**0.338**	**0.682**

Table 4.3: MAE of estimated disparity variation for all unoccluded pixels

		Bamboo2		Temple1	
		clean	final	clean	final
Central View	OSF[55]	0.539	0.518	1.491	3.159
	PRSM[56]	0.173	0.171	0.165	0.175
	OLFW[63]	0.356	0.345	0.152	0.162
	PWC-Net [80]	0.820	0.878	0.299	0.416
	Ours (S2D)	**0.113**	**0.116**	0.131	0.130
	Ours (Reg)	0.146	0.153	**0.098**	**0.116**
All Views	PWC-Net [80]	0.869	0.938	0.295	0.418
	Ours (S2D)	**0.111**	**0.115**	0.132	0.131
	Ours (Reg)	0.150	0.157	**0.105**	**0.127**

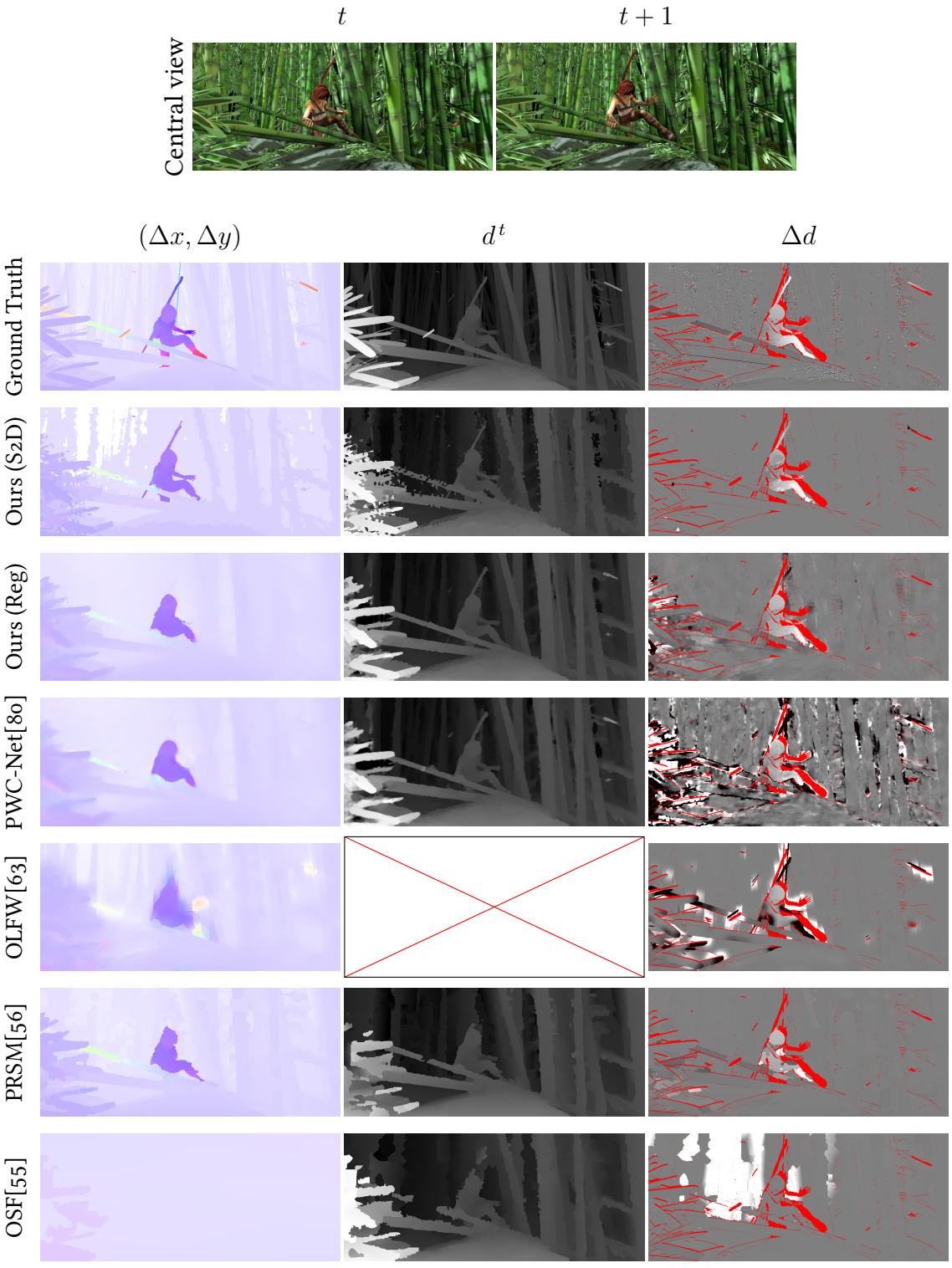


Figure 4.8: Visual comparison of our methods with [55, 56, 63, 80] on a frame of *Bamboo2 clean*. The optical flows are visualized with the Middlebury color code, and the disparity maps and disparity variations are visualized using a gray-scale representation. The red pixels are the occlusion mask where there is no ground truth disparity variation available.

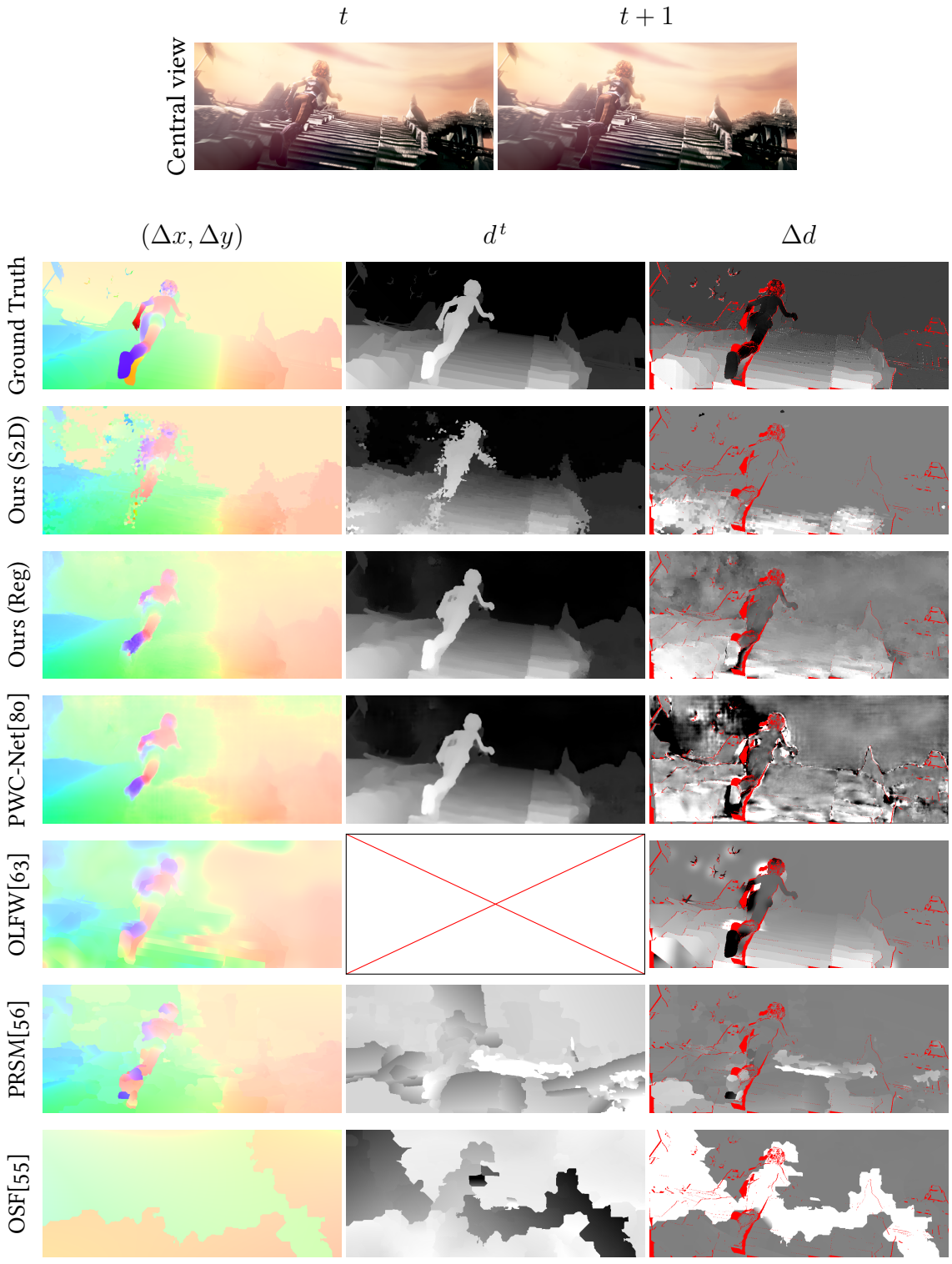


Figure 4.9: Visual comparison of our method with [55, 56, 63, 80] on a frame of *Temple1 final*. The optical flows are visualized with the Middlebury color code, and the disparity maps and disparity variations are visualized using a gray-scale representation. The red pixels are the occlusion mask where there is no ground truth disparity variation available.

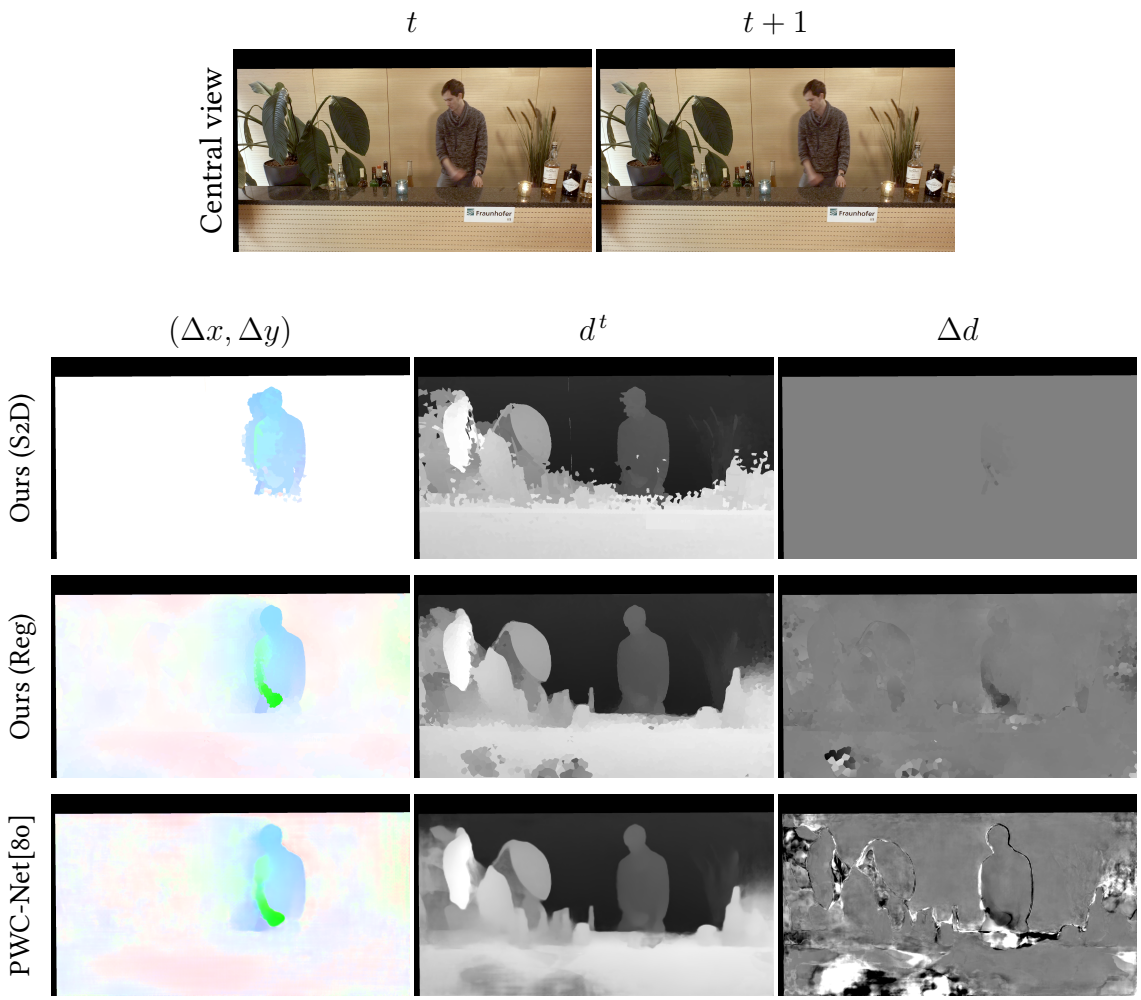


Figure 4.10: Visual comparison of our methods with [80]. The optical flows are visualized with the Middlebury color code, and the disparity maps and disparity variations are visualized using a gray-scale representation. The light field frames are taken from [32].

Table 4.4: EPE of estimated optical flow for all pixels

		NewSecretaire		Mario		Drawing		Balls		NewBalls	
		Small	Big	Small	Big	Small	Big	Small	Big	Small	Big
	PRSM[56]	1.261	1.809	1.120	1.395	1.257	3.093	0.289	0.495	0.670	0.892
	OSF[55]	0.877	1.513	2.749	6.239	-	4.355	0.744	1.613	0.547	0.794
Central	LDOF[107]	3.780	3.174	2.136	4.524	1.129	1.766	0.440	0.587	1.259	1.883
View	OLFW[63]	2.265	4.441	4.893	7.166	2.495	5.005	1.167	6.073	1.206	13.713
	FVOF [85]	0.781	1.393	**0.826**	1.012	0.928	**1.324**	**0.284**	0.481	**0.496**	0.693
	Ours (S2D)	1.229	1.079	1.769	1.097	**0.825**	1.462	0.454	**0.368**	0.680	**0.586**
	Ours (Reg)	**0.771**	**0.851**	1.065	**0.865**	1.146	1.661	0.390	0.430	0.602	0.708
All	FVOF[85]	1.337	1.853	**1.174**	1.299	1.019	**1.427**	0.397	0.555	**0.588**	0.807
Views	Ours (S2D)	1.688	1.549	2.084	1.392	**0.842**	1.504	0.477	**0.398**	0.738	**0.653**
	Ours (Reg)	**1.247**	**1.333**	1.381	**1.201**	1.158	1.677	**0.395**	0.435	0.608	0.719

We can observe that our regularization method always yields the most accurate optical flows (see Table 4.1), and disparity maps (see Table 4.2). Our method is only outperformed by the FDE method [73] for the disparity of the central view on *Bamboo2 clean & final* and *Temple1 clean*. However, even for these light fields, our method provides better average results than FDE when considering the disparity maps of all the views, which indicates a better consistency between views. Furthermore, the FDE method only estimates disparity (and not optical flow or disparity variation), while our approach computes the full scene flow. Even though, we did not choose the best combination of K and N parameters for the disparity variation, the mean absolute error is the lowest for the *Temple1* sequence and the second lowest among every tested method for the *Bamboo2* sequence (see Table 4.3). It is only outperformed by a small margin by our sparse-to-dense interpolation method. As for the optical flow and disparity estimation given this latter, the endpoint errors on the optical flow are comparable to state-of-the-art method as PWC-Net [80] but the mean absolute errors on disparity are the highest among the light field based methods. Note that the two stereo methods failed to estimate an accurate disparity in the *Temple1* sequence. These methods were mainly developed in the context of autonomous driving and their default parameters were fine-tuned for urban scenes.

We also performed some qualitative assessment of the methods on frames of *Bamboo2 clean* (Figure 4.8), *Temple1 final* (Figure 4.9) and of *Bar* (Figure 4.10). We can notice that our two methods gives sharper optical flow and disparity maps than the initial scene flow computed by [80], while correcting occlusion errors.

Finally, we tested our methods on a dense dataset provided by [85]. In order to keep the complexity low for our our methods, we estimated an initial scene flow on a set of nine views (central view, corner views and top, bottom, left and right views). Then,

Table 4.5: RMSE of estimated disparities for all pixels

		NewSecretaire		Mario		Drawing		Balls		NewBalls	
		Small	Big	Small	Big	Small	Big	Small	Big	Small	Big
	GCDL[98]	0.134	0.123	0.176	0.273	0.084	0.067	0.277	0.211	0.092	0.069
	PSDE[51]	0.350	0.136	0.543	0.092	0.115	0.119	0.595	0.111	0.148	0.077
	OADE[99]	0.193	0.138	0.196	0.165	0.074	0.068	0.245	0.111	0.172	0.079
Central	PRSM[56]	0.136	0.125	0.139	0.102	0.079	0.061	0.051	0.036	0.059	0.048
View	OSF[55]	0.131	0.120	0.216	0.103	-	0.141	0.062	0.053	0.068	0.061
	OLFW[63]	0.147	0.126	0.188	0.123	0.097	0.067	0.094	0.064	0.077	0.057
	FVOF [85]	**0.110**	**0.080**	0.136	0.073	**0.058**	**0.039**	0.068	0.036	**0.051**	0.041
	Ours (S2D)	0.223	0.151	0.248	0.137	0.117	0.089	0.115	0.077	0.098	0.075
	Ours (Reg)	0.113	0.084	**0.084**	**0.058**	0.061	0.053	**0.049**	**0.033**	0.053	**0.039**
All	FVOF [85]	0.123	0.088	0.145	0.082	**0.062**	**0.045**	0.090	0.038	0.059	0.044
Views	Ours (S2D)	0.232	0.155	0.277	0.143	0.118	0.089	0.135	0.083	0.112	0.079
	Ours (Reg)	**0.114**	**0.086**	**0.090**	**0.060**	0.064	0.054	**0.053**	**0.034**	**0.054**	**0.040**

clustering every views, we were able to fit a scene flow model, interpolate and regularize the scene flow on every view of the light field.

For the disparity estimation, some light field depth estimation methods were added to compare: globally consistent depth labeling (GCDL)[98], phase-shift based depth estimation (PSDE)[51] and occlusion-aware depth estimation (OADE)[99]. Note that the metric used to compute the disparity estimation is Root Mean Square Error (RMSE) as it was in the original paper[85]. The results are given in Tables 4.4 and 4.5. We see that, in terms of optical flow and disparity, our regularization method yields similar results to [85] for the central view, and that, in most scenes, it gives more accurate estimation when taking every view into account. Both methods outperform every other tested method based on light fields or stereo images. As for the sparse-to-dense interpolation method, it gives comparable errors as our regularization method and the method in [85] in terms of optical flow but higher errors in terms of disparity.

4.5.4 *Model validation*

In order to validate the affine model, we used the ground truth scene flow as initial estimation and then we performed the clustering step as well as the fitting of the model with the same hyperparameters as in Section 4.5.2. This gives us the minimum errors that can be obtained with our method, due to the model approximation. The results of this experiment for every scene and rendering are summarized in Table 4.6. In comparison with state-of-the-art methods, the endpoint errors for the optical flow and the mean absolute errors for the disparity that we obtain are substantially lower by a factor of 6.

Table 4.6: Validation of the affine model according to the scene

	Bamboo2		Temple1	
	clean	final	clean	final
EPE $(\Delta x, \Delta y)$	0.159	0.165	0.172	0.199
MAE d^t	0.347	0.309	0.062	0.061
MAE Δd	0.118	0.119	0.064	0.064

The mean absolute errors for the disparity variation also decrease but less significantly, because they are already very low.

Visual comparisons between the estimated scene flow and the corresponding ground truth are also presented in Figures 4.11 and 4.12 for the light field frames that have the highest endpoint errors in each scene rendered in *final* mode. We observe that, although these frames are the worst frames in their respective sequences, thin structures are well reconstructed. However, our estimated optical flow is inaccurate for objects whose disparity is so high that it disappears in other views of the light field, making it very difficult to accurately cluster the object and fit an accurate affine model. This is why the optical flow of the butterfly on the *bamboo* frame in Figure 4.11 is visually different from the ground truth. Inaccuracies are also observed when small objects have colors that are very close to the background: this leads to a weighted graph with strong edges between the aforementioned object and the background clusters. This is the case for the optical flow of the dragons on the *temple* frame in Figure 4.12.

In order to provide more insights on where our affine model fails to accurately represent the ground truth, we measured the influence of temporal occlusions, motion amplitude and disparity with the Sintel dataset. In Table 4.7, we computed errors on temporally non-occluded and occluded regions, respectively referred to as NOC and OCC. The last row is the ratio of each region for the whole dataset (e. g., there are 96% pixels that are not occluded in the Sintel dataset). We can notice that the errors are much higher in occluded areas. Since occlusions are typically located on objects boundaries, a bad clustering that groups pixels from different objects will cause errors during the model fitting step, thus giving a scene flow with more errors in the occluded areas.

In Table 4.8, the impact of motion amplitude is measured. Let $s = \sqrt{\Delta x^2 + \Delta y^2}$ be the amplitude of motion of a pixel, s10, s10-40, s40 respectively represent the regions where $s < 10$, $s \in [10, 40]$ and $s > 40$. The results show that the error of the model is larger for objects with a very large motion ($s > 40$).

Finally, we evaluate the influence of disparity on the errors in Table 4.9. Low disparity areas correspond to background objects, which tend to have lower motion amplitude than objects of the background whose disparities are higher. Therefore, high disparity and large motion are inherently related in the tested scenes, which explains why the areas with large disparity have higher errors.

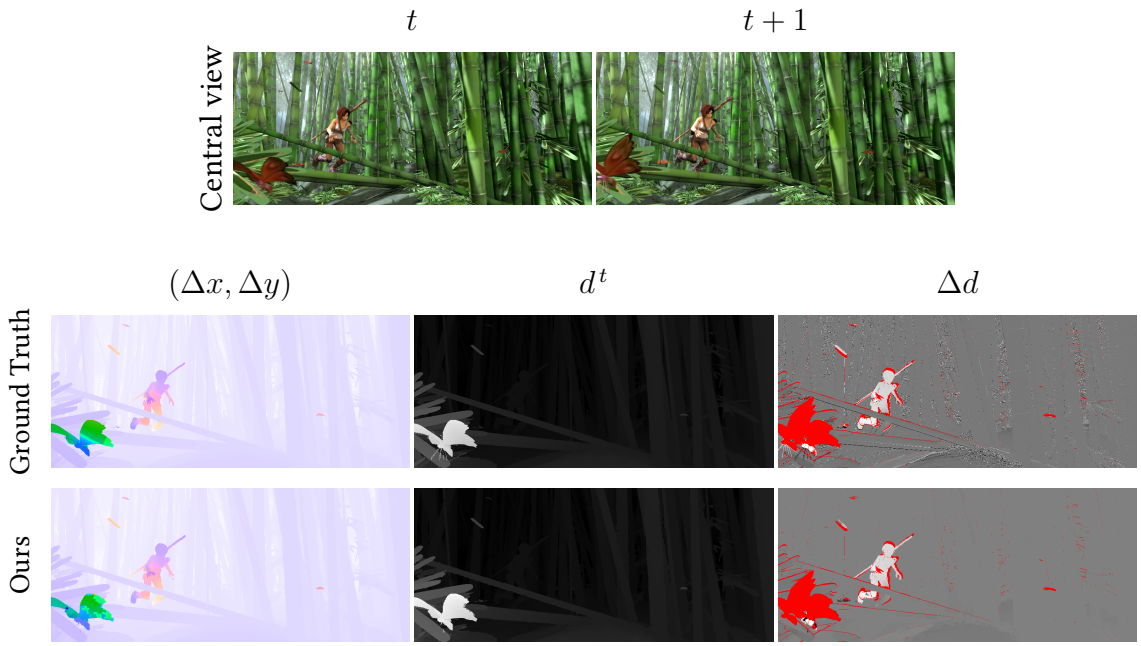


Figure 4.11: Visual comparison of the ground truth scene flow and the one obtained with our method using the ground truth scene flow as initialization, with a *Bamboo2 final* frame.

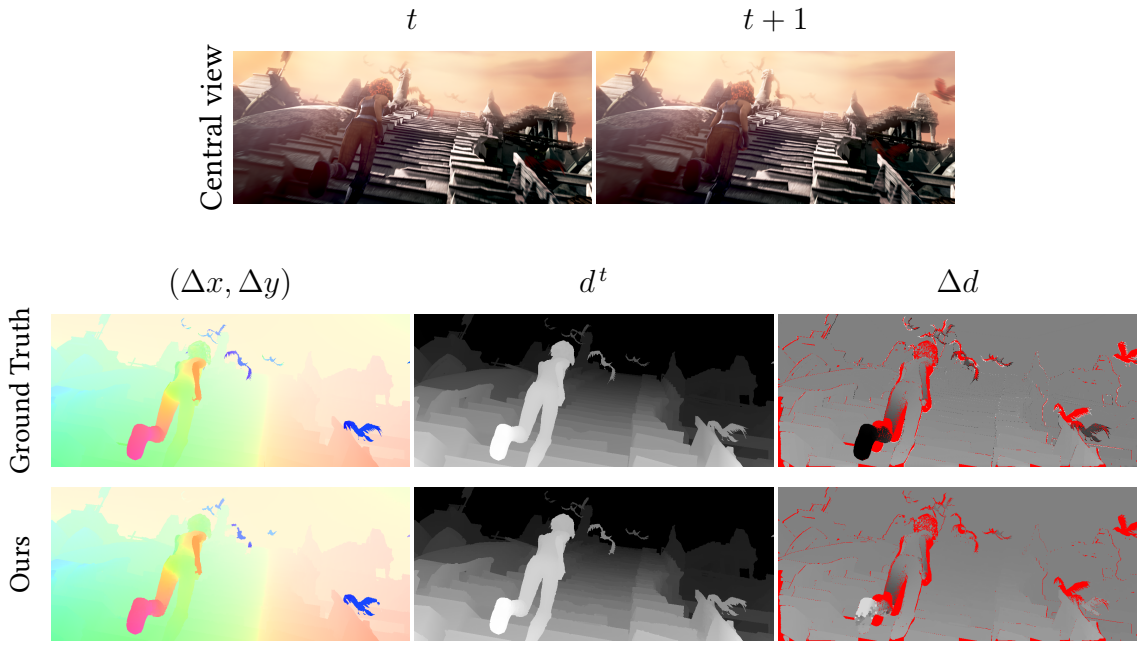


Figure 4.12: Visual comparison of the ground truth scene flow and the one obtained with our method using the ground truth scene flow as initialization, with a *Temple1 final* frame.

Table 4.7: Influence of occlusions on the affine model

	NOC	OCC
EPE $(\Delta x, \Delta y)$	0.112	1.609
MAE d^t	0.174	0.671
MAE Δd	0.091	/
Ratio (%)	96	4

Table 4.8: Influence of the motion amplitude on the affine model

	s10	s10-40	s40
EPE $(\Delta x, \Delta y)$	0.096	0.647	4.505
MAE d^t	0.202	0.084	0.475
MAE Δd	0.080	0.088	0.957
Ratio (%)	91	8	1

Table 4.9: Influence of the disparity on the affine model

	$d^t < 10$	$d^t > 10$
EPE $(\Delta x, \Delta y)$	0.146	0.351
MAE d^t	0.172	0.333
MAE Δd	0.062	0.245
Ratio (%)	86	14

4.5.5 *Complexity*

Using the optimal hyperparameters, the computation takes one hour per light field frame on average for the sparse dataset, with our laptop equipped with an *Intel Core i7 - 6600U* CPU and 16 GB RAM. Note that the aforementioned duration is calculated with a non-optimal and fully sequential implementation. However, most of the steps (i. e., scene flow initialization, clustering, nearest neighbor search, model fitting) could benefit from a parallel implementation. The authors in [100] implements their clustering on a GPU. Then, once the weighted graph is built, we can simultaneously search for the nearest neighbors of each cluster. Finally, the model fitting step can be performed independently on each cluster.

Let $M_a \times M_a$ and $M_s \times M_s$ be the angular and spatial resolutions of our light field, K be the number of clusters, N the number of neighbors, I the number of iterations and S the number of initial scene flow estimates per cluster. Then, the time complexity of fitting our model to every cluster is $\mathcal{O}(13IKN^2S)$, where 13 corresponds to the number of parameters of our model. Complexity changes quadratically with the number of neighbors N due to the propagation step added in the RANSAC algorithm. The complexity of the sparse-to-dense interpolation is lower than the regularization one because S is lower (≈ 25 times lower with our setup), which is why we can afford to use a higher number of neighbors N. In the case of regularization, where we estimate a initial scene flow on every view, we have $S = M_a^2 M_s^2 / K$ and the complexity becomes $\mathcal{O}(13IN^2M_a^2M_s^2)$. However, if we want to reduce the complexity of our regularization method, we do not need to have an initial scene flow estimate on every view. This is what we did for the dense dataset, where we took 3×3 views (instead of 9×9 views) in the initialization step. In this case, the complexity becomes $\mathcal{O}(117IN^2M_s^2)$.

4.5.6 *Limitations*

We further test our methods (with the PWC-Net[80] initialization) on dense synthetic datasets ray-traced using POV-Ray, *Apples* and *Snails* provided by [61], that have very narrow baselines. Their angular resolution is 9×9 with a respective disparity range of $[1.1, 1.7]$ and $[0.3, 1.4]$. The scenes are photo-realistic with strong specular reflections, strong shadows and non-lambertian surfaces. Therefore they are very challenging light fields. We compare the mean square errors (MSE) produced by our estimations with those obtained by the method proposed in [61] (denoted PPDA for Preconditioned Primal-Dual Algorithm). The results are summarized in Table 4.10.

We can see that our methods fail to accurately estimate the scene flow on this dataset. In the case of regularization, this failure is mostly caused by the initialization step which produces too many outliers for the fitting of the model. The method we used, i. e., PWC-Net [80], is indeed not very robust to strong specularity and does not handle ambiguous situations, e. g., when a shadow is moving, it is unclear whether the optical flow should represent the apparent motion of the shadow or the motion of the surface

Table 4.10: MSE of estimated optical flow and disparity

		Apples	Snails
MSE Δx:	PPDA[61]	0.3114	0.0996
	Ours (S2D)	0.2907	0.1446
	Ours (Reg)	0.3283	0.2652
MSE Δy:	PPDA[61]	0.0245	0.0406
	Ours (S2D)	0.0635	0.0439
	Ours (Reg)	0.0321	0.1095
MSE d^t:	PPDA[61]	0.0025	0.0036
	Ours (S2D)	6.5244	3.1071
	Ours (Reg)	0.00023	0.0043

the shadow is projected on. On the other hand, our sparse-to-dense method is more robust to these ambiguous situations but completely fails to estimate disparity. This is mostly due to the very small baseline of the light field: for disparity less than a pixel patch-based methods are very inaccurate. The method in [61] operating on epipolar plane images is on the contrary well suited for such light fields with narrow baselines but cannot be used when the disparity is large, the case we focus on in this chapter.

4.6 SUMMARY

In this chapter, we have presented a new model of scene flow that takes into account the epipolar structure of light fields. Using the developed model, we proposed two methods to estimate scene flows from light field videos. These methods are based on the three following steps: first an initial scene flow estimation, then a 4D clustering of the light field, and finally a fitting of the model for each cluster. One method use a sparse initial scene flow estimation while the other use a dense estimation. For the performance evaluation, we have generated a synthetic dataset from the open source movie Sintel in order to extend the popular MPI Sintel benchmark to sparsely sampled light fields and scene flow. We also assessed our methods on a dense light field dataset. Some qualitative tests were finally run on real light fields using the Fraunhofer dataset. For the sparse dataset, our regularization method had lower errors for the optical flow, the disparity and the disparity variations than any other state-of-the-art scene flow approaches. Our sparse-to-dense method gave higher errors on optical flow and disparity estimation while yielding more accurate disparity variation estimation on several scenes. On the dense dataset, the regularization method gave comparable performances with the state-of-the-art light field method regarding the horizontal and vertical displacements (i. e., the optical flow) and disparity of the central view while yielding more accurate results on the whole 4D light field. On the other hand, even if the sparse-to-dense interpolation gives comparable errors to the regularization approach in terms of optical flow, its disparity estimate is less accurate. That is why we would advise using the regularization

method instead to compute optical flows. Overall the regularization gives more accurate results and provides convincing results on both sparse and dense datasets. Using the ground truth scene flow as initialization, we have shown that the ground truth locally conforms to our affine model. This model is also a light way of describing a dense scene flow on the whole light field as it requires only 13 parameters per cluster. It could therefore be incorporated in a light field coding scheme as it would provide a prediction of every view of the light field at time $t + 1$, only transmitting the central view at t alongside with the scene flow parameters. At the very least, this scene flow method could be used to perform light field frame interpolations, this is what the next chapter will demonstrate.

5

ANGULARLY CONSISTENT FRAME INTERPOLATION

5.1 INTRODUCTION

Increasing the video frame rate by temporal frame interpolation has been a widely addressed problem, e. g., for compression purposes, or to create high quality videos or to produce slow motion effects. In the meantime, light fields have shown that they could be useful in numerous computer vision and image processing applications such as depth estimation, optical flow, refocusing or view interpolation. In this chapter, we focus on synthesizing a whole light field frame between two consecutive light fields frames. The synthesized light field should be consistent with the previous and following frames but it should also preserve the epipolar structure. In a nutshell: the generated light field sequence should be temporally and angularly consistent.

We choose to have a full light field approach that does not require an additional single view video with higher frame rate to interpolate the light field frames. We also take into consideration that there are very few light field video datasets and therefore a neural network that would take light fields as inputs and outputs would be difficult to train. Furthermore, such an approach might require a model with a very larger number of parameters to handle the high dimensional data represented by light field videos. Instead, we enforce the angular consistency of the light field during the optical flow estimation. Then, temporal consistency is enforced by synthesizing individually every view of the light field using a neural network trained on traditional videos and taking best advantage of the optical flows obtained previously. For the neural network architecture and model, we took inspiration from the arbitrary-time flow interpolation network of [88].

In order to validate the proposed method, we use the synthetic light field video dataset based on the Sintel movie detailed in Chapter 4. We assess our algorithm in comparison with the Super SloMo method in [88], and with two other video frame interpolation methods, Separable Adaptive Convolution [91] and Deep Voxel Flow [86], that we separately apply on each view of the light field. For each method, we estimate the PSNR and the SSIM of each interpolated view, alongside a new metric that we introduce to quantify the angular consistency of a given light field frame. Experimental results show that the proposed method gives a better angular consistency among the synthesized light field. While our PSNR and SSIM results are comparable to state-of-the-art video

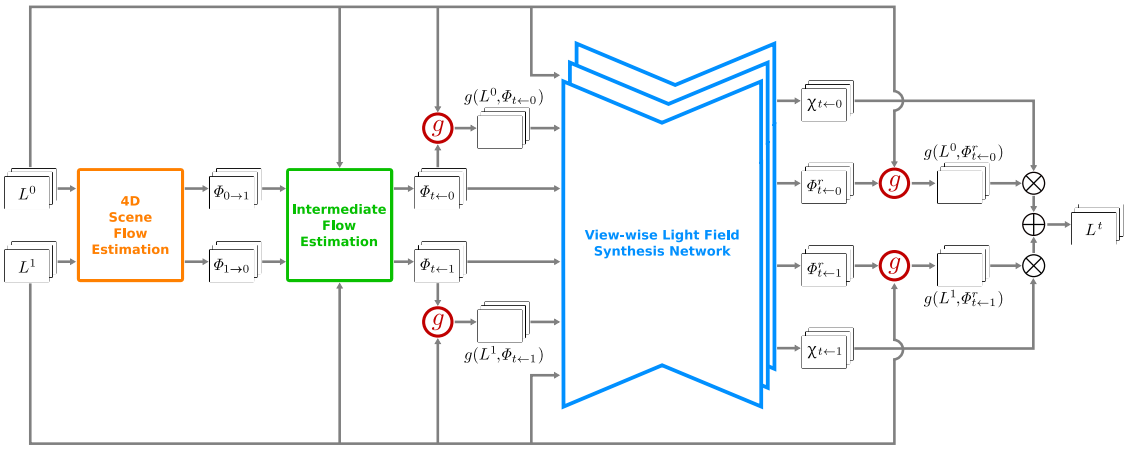


Figure 5.1: Block diagram of our interpolation method.

interpolation methods independently applied on each view, visual improvement is also observed, in particular for areas with large motion.

5.2 LIGHT FIELD FRAME INTERPOLATION METHOD

Let L^0 and L^1 be two consecutive light fields in a video. The goal of this method is to synthesize a light field L^t at intermediate time instant $t \in [0, 1]$. First, we compute an angularly consistent bidirectional optical flow between L^0 and L^1. Then, we use a convolutional neural network to refine independently the flows of each view of the light field and to synthesize the light field at an intermediate time instant t. The method is summarized in Figure 5.1.

5.2.1 *4D consistent optical flow*

The first step of the proposed light field frame interpolation method is to estimate an optical flow for every view of the light field ("4D scene flow estimation" block in Figure 5.1). In order for the interpolation to be angularly consistent, we regularize optical flows of each view with the method described in Chapter 4. The method first constructs clusters of light rays in the 4D space and then fits a 4D local affine model to each cluster taking into account the epipolar structure of the light field. We estimate a bidirectional optical flow, that is the optical flow $\boldsymbol{\phi}_{0\to1}$ from L^0 to L^1, and the optical flow $\boldsymbol{\phi}_{1\to0}$ from L^1 to L^0. The vectors $\boldsymbol{\phi}_{0\to1}$ and $\boldsymbol{\phi}_{1\to0}$ are defined as in Equation (4.6) for each 4D point $P = (u, v, x, y)$ of the light fields.

In order to interpolate the light field at frame t, we then need to compute the intermediate optical flows $\boldsymbol{\phi}_{0\to t}$ and $\boldsymbol{\phi}_{1\to t}$ ("Intermediate flow estimation" block in Figure 5.1). In [88], this step is simply performed as a weighted mean of $\boldsymbol{\phi}_{0\to1}$ and $\boldsymbol{\phi}_{1\to0}$. However, this approach does not estimate accurately the optical flow on the edges of an object and on the occluded areas. The authors used this simple estimation because it occurs

between two convolutional neural networks (the first one estimates a bidirectional optical flow and the second one performs the frame synthesis). Therefore, they need this step to be easily differentiable in order to train simultaneously their two networks. However, in our case, we do not train a network to estimate a bidirectional optical flow. As a result, we are not constrained to use a differentiable method.

We choose to use a method similar to the one described in [108]. The algorithm is the following:

1. Forward-warp the flow $\boldsymbol{\phi}_{0\to1}$ to time t to obtain $\boldsymbol{\phi}_t^0$ where:

$$\boldsymbol{\phi}_t^0(\text{round}(P + t\boldsymbol{\phi}_{0\to1}(P))) = \boldsymbol{\phi}_{0\to1}(P) \tag{5.1}$$

   The flow vectors are splatted with a splatting radius of 0.5 and when multiple flow vectors are projected to the same pixel location, we choose the vector that provides the best photo-consistency, that is the flow vector $\boldsymbol{\phi}_{0\to1}(P)$ that minimizes the projection error $|L^0(P) - L^1(P + \boldsymbol{\phi}_{0\to1}(P))|$.
2. Forward-warp the flow $\boldsymbol{\phi}_{1\to0}$ to time t to obtain $\boldsymbol{\phi}_t^1$ where:

$$\boldsymbol{\phi}_t^1(\text{round}(P + (1-t)\boldsymbol{\phi}_{1\to0}(P))) = \boldsymbol{\phi}_{1\to0}(P) \tag{5.2}$$

   We perform the same splatting operation and photo-consistency check as in the previous step
3. Merge the two flows into $\boldsymbol{\phi}_t$:

$$\boldsymbol{\phi}_t(P) = \begin{cases} +\boldsymbol{\phi}_t^0(P) & \text{if } \boldsymbol{\phi}_t^1(P) \text{ is not defined} \\ -\boldsymbol{\phi}_t^1(P) & \text{if } \boldsymbol{\phi}_t^0(P) \text{ is not defined} \\ (1-t)\boldsymbol{\phi}_t^0(P) - t\boldsymbol{\phi}_t^1(P) & \text{otherwise} \end{cases} \tag{5.3}$$

4. Inpaint the holes in $\boldsymbol{\phi}_t$ using an outside-in interpolation
5. The final intermediate flows are finally estimated as:

$$\boldsymbol{\phi}_{t\leftarrow0} = -t\boldsymbol{\phi}_t \tag{5.4}$$
$$\boldsymbol{\phi}_{t\leftarrow1} = (1-t)\boldsymbol{\phi}_t \tag{5.5}$$

5.2.2 *View-wise light field synthesis network*

Once we have estimated the intermediate flows $\boldsymbol{\phi}_{t\leftarrow0}$ and $\boldsymbol{\phi}_{t\leftarrow1}$, we could directly use them to back-warp the views of the light field from L^0 or L^1. However, this simple approach produces annoying artifacts especially around motion boundaries. To tackle this issue, similarly to [88], we use a convolutional neural network with a U-Net architecture to refine independently the flows of each view of the light field and to produce visibility maps that handle temporal occlusions ("View-wise light field synthesis network" block in Figure 5.1). Let V_{uv}^t be a view of L^t at angular position

(u, v). For each view V_{uv}^t to estimate, we want the CNN to produce refined optical flows $\boldsymbol{\phi}_{t\leftarrow 0}^r$, $\boldsymbol{\phi}_{t\leftarrow 1}^r$ and soft visibility maps $\chi_{t\leftarrow 0}$, $\chi_{t\leftarrow 1}$ such that:

$$\boldsymbol{\phi}_{t\leftarrow 0}^r = \boldsymbol{\phi}_{t\leftarrow 0}(u, v) + \Delta\boldsymbol{\phi}_{t\leftarrow 0} \tag{5.6}$$

$$\boldsymbol{\phi}_{t\leftarrow 1}^r = \boldsymbol{\phi}_{t\leftarrow 1}(u, v) + \Delta\boldsymbol{\phi}_{t\leftarrow 1} \tag{5.7}$$

$$\chi_{t\leftarrow 1} = 1 - \chi_{t\leftarrow 0} \quad \text{w.r.t.} \quad \chi_{t\leftarrow 0} \in [0, 1] \tag{5.8}$$

In practice, the neural network only gives $\chi_{t\leftarrow 0} \in [0, 1]$ and we then compute $\chi_{t\leftarrow 1}$ from it to ensure the validity of Equation 5.8. Moreover, instead of directly estimating the refined optical flows from the network, we make it generate $\Delta\boldsymbol{\phi}_{t\leftarrow 0}$ and $\Delta\boldsymbol{\phi}_{t\leftarrow 1}$, this produces better results according to [88].

Using the refined optical flows, we produce back-warped views from V_{uv}^0 and V_{uv}^1 that we merge thanks to $\chi_{t\leftarrow 0}$ and $\chi_{t\leftarrow 1}$ to form the intermediate light field view $\hat{V}_{uv}^t$:

$$\hat{V}_{uv}^t = (\, \alpha_0 \odot g(V_{uv}^0, \boldsymbol{\phi}_{t\leftarrow 0}^r) + \alpha_1 \odot g(V_{uv}^1, \boldsymbol{\phi}_{t\leftarrow 1}^r)\,) \oslash Z, \tag{5.9}$$

where $g(\cdot, \cdot)$, $\odot$ and $\oslash$ respectively denote differentiable back-warp operator, Hadamard product and pixel-wise division. α_0, α_1 and Z are defined as follows:

$$\alpha_0 = (1 - t)\chi_{t\leftarrow 0} \quad \text{and} \quad \alpha_1 = t\chi_{t\leftarrow 1} \tag{5.10}$$

$$Z = \alpha_0 + \alpha_1 \tag{5.11}$$

For the architecture of the CNN, we take the same model as in [88], that is a U-Net architecture, consisting of an encoder and a decoder with skipped connections between encoder and decoder layers of the same size. The encoder consists of 6 hierarchies which are composed of two convolutional and one Leaky ReLu (with $\alpha = 0.2$) layers. Every hierarchy except the last one ends with an average pooling layer to decrease the spatial dimension by 2. The decoder consists of 5 hierarchies that start with a bilinear upsampling layer to increase the spatial dimension by a factor of 2. It is followed by two convolutional and Leaky ReLU ($\alpha = 0.2$) layers. For each convolutional layer, the kernel size is set to 3×3.

For the training, the loss function is a linear combination of a reconstruction loss $\mathcal{L}_r$, a warping loss $\mathcal{L}_w$, a perceptual loss $\mathcal{L}_p$ and a smoothness loss $\mathcal{L}_s$. Compared to [88] we only change the smoothness term, imposing the smoothing constraint to the final optical flows. Originally, the smoothness constraint was applied on $\boldsymbol{\phi}_{0\to 1}$ and $\boldsymbol{\phi}_{1\to 0}$ since they were simultaneously estimated with another network.

$$\mathcal{L} = w_r\mathcal{L}_r + w_w\mathcal{L}_w + w_p\mathcal{L}_p + w_s\mathcal{L}_s \tag{5.12}$$

with:

$$\mathcal{L}_r = \|I^t - \hat{I}^t\|_1 \tag{5.13}$$

$$\mathcal{L}_w = \|I^t - g(I^0, \boldsymbol{\phi}_{t\leftarrow 0}^r)\|_1 + \|I^t - g(I^1, \boldsymbol{\phi}_{t\leftarrow 1}^r)\|_1 \tag{5.14}$$

$$\mathcal{L}_p = \|\boldsymbol{\psi}(I^t) - \boldsymbol{\psi}(\hat{I}^t)\|_2 \tag{5.15}$$

$$\mathcal{L}_s = \|\nabla\boldsymbol{\phi}_{t\leftarrow 0}^r\|_1 + \|\nabla\boldsymbol{\phi}_{t\leftarrow 1}^r\|_1 \tag{5.16}$$

where I^t, $\hat{I}^t$, ∇ and $\boldsymbol{\psi}$ respectively denote the ground truth intermediate frame, the synthesized frame estimated with Equation5.9, the gradient operator and the conv4_3 features of a pre-trained VGG16 model [109].

Like in [88], the weights are set to $w_r = 0.8$, $w_w = 0.4$, $w_p = 0.005$ and $w_s = 1$.

5.3 EXPERIMENTS

5.3.1 *Training*

To generate an intermediate light field frame, we choose a view-wise approach for the neural network. The reason is that finding enough light field videos to train a neural network is very challenging. Instead, having a view-wise approach enables us to use 2D videos for the training. So, in order to train our network, we use the MPI Sintel dataset [76]. We removed the *Bamboo2* and *Temple1* sequences from the training dataset since they will be used in the evaluation.

We also need initial optical flows for these 2D frames. Since the purpose of the neural network is to refine the optical flows and generate corresponding visibility maps for optimal warping, we have to use estimated optical flows as inputs. Hence, we need to estimate optical flows with similar accuracy as the ones produced by the method presented in Section 5.2.1, based on the 4D consistent optical flows of Chapter 4. Since this 4D approach cannot be applied to the 2D dataset, we use instead the PWC-Net [80] optical flow estimation network that is used for the initialization step of Chapter 4.

First, because we have the same architecture and a similar loss function as in [88], we initialize the weights of our network with those obtained by the training of [88] on the adobe240fps dataset. Then, in the training, for every clip of the MPI Sintel dataset, we use the odd frames as input frames and the even frames as targets. During the training, we perform some data augmentation on the frames such as random cropping, horizontal flip or time inversion.

5.3.2 *Evaluation*

To test our method and compare it with other state-of-the-art methods, we use Peak Signal-to-Noise Ratio (PSNR) and Structural Similarity Index (SSIM) between every view of the interpolated light field and those of the ground truth light field frame. While these metrics provide valuable insights on the quality of independently interpolated views, they fail to assess the angular consistency of the whole interpolated light field. So, we define a new metrics that we call Light Field Epipolar Consistency (LFEC).

	Bamboo2		Temple1	
	clean	final	clean	final
DVF [86]	21.98	22.05	18.92	21.05
SAC [91]	**27.15**	**27.10**	**24.66**	27.53
SSM [88]	26.05	26.07	24.12	**27.90**
Ours	26.50	26.52	24.06	27.86

Table 5.1: PSNR of synthesized light field for all views

Light Field Epipolar Consistency metric

Let $N_u \times N_v$ and $W \times H$ be the respective angular and spatial resolution of our light field. In the synthetic dataset that we use, ground truth disparity maps are provided, so we can back-warp every non-central view into the central view, taking into account the angular occlusions. We respectively denote G^t_{uv} and $\bar{G}^t$ the warped views and the view obtained when averaging every warped views. Then, for every pixel of the central view, we can compute a variance of every warped colors. This gives us a variance map σ^2:

$$\sigma^2 = \frac{1}{N_u N_v} \sum_{u,v}^{N_u,N_v} (G^t_{uv} - \bar{G}^t)^2 \tag{5.17}$$

The Light Field Epipolar Consistency LFEC of the synthesized light field is then computed similarly to a PSNR:

$$\text{LFEC} = 10 \log_{10} \left(\frac{d^2}{\overline{\sigma^2}} \right) \tag{5.18}$$

where d is the color range of the pixel values (for an 8-bit encoded light field, $d = 255$) and $\overline{\sigma^2}$ the mean of σ^2.

This metric measures the color consistency of every ray of the light field along the epipolar plane. If a ray has the wrong color but shares this color with the rest of rays along its epipolar plane, the metric will be high. Inversely, if none of the light field rays aligned on an epipolar plane has the same value, the metric will be low. Therefore, it enables us to measure the angular consistency of an interpolated light field.

PSNR, SSIM and LFEC measures

We compare our method with Super SloMo [88], Separable Adaptive Convolution [91] and Deep Voxel Flow [86], respectively denoted SSM, SAC and DVF on the dataset proposed in Chapter 4. For each of the metrics, we separately compute them on the two scenes (Bamboo2 and Temple1) rendered with two methods (final and clean). The "clean" rendering has no lighting effect or motion blur while the "final" rendering is more photorealistic.

	Bamboo2		Temple1	
	clean	final	clean	final
DVF [86]	0.686	0.694	0.643	0.756
SAC [91]	**0.928**	**0.928**	**0.893**	0.932
SSM [88]	0.904	0.905	0.886	0.933
Ours	0.908	0.910	0.890	**0.935**

Table 5.2: SSIM of synthesized light field for all views

	Bamboo2		Temple1	
	clean	final	clean	final
DVF [86]	29.33	28.98	29.24	29.41
SAC [91]	28.89	28.99	28.28	29.15
SSM [88]	29.19	29.08	29.13	29.45
Ours	**29.51**	**29.48**	**29.80**	**29.71**

Table 5.3: LFEC of synthesized light field

In terms of PSNR, our method ranks second on the Bamboo sequences and third on the Temple sequences. For these latter results, we achieve very comparable results to [88]. Our method gives a higher SSIM than [88] and [86] on every sequence and outperforms [91] on Temple1 - final. However, the PSNR and SSIM metrics are not ideal to assess the visual quality of the reconstructed views. If we look at visual comparisons for the four sequences (see Figure 5.2), we can see that our method provides more natural results for objects with large motions which are effectively propagated at the expected intermediate position. On the other hand, [91] which gives the highest SSIM and PSNR, shows blurry results in such areas. As a result, the frame interpolated with [91] is visually more similar to the average of the two input frames (i. e., overlayed input in Figure 5.2). Although our method is close to [88], our PSNR is higher. This is mostly because using our 4D approach of the scene flow estimation allows us to have more robust results than view-wise methods.

As for LFEC, our method systematically outperforms every other tested method, especially [91], which had overall better results in terms of PSNR and SSIM. On the Temple1 - clean sequence, we are even 1.5 dB ahead of [91]. If we look at the backwarp variance maps in Figure 5.3, our method gives lower variance on occluded areas than [86, 88, 91]. So, even though the method in [91] gives higher PSNR and SSIM on a few sequences, the angular consistency of the generated light fields is the lowest among every tested method. On the other hand, our method offers a good trade-off between angular and temporal consistency.

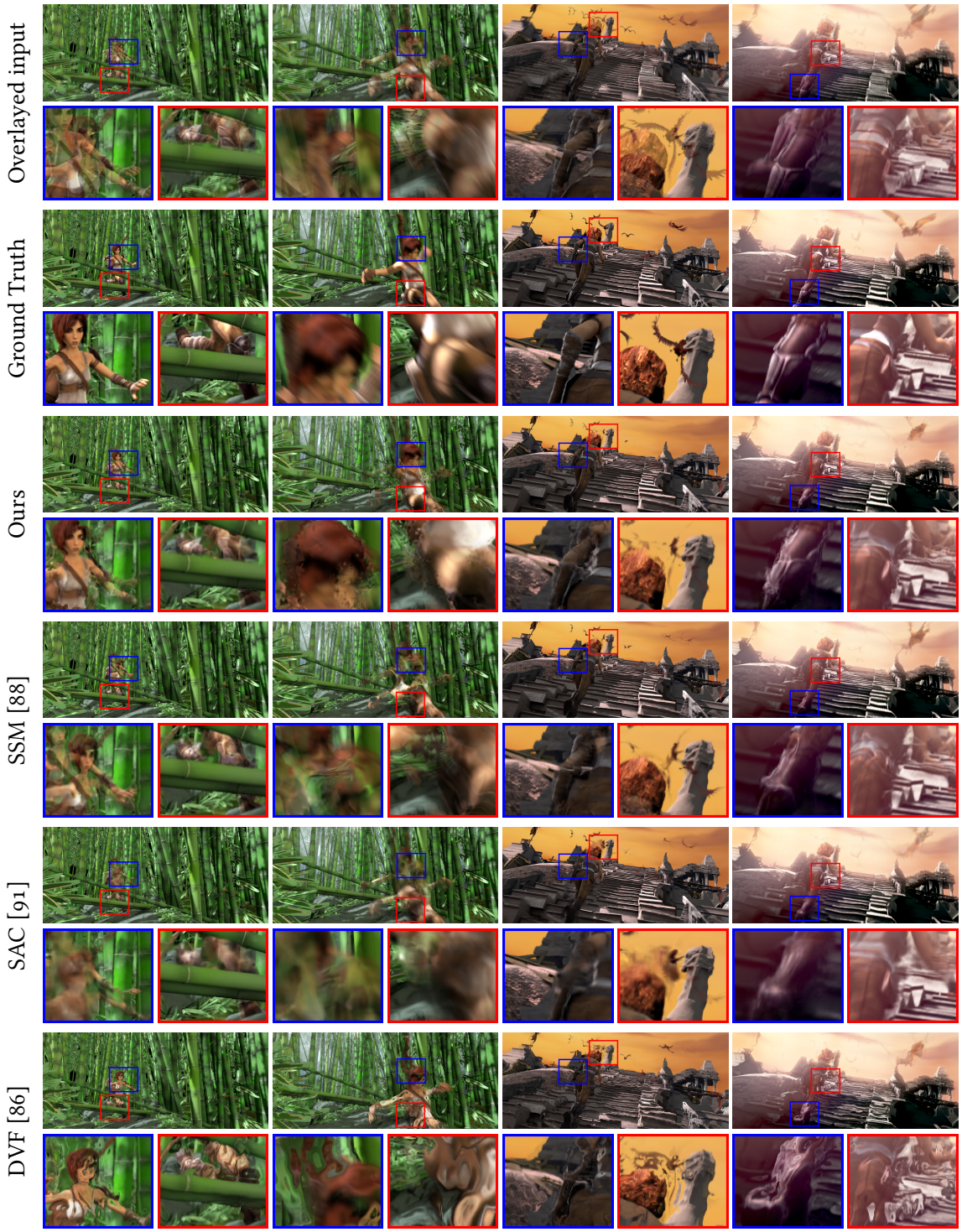


Figure 5.2: Visual comparison of our method with [86, 88, 91] for the central view.

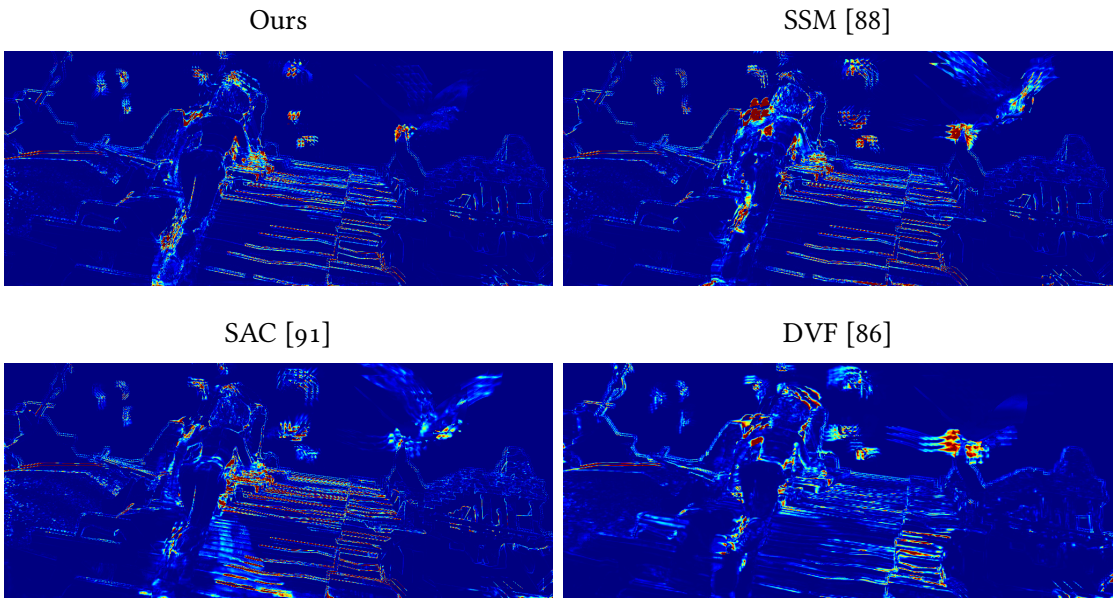


Figure 5.3: Backwarp variance map σ^2 for each method

5.3.3 *Multi-step prediction*

We can use our method to interpolate multiple light fields between two consecutive ones. The final intermediate flows described in Subsection 5.2.1 and the interpolated light field (Equation 5.9) can be computed for any value of $t \in [0, 1]$. To assess how our method performs on multi-step prediction, we use the dataset given in [96], which consists of light field videos of 8×8 views shot at only 3 fps. We generate 7 intermediate light fields between each original frame to have a final frame rate of 24 fps. There is no ground truth available for the sequence, so we visually compare our results with those given by the previous methods [86, 88, 91]. For [86, 91], it is only possible to interpolate a frame at $t = 0.5$. So, for these methods, we recursively interpolate the intermediate frames in order to have 24 fps sequences. Some visual comparisons of the central frames and epipolar plane images (EPI) are shown on Figure 5.4 for the 3rd frame (among the 7 generated) and full sequences can be found on our project webpage[1]. Our method gives the sharpest results on views and EPIs. We can observe sharp epipolar lines on our EPIs unlike those generated by [86, 91]. Furthermore, we can notice that [88] produces curved and discontinuous lines on the EPIs, proving that our method is more angularly consistent than any other tested method.

5.4 SUMMARY

In this chapter, we proposed a method to interpolate an angularly consistent intermediate light field frame from a pair of two consecutive light field frames. We divided the

1 http://clim.inria.fr/research/LFVideoInterpolation

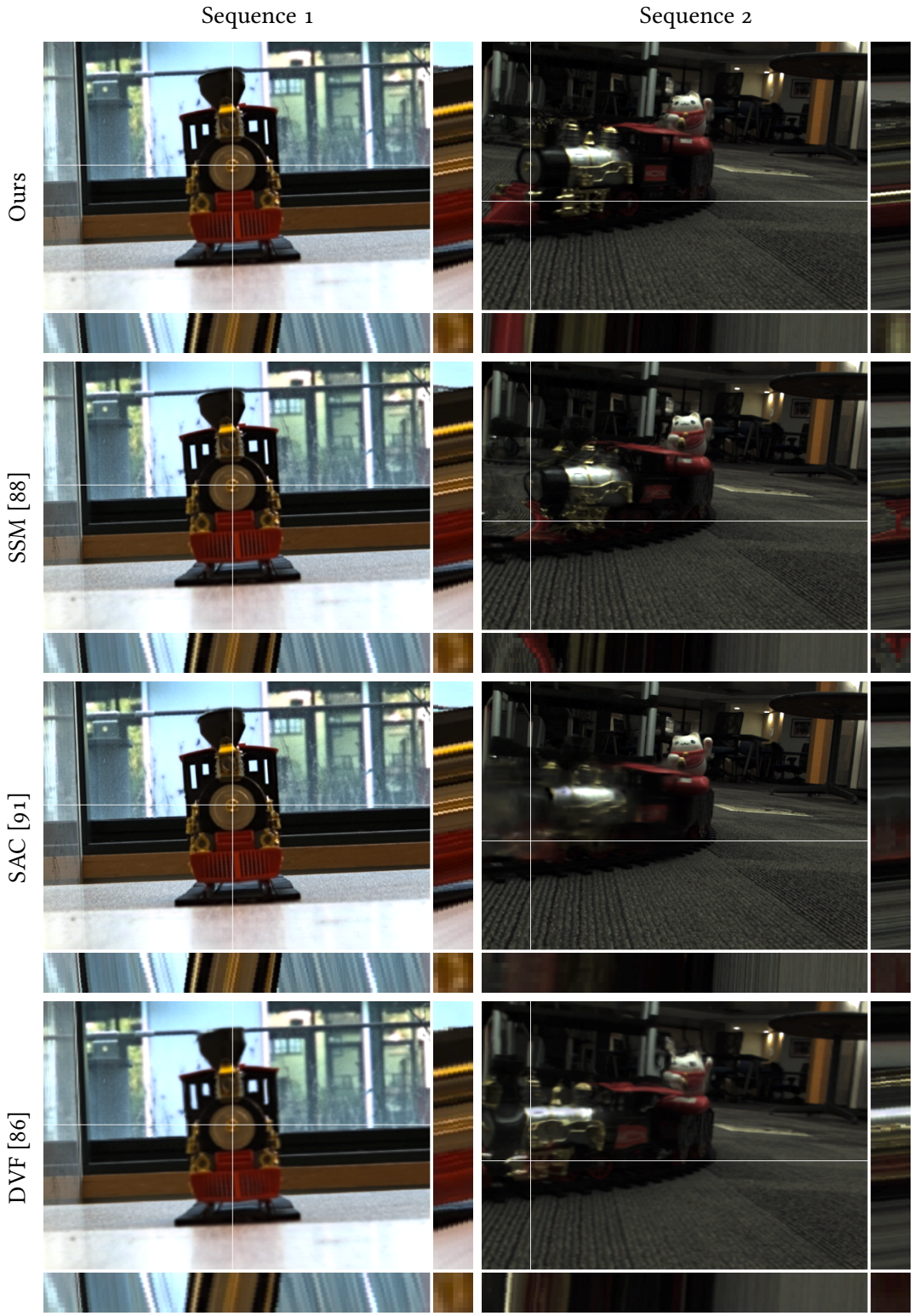


Figure 5.4: Visual comparison of our method with [86, 88, 91] for the 3rd interpolated frame of the central view.

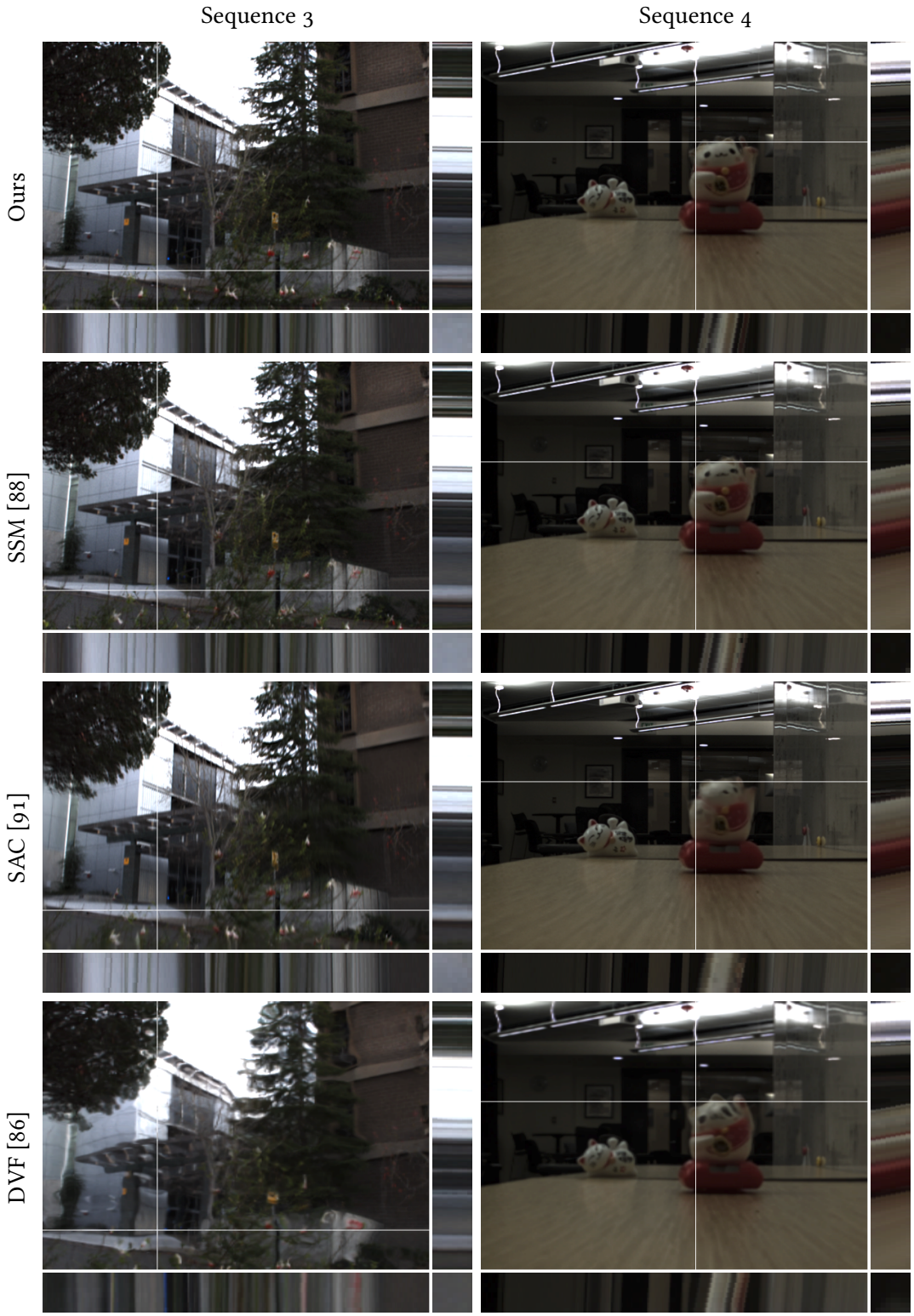


Figure 5.4: Visual comparison of our method with [86, 88, 91] for the 3rd interpolated frame of the central view.

interpolation problem in two parts: motion estimation and view synthesis. To enforce the angular consistency of our method, we estimated an optical flow on the whole 4D light field and to enforce the temporal consistency, we independently synthesized the light field views using a convolutional neural network that we trained on a synthetic dataset. To estimate the angular consistency of the reconstructed light field, we proposed a new metrics called LFEC. We compared our method with state-of-the-art deep learning approaches and achieved comparable results in terms of PSNR and SSIM. Visual inspection further indicates that our approach better keeps the temporal consistency in the case of large motions, where concurrent methods tend to produce blurred results. Furthermore, regarding the angular consistency, improved LFEC scores are obtained thanks to the use of 4D consistent optical flows, which was made possible by decoupling the optical flow estimation and the frame interpolation network.

Therefore, our results demonstrate the advantage of taking into account angular information for the temporal interpolation of light field videos. Future work in that direction would thus include a simultaneous processing of the light field views, not only for the optical flow estimation, but also for the frame interpolation.

CLOSING

GENERAL CONCLUSION

THESIS SUMMARY

Recently, light field imaging has asserted itself as a promising imaging modality, particularly in the field of Virtual and Augmented Reality. By capturing light as a collection of rays coming from different positions and directions, displaying light fields gives more immersion to the observer by adding depth cues. Light field can also be used to retrieve information about the geometry of the captured scene.

There are two types of devices used to capture light field: arrays of cameras and plenoptic cameras. Both can theoretically be used to capture still light fields and light field videos. In both cases, the captured light fields need to be extracted and put into usable form. This is challenging, especially for plenoptic cameras where information on direction and position of rays is mixed together. Plus, raw images captured with plenoptic cameras have a lenticular structure caused by the optical design of plenoptic cameras.

In Chapter 2, we analyzed one of the most used extracting pipelines for plenoptic cameras. We created raw images from synthetic light fields and used them to find the flaws of the pipeline. We showed that the lenticular structure of the raw image needed to be taken into account for the demosaicing and alignment steps or else ghost artifacts would appear on the extracted views of the light field. We developed two methods of demosaicing and alignment where the interpolations were guided by a white lenslet image. Replacing the demosaicing and alignment steps of the pipeline with our method, we showed that we were able to reduce artifacts and produce higher quality views on real light fields.

Once the light field views had been correctly extracted from a plenoptic camera or an array of cameras, information on the geometry of the captured scene could be retrieved. Classically, a depth map is estimated from still light fields. This issue has been widely investigated and a lot of different methods have been proposed. However, fewer papers had considered light field videos and the issue of motion analysis, i. e., optical flow or scene flow estimation. In Chapter 4, we proposed a model of scene flow that enforces consistency in the four dimensions of the light field. Based on this model, we explored two applications: sparse-to-dense interpolation and regularization of scene flow. The proposed model is local and affine, therefore in both cases, we perform a 4D clustering of the light field and then estimate the parameters of the model in each cluster, using initial scene flow estimates that are sparse in one case, dense in the other. In order to assess how well our methods performed against state-of-the-art methods, we generated a synthetic video dataset of sparsely-sampled light fields, using the 3D scenes of the open source movie Sintel. We also evaluated our methods on a dense

dataset and on real light fields captured with an array of cameras. We obtained the best results on light fields with wide baseline (i. e., taken with arrays of cameras). While the sparse-to-dense method provided less accurate optical flow and disparity estimations than state-of-the-art methods, it gave very accurate disparity variation maps. On the other hand, our regularization method had on average the lowest errors for optical flow, disparity and disparity estimation on our dataset and had similar results to the state-of-the-art methods on the dense dataset. Finally, we also showed that our affine model locally conforms to the ground truth and is therefore accurate to describe scene flow.

In Chapter 5, we explored how our previously estimated scene flow could be used to perform temporal interpolation on light field videos. Given two consecutive light field frames, our goal was to propose a method that could enforce not only temporal consistency but also angular consistency across the views of the generated light field. In this regard, we used our scene flow regularization method to estimate a bidirectionnal optical flow to enforce angular consistency and then we used a convolutional neural network to independently perform the view synthesis to enforce temporal consistency. In order to assess the angular consistency of the interpolated light field, we proposed a metric called Light Field Epipolar Consistency. We compared our method to state-of-the-art video interpolation methods that we had applied on the light field views. Ours results were similar to the others in terms of PSNR and SSIM but we achieved a higher epipolar consistency. Visually, our interpolated frames were also sharper.

FUTURE WORK AND PERSPECTIVES

Different approaches could be examined to extend the work presented in this thesis. We proposed a method to demosaic lenslet images and tested it in a plenoptic 1.0 extracting pipeline in Chapter 2. However our method is not exclusive to plenoptic 1.0 and it could therefore be tested for plenoptic 2.0 raw images. Additionally, our methods still require to have a white lenslet image to guide the interpolations. We could dispose of this constraint by replacing it with a model based on the grid parameters and the vignetting profile of the lenslet array. Furthermore, the weights of the demosaicing kernels were derived from the original paper and adapted using the white lenslet image. Future work could include designing optimal kernels from scratch or to use the synthetic lenslet image to learn the optimal kernel weights. One could also detect occlusions in the light field and use occlusion masks to independently demosaic different depth layers. This would potentially help extracting views with sharper edges in occluded areas.

In Chapter 4, a 4D local affine model for scene flow was presented. With only 13 parameters per cluster, this model is a light way of describing a dense scene flow on the whole light field. It could therefore be incorporated in a light field video coding scheme as it would provide a prediction of every view of the light field at time $t + 1$, only transmitting the central view at t alongside with the scene flow parameters. In the thesis, we tested our model with only one method of clustering. We could also explore other over-segmentation methods and observe if some methods are more adapted to our

model. In the presented work, we find the parameters of every cluster with a RANSAC approach and with a sampling that minimizes the condition number of the matrix to invert. We could also use other sampling methods to select the different estimates from which we generate the model. Furthermore, the model itself could also be changed. For example, instead of an affine model and an over-segmentation of the light field, we could imagine a more complex model (e. g., Gaussian mixture or learned dictionary-based models) and a semantic segmentation of the light field.

The last contribution of this thesis was a method to interpolate an intermediate light field frame from two consecutive ones. We showed how we could enforce angular consistency in the generated light field by using an angularly consistent optical flow estimation. In this perspective, future work in that direction could include a simultaneous processing of the light field views, not only for the optical flow estimation, but also for the frame interpolation. This would however be challenging as there are not so many light field video datasets to train a neural network. Moreover, in Chapter 5, we only used the optical flow components of the scene flow we previously estimated. An interesting development could be to incorporate the depth information into the frame interpolation problem. We can imagine two ways of doing so: first, we could propose a method to perform a light field frame interpolation by only using the central views of two consecutive light field frame and their corresponding scene flow. Second, we could also use depth maps as input of the view synthesis neural network. Theoretically, this would allow the neural network to better handle the temporal and angular occlusions.

BIBLIOGRAPHY

[1] Gabriel Lippmann. "Épreuves réversibles donnant la sensation du relief." In: *Journal de Physique Théorique et Appliquée* 7.1 (1908), pp. 821–825 (cited on pp. 29, 38).

[2] Herbert E. Ives. "Parallax Panoramagrams Made with a Large Diameter Lens." In: *Journal of the Optical Society of America* 20.6 (1930), pp. 332–342 (cited on p. 29).

[3] Herbert E. Ives. "Making Stereoscopic Parallax Panoramagrams from Pseudoscopic Parallax Panoramagrams." Pat. US 1,905,716. Apr. 1933 (cited on p. 29).

[4] Herbert E. Ives. "Parallax Panoramagram." Pat. US 1,918,705. July 1933 (cited on p. 29).

[5] Gerd Heymer. "Printing Pictures of a Lenticular Film." Pat. US 1,874,529. Aug. 1932 (cited on p. 29).

[6] Douglas F. Winnek Coffey. "Apparatus for Making a Composite Stereograph." Pat. US 2,063,985. Dec. 1936 (cited on p. 29).

[7] John T. Gruetzner. "Means for Obtaining Three-Dimensional Photography." Pat. US 2,724,312. Nov. 1955 (cited on p. 29).

[8] Andreī Gershun. "The Light Field." In: *Journal of Mathematics and Physics* 18.1 (1939), pp. 51–151 (cited on p. 30).

[9] Edward H. Adelson, James R. Bergen, et al. *The Plenoptic Function and the Elements of Early Vision.* Vol. 2. Vision and Modeling Group, Media Laboratory, Massachusetts Institute of Technology, 1991 (cited on pp. 30, 31).

[10] Edward H. Adelson and John Y.A. Wang. "Single Lens Stereo with a Plenoptic Camera." In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 14.2 (1992), pp. 99–106 (cited on p. 30).

[11] Marc Levoy and Pat Hanrahan. "Light Field Rendering." In: *23rd annual conference on Computer graphics and interactive techniques.* 1996, pp. 31–42 (cited on pp. 30–32, 37).

[12] Vaibhav Vaish, Bennett Wilburn, Neel Joshi, and Marc Levoy. "Using Plane+Parallax for Calibrating Dense Camera Arrays." In: *IEEE Conference on Computer Vision and Pattern Recognition.* Vol. 1. IEEE, pp. I–2 (cited on pp. 30, 42).

[13] Ren Ng, Marc Levoy, Mathieu Brédif, Gene Duval, Mark Horowitz, Pat Hanrahan, et al. "Light Field Photography with a Hand-Held Plenoptic Camera." In: *Computer Science Technical Report* 2.11 (2005), pp. 1–11 (cited on pp. 30, 38).

[14] Ren Ng. "Fourier Slice Photography." In: *ACM Transactions on Graphics* 24.3 (July 2005), pp. 735–744 (cited on pp. 30, 35).

[15] Andrew Lumsdaine and Todor Georgiev. "The Focused Plenoptic Camera." In: *IEEE International Conference on Computational Photography*. IEEE. 2009, pp. 1–8 (cited on pp. 30, 38, 43).

[16] Todor Georgeiv, Ke Colin Zheng, Brian Curless, David Salesin, Shree Nayar, and Chintan Intwala. "Spatio-Angular Resolution Tradeoffs in Integral Photography." In: *17th Eurographics Conference on Rendering Techniques*. Eurographics Association, 2006, pp. 263–272 (cited on p. 30).

[17] Leonard McMillan and Gary Bishop. "Plenoptic Modeling: An Image-Based Rendering System." In: *22nd annual conference on Computer graphics and interactive techniques*. 1995, pp. 39–46 (cited on p. 31).

[18] Steven J. Gortler, Radek Grzeszczuk, Richard Szeliski, and Michael F. Cohen. "The Lumigraph." In: *23rd annual conference on Computer graphics and interactive techniques*. 1996, pp. 43–54 (cited on pp. 31, 32).

[19] Marc Levoy. "Light Fields and Computational Imaging." In: *Computer* 39.8 (2006), pp. 46–55 (cited on p. 32).

[20] Xiaoran Jiang, Mikaël Le Pendu, Reuben A. Farrugia, and Christine Guillemot. "Light Field Compression with Homography-Based Low-Rank Approximation." In: *IEEE Journal of Selected Topics in Signal Processing* 11.7 (2017), pp. 1132–1145 (cited on pp. 34, 54, 56, 57, 83).

[21] Aaron Isaksen, Leonard McMillan, and Steven J. Gortler. "Dynamically Reparameterized Light Fields." In: *27th annual conference on Computer graphics and interactive techniques*. 2000, pp. 297–306 (cited on p. 34).

[22] Billy Chen. *Building a Projection Autostereoscopic Display*. Tech. rep. http://graphics.stanford.edu/~billyc/research/autostereo. Stanford Computer Graphics Laboratory, 2002 (cited on p. 34).

[23] Gordon Wetzstein, Douglas Lanman, Matthew Hirsch, and Ramesh Raskar. "Tensor Displays: Compressive Light Field Synthesis Using Multilayer Displays with Directional Backlighting." In: *ACM Transactions on Graphics* 31.4 (July 2012) (cited on p. 35).

[24] Fu-Chung Huang, Kevin Chen, and Gordon Wetzstein. "The Light Field Stereoscope: Immersive Computer Graphics via Factored near-Eye Light Field Displays with Focus Cues." In: *ACM Transactions on Graphics* 34.4 (July 2015) (cited on p. 35).

[25] Seungjae Lee, Changwon Jang, Seokil Moon, Jaebum Cho, and Byoungho Lee. "Additive Light Field Displays: Realization of Augmented Reality with Holographic Optical Elements." In: *ACM Transactions on Graphics* 35.4 (July 2016) (cited on p. 35).

[26] Juliet Fiss, Brian Curless, and Richard Szeliski. "Refocusing Plenoptic Images Using Depth-Adaptive Splatting." In: *IEEE International Conference on Computational Photography*. IEEE. 2014, pp. 1–9 (cited on pp. 35, 44).

[27] Sven Wanner, Stephan Meister, and Bastian Goldluecke. "Datasets and Benchmarks for Densely Sampled 4D Light Fields." In: *International Symposium on Vision, Modeling and Visualization*. Citeseer. 2013, pp. 225–226 (cited on pp. 37, 48, 50, 55).

[28] Jason C. Yang, Matthew Everett, Chris Buehler, and Leonard McMillan. "A Real-Time Distributed Light Field Camera." In: *Rendering Techniques* (2002), pp. 77–86 (cited on p. 37).

[29] Cha Zhang and Tsuhan Chen. "A Self-Reconfigurable Camera Array." In: *ACM SIGGRAPH 2004 Sketches*. 2004, p. 151 (cited on p. 37).

[30] Bennett Wilburn, Neel Joshi, Vaibhav Vaish, Eino-Ville Talvala, Emilio Antunez, Adam Barth, Andrew Adams, Mark Horowitz, and Marc Levoy. "High Performance Imaging Using Large Camera Arrays." In: *ACM SIGGRAPH 2005 Papers*. 2005, pp. 765–776 (cited on pp. 37, 42, 83).

[31] Kartik Venkataraman, Dan Lelescu, Jacques Duparré, Andrew McMahon, Gabriel Molina, Priyam Chatterjee, Robert Mullis, and Shree Nayar. "PiCam: An Ultra-Thin High Performance Monolithic Camera Array." In: *ACM Transactions on Graphics* 32.6 (Nov. 2013) (cited on p. 37).

[32] Łukasz Dąbała, Matthias Ziegler, Piotr Didyk, Frederik Zilly, Joachim Keinert, Karol Myszkowski, Hans-Peter Seidel, Przemysław Rokita, and Tobias Ritschel. "Efficient Multi-image Correspondences for On-line Light Field Video Processing." In: *Computer Graphics Forum*. Vol. 35. 7. Wiley Online Library. 2016, pp. 401–410 (cited on pp. 42, 83, 90).

[33] Neus Sabater et al. "Dataset and Pipeline for Multi-View Light-Field Video." In: *IEEE Conference on Computer Vision and Pattern Recognition Workshops*. 2017, pp. 30–40 (cited on pp. 42, 83).

[34] Manolis I. A. Lourakis and Antonis A. Argyros. "SBA: A Software Package for Generic Sparse Bundle Adjustment." In: *ACM Transactions on Mathematical Software* 36.1 (Mar. 2009) (cited on p. 42).

[35] Elijs Dima, Mårten Sjostrom, and Roger Olsson. "Assessment of Multi-Camera Calibration Algorithms for Two-Dimensional Camera Arrays Relative to Ground Truth Position and Direction." In: *2016 3DTV-Conference: The True Vision-Capture, Transmission and Display of 3D Video (3DTV-CON)*. IEEE. 2016, pp. 1–4 (cited on p. 42).

[36] Donald G. Dansereau, Oscar Pizarro, and Stefan B. Williams. "Decoding, Calibration and Rectification for Lenselet-Based Plenoptic Cameras." In: *IEEE Conference on Computer Vision and Pattern Recognition*. IEEE. 2013, pp. 1027–1034 (cited on pp. 42, 43, 45, 46, 54–57).

[37] Donghyeon Cho, Minhaeng Lee, Sunyeong Kim, and Yu-Wing Tai. "Modeling the Calibration Pipeline of the Lytro Camera for High Quality Light-Field Image Reconstruction." In: *IEEE International Conference on Computer Vision*. IEEE. 2013, pp. 3280–3287 (cited on pp. 42, 44).

[38] Shan Xu, Zhi-Liang Zhou, and Nicholas Devaney. "Multi-View Image Restoration from Plenoptic Raw Images." In: *Asian Conference on Computer Vision*. Springer. 2014, pp. 3–15 (cited on pp. 43, 45).

[39] Sven Wanner, Janis Fehr, and Bernd Jähne. "Generating EPI Representations of 4D Light Fields with a Single Lens Focused Plenoptic Camera." In: *International Symposium on Visual Computing*. Springer. 2011, pp. 90–101 (cited on p. 43).

[40] Matthieu Hog, Neus Sabater, Benoît Vandame, and Valter Drazic. "An Image Rendering Pipeline for Focused Plenoptic Cameras." In: *IEEE Transactions on Computational Imaging* 3.4 (2017), pp. 811–821 (cited on pp. 44, 63).

[41] Yongwei Li, Roger Olsson, and Mårten Sjöström. "An Analysis of Demosaicing for Plenoptic Capture Based on Ray Optics." In: *3DTV Conference*. IEEE. 2018, pp. 1–4 (cited on p. 45).

[42] Henrique S. Malvar, Li-wei He, and Ross Cutler. "High-Quality Linear Interpolation for Demosaicing of Bayer-Patterned Color Images." In: *IEEE International Conference on Acoustics, Speech, and Signal Processing*. Vol. 3. IEEE. 2004, pp. 485–488 (cited on pp. 45–48, 50, 51).

[43] Yongwei Li and Mårten Sjöström. "Depth-Assisted Demosaicing for Light Field Data in Layered Object Space." In: *IEEE International Conference on Image Processing*. IEEE. 2019, pp. 3746–3750 (cited on p. 45).

[44] Xiang Huang and Oliver Cossairt. "Dictionary Learning Based Color Demosaicing for Plenoptic Cameras." In: *IEEE Conference on Computer Vision and Pattern Recognition Workshops*. IEEE. 2014, pp. 449–454 (cited on p. 45).

[45] Mozhdeh Seifi, Neus Sabater, Valter Drazic, and Patrick Perez. "Disparity-Guided Demosaicking of Light Field Images." In: *IEEE International Conference on Image Processing*. IEEE. 2014, pp. 5482–5486 (cited on p. 45).

[46] Zhan Yu, Jingyi Yu, Andrew Lumsdaine, and Todor Georgiev. "An Analysis of Color Demosaicing in Plenoptic Cameras." In: *IEEE Conference on Computer Vision and Pattern Recognition*. IEEE. 2012, pp. 901–908 (cited on p. 45).

[47] Anat Levin and Fredo Durand. "Linear View Synthesis Using a Dimensionality Gap Light Field Prior." In: *IEEE Conference on Computer Vision and Pattern Recognition*. IEEE. 2010, pp. 1831–1838 (cited on p. 45).

[48] Pierre Matysiak, Mairéead Grogan, Mikaël Le Pendu, Martin Alain, Emin Zerman, and Aljosa Smolic. "High Quality Light Field Extraction and Post-Processing for Raw Plenoptic Data." In: *IEEE Transactions on Image Processing* 29 (2020), pp. 4188–4203 (cited on p. 54).

[49] Sven Wanner and Bastian Goldluecke. "Variational Light Field Analysis for Disparity Estimation and Super-Resolution." In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 36.3 (2013), pp. 606–619 (cited on pp. 61, 62, 66).

[50] Shuo Zhang, Hao Sheng, Chao Li, Jun Zhang, and Zhang Xiong. "Robust Depth Estimation for Light Field via Spinning Parallelogram Operator." In: *Computer Vision and Image Understanding* 145 (2016), pp. 148–159 (cited on pp. 61, 62).

[51] Hae-Gon Jeon, Jaesik Park, Gyeongmin Choe, Jinsun Park, Yunsu Bok, Yu-Wing Tai, and In So Kweon. "Accurate Depth Map Estimation from a Lenslet Light Field Camera." In: *IEEE Conference on Computer Vision and Pattern Recognition*. IEEE. 2015, pp. 1547–1555 (cited on pp. 61, 62, 70, 92).

[52] Chao-Tsung Huang. "Empirical Bayesian Light-Field Stereo Matching by Robust Pseudo Random Field Modeling." In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 41.3 (2018), pp. 552–565 (cited on pp. 61, 62).

[53] Xiaoran Jiang, Mikaël Le Pendu, and Christine Guillemot. "Depth Estimation with Occlusion Handling from a Sparse Set of Light Field Views." In: *IEEE International Conference on Image Processing*. IEEE. 2018, pp. 634–638 (cited on pp. 61, 62, 81).

[54] Sundar Vedula, Simon Baker, Peter Rander, Robert Collins, and Takeo Kanade. "Three-Dimensional Scene Flow." In: *IEEE International Conference on Computer Vision*. Vol. 2. IEEE. 1999, pp. 722–729 (cited on p. 61).

[55] Moritz Menze, Christian Heipke, and Andreas Geiger. "Object Scene Flow." In: *ISPRS Journal of Photogrammetry and Remote Sensing* 140 (2018), pp. 60–76 (cited on pp. 61, 65, 70, 86–89, 91, 92).

[56] Christoph Vogel, Konrad Schindler, and Stefan Roth. "3D Scene Flow Estimation with a Piecewise Rigid Scene Model." In: *International Journal of Computer Vision* 115.1 (2015), pp. 1–28 (cited on pp. 61, 65, 70, 86–89, 91, 92).

[57] Aseem Behl, Omid Hosseini Jafari, Siva Karthik Mustikovela, Hassan Abu Alhaija, Carsten Rother, and Andreas Geiger. "Bounding Boxes, Segmentations and Object Coordinates: How Important Is Recognition for 3D Scene Flow Estimation in Autonomous Driving Scenarios?" In: *IEEE International Conference on Computer Vision*. IEEE. 2017, pp. 2574–2583 (cited on pp. 61, 65).

[58] Deqing Sun, Erik B. Sudderth, and Hanspeter Pfister. "Layered RGBD Scene Flow Estimation." In: *IEEE Conference on Computer Vision and Pattern Recognition*. IEEE. 2015, pp. 548–556 (cited on pp. 61, 65).

[59] Julian Quiroga, Frédéric Devernay, and James Crowley. "Local/Global Scene Flow Estimation." In: *IEEE International Conference on Image Processing*. IEEE. 2013, pp. 3850–3854 (cited on pp. 61, 65).

[60] Mariano Jaimez, Christian Kerl, Javier Gonzalez-Jimenez, and Daniel Cremers. "Fast Odometry and Scene Flow from RGB-D Cameras Based on Geometric Clustering." In: *IEEE International Conference on Robotics and Automation*. IEEE. 2017, pp. 3992–3999 (cited on pp. 61, 65).

[61] Stefan Heber and Thomas Pock. "Scene Flow Estimation from Light Fields via the Preconditioned Primal-Dual Algorithm." In: *German Conference on Pattern Recognition*. Springer. 2014, pp. 3–14 (cited on pp. 61, 65, 96, 97).

[62] Julia Navarro and Juan Francisco Garamendi. "Variational Scene Flow and Occlusion Detection from a Light Field Sequence." In: *International Conference on Systems, Signals and Image Processing*. IEEE. 2016, pp. 1–4 (cited on pp. 61, 65).

[63] Pratul P. Srinivasan, Michael W. Tao, Ren Ng, and Ravi Ramamoorthi. "Oriented Light-Field Windows for Scene Flow." In: *IEEE International Conference on Computer Vision*. IEEE. 2015, pp. 3496–3504 (cited on pp. 61, 65, 66, 70, 86–89, 91, 92).

[64] Robert C. Bolles, H. Harlyn Baker, and David H. Marimont. "Epipolar-Plane Image Analysis: An Approach to Determining Structure from Motion." In: *International Journal of Computer Vision* 1.1 (1987), pp. 7–55 (cited on p. 62).

[65] Yi-Hao Kao, Chia-Kai Liang, Li-Wen Chang, and Homer H. Chen. "Depth Detection of Light Field." In: *IEEE International Conference on Acoustics, Speech and Signal Processing*. Vol. 1. 2007, pp. I–893–I–896 (cited on p. 63).

[66] Antoine Mousnier, Elif Vural, and Christine Guillemot. *Partial Light Field Tomographic Reconstruction from a Fixed-Camera Focal Stack*. Tech. rep. http://arxiv.org/abs/1503.01903. Inria, 2015 (cited on p. 63).

[67] Haiting Lin, Can Chen, Sing Bing Kang, and Jingyi Yu. "Depth Recovery from Light Field Using Focal Stack Symmetry." In: *Proceedings of the IEEE International Conference on Computer Vision*. 2015, pp. 3451–3459 (cited on p. 63).

[68] Michael W. Tao, Sunil Hadap, Jitendra Malik, and Ravi Ramamoorthi. "Depth from Combining Defocus and Correspondence Using Light-Field Cameras." In: *IEEE International Conference on Computer Vision*. IEEE. 2013, pp. 673–680 (cited on pp. 63, 65).

[69] Stefan Heber, Wei Yu, and Thomas Pock. "Neural EPI-Volume Networks for Shape from Light Field." In: *IEEE International Conference on Computer Vision*. IEEE. 2017, pp. 2252–2260 (cited on p. 63).

[70] Changha Shin, Hae-Gon Jeon, Youngjin Yoon, In So Kweon, and Seon Joo Kim. "EPINET: A Fully-Convolutional Neural Network Using Epipolar Geometry for Depth from Light Field Images." In: *IEEE Conference on Computer Vision and Pattern Recognition*. IEEE. 2018, pp. 4748–4757 (cited on p. 63).

[71] Nikolaus Mayer, Eddy Ilg, Philip Hausser, Philipp Fischer, Daniel Cremers, Alexey Dosovitskiy, and Thomas Brox. "A Large Dataset to Train Convolutional Networks for Disparity, Optical Flow, and Scene Flow Estimation." In: *IEEE Conference on Computer Vision and Pattern Recognition*. IEEE. 2016, pp. 4040–4048 (cited on p. 63).

[72] Eddy Ilg, Nikolaus Mayer, Tonmoy Saikia, Margret Keuper, Alexey Dosovitskiy, and Thomas Brox. "Flownet 2.0: Evolution of Optical Flow Estimation with Deep Networks." In: *IEEE Conference on Computer Vision and Pattern Recognition*. IEEE. 2017, pp. 2462–2470 (cited on p. 63).

[73] Jinglei Shi, Xiaoran Jiang, and Christine Guillemot. "A Framework for Learning Depth from a Flexible Subset of Dense and Sparse Light Field Views." In: *IEEE Transactions on Image Processing* 28.12 (2019), pp. 5867–5880 (cited on pp. 63, 86, 87, 91).

[74] Bruce D. Lucas and Takeo Kanade. "An Iterative Image Registration Technique with an Application to Stereo Vision." In: *International Joint Conference on Artificial Intelligence*. Vol. 2. Morgan Kaufmann Publishers Inc., 1981, pp. 674–679 (cited on pp. 63, 66).

[75] Berthold K.P. Horn and Brian G. Schunck. "Determining Optical Flow." In: *Techniques and Applications of Image Understanding*. Vol. 281. International Society for Optics and Photonics. 1981, pp. 319–331 (cited on pp. 63, 66).

[76] Daniel J. Butler, Jonas Wulff, Garrett B. Stanley, and Michael J. Black. "A Naturalistic Open Source Movie for Optical Flow Evaluation." In: *European Conference on Computer Vision*. Springer. 2012, pp. 611–625 (cited on pp. 63, 69, 82, 83, 103).

[77] Moritz Menze and Andreas Geiger. "Object Scene Flow for Autonomous Vehicles." In: *IEEE Conference on Computer Vision and Pattern Recognition*. IEEE. 2015, pp. 3061–3070 (cited on pp. 63, 65).

[78] Alexey Dosovitskiy, Philipp Fischer, Eddy Ilg, Philip Hausser, Caner Hazirbas, Vladimir Golkov, Patrick Van Der Smagt, Daniel Cremers, and Thomas Brox. "Flownet: Learning Optical Flow with Convolutional Networks." In: *IEEE International Conference on Computer Vision*. IEEE. 2015, pp. 2758–2766 (cited on p. 63).

[79] Anurag Ranjan and Michael J. Black. "Optical Flow Estimation Using a Spatial Pyramid Network." In: *IEEE Conference on Computer Vision and Pattern Recognition*. IEEE. 2017, pp. 4161–4170 (cited on p. 63).

[80] Deqing Sun, Xiaodong Yang, Ming-Yu Liu, and Jan Kautz. "PWC-Net: CNNs for Optical Flow Using Pyramid, Warping, and Cost Volume." In: *IEEE Conference on Computer Vision and Pattern Recognition*. IEEE. 2018, pp. 8934–8943 (cited on pp. 63, 70, 81, 86–91, 96, 103).

[81] Jason Pacheco, Silvia Zuffi, Michael Black, and Erik Sudderth. "Preserving Modes and Messages via Diverse Particle Selection." In: *International Conference on Machine Learning*. 2014, pp. 1152–1160 (cited on p. 65).

[82] Sizhuo Ma, Brandon M. Smith, and Mohit Gupta. "3D Scene Flow from 4D Light Field Gradients." In: *European Conference on Computer Vision*. Springer. 2018 (cited on p. 65).

[83] Michael Tao, Jiamin Bai, Pushmeet Kohli, and Sylvain Paris. "SimpleFlow: A Non-iterative, Sublinear Optical Flow Algorithm." In: *Computer graphics forum*. Vol. 31. 2pt1. Wiley Online Library. 2012, pp. 345–353 (cited on p. 65).

[84] Armin Mustafa, Marco Volino, Jean-Yves Guillemaut, and Adrian Hilton. "4D Temporally Coherent Light-Field Video." In: *IEEE International Conference on 3D Vision*. IEEE. 2017, pp. 29–37 (cited on p. 66).

[85] Hao Zhu, Xiaoming Sun, Qi Zhang, Qing Wang, Antonio Robles-Kelly, Hongdong Li, and Shaodi You. "Full View Optical Flow Estimation Leveraged from Light Field Superpixel." In: *IEEE Transactions on Computational Imaging* (2019) (cited on pp. 66, 70, 83, 91, 92).

[86] Ziwei Liu, Raymond A. Yeh, Xiaoou Tang, Yiming Liu, and Asee Agarwala. "Video Frame Synthesis Using Deep Voxel Flow." In: *IEEE International Conference on Computer Vision*. IEEE. 2017, pp. 4463–4471 (cited on pp. 66, 67, 99, 104–109).

[87] Yang-Ho Cho, Ho-Young Lee, and Du-Sik Park. "Temporal Frame Interpolation Based on Multiframe Feature Trajectory." In: *IEEE Transactions on Circuits and Systems for Video Technology* 23.12 (2013), pp. 2105–2115 (cited on p. 66).

[88] Huaizu Jiang, Deqing Sun, Varun Jampani, Ming-Hsuan Yang, Erik Learned-Miller, and Jan Kautz. "Super SloMo: High Quality Estimation of Multiple Intermediate Frames for Video Interpolation." In: *IEEE Conference on Computer Vision and Pattern Recognition*. IEEE. 2018, pp. 9000–9008 (cited on pp. 67, 99–109).

[89] Tianfan Xue, Baian Chen, Jiajun Wu, Donglai Wei, and William T. Freeman. "Video Enhancement with Task-Oriented Flow." In: *International Journal of Computer Vision* 127.8 (2019), pp. 1106–1125 (cited on p. 67).

[90] Simon Niklaus, Long Mai, and Feng Liu. "Video Frame Interpolation via Adaptive Convolution." In: *IEEE Conference on Computer Vision and Pattern Recognition*. IEEE. 2017, pp. 670–679 (cited on p. 67).

[91] Simon Niklaus, Long Mai, and Feng Liu. "Video Frame Interpolation via Adaptive Separable Convolution." In: *IEEE International Conference on Computer Vision*. IEEE. 2017, pp. 261–270 (cited on pp. 67, 99, 104–109).

[92] Wenbo Bao, Wei-Sheng Lai, Xiaoyun Zhang, Zhiyong Gao, and Ming-Hsuan Yang. "MEMC-Net: Motion Estimation and Motion Compensation Driven Neural Network for Video Interpolation and Enhancement." In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* (2019) (cited on p. 67).

[93] Wenbo Bao, Wei-Sheng Lai, Chao Ma, Xiaoyun Zhang, Zhiyong Gao, and Ming-Hsuan Yang. "Depth-Aware Video Frame Interpolation." In: *IEEE International Conference on Computer Vision*. IEEE. 2019 (cited on p. 67).

[94] Piotr Didyk, Pitchaya Sitthi-Amorn, William Freeman, Frédo Durand, and Wojciech Matusik. "Joint View Expansion and Filtering for Automultiscopic 3D Displays." In: *ACM Transactions on Graphics* 32.6 (Nov. 2013) (cited on p. 67).

[95] Simone Meyer, Oliver Wang, Henning Zimmer, Max Grosse, and Alexander Sorkine-Hornung. "Phase-Based Frame Interpolation for Video." In: *IEEE Conference on Computer Vision and Pattern Recognition*. IEEE. 2015, pp. 1410–1418 (cited on p. 67).

[96] Ting-Chun Wang, Jun-Yan Zhu, Nima Khademi Kalantari, Alexei A. Efros, and Ravi Ramamoorthi. "Light Field Video Capture Using a Learning-Based Hybrid Imaging System." In: *ACM Transactions on Graphics* 36.4 (July 2017) (cited on pp. 67, 107).

[97] Jonas Wulff, Daniel J. Butler, Garrett B. Stanley, and Michael J. Black. "Lessons and Insights from Creating a Synthetic Optical Flow Benchmark." In: *European Conference on Computer Vision*. Springer. 2012, pp. 168–177 (cited on pp. 69, 82).

[98] Sven Wanner, Christoph Straehle, and Bastian Goldluecke. "Globally Consistent Multi-Label Assignment on the Ray Space of 4D Light Fields." In: *IEEE Conference on Computer Vision and Pattern Recognition*. IEEE. 2013, pp. 1011–1018 (cited on pp. 70, 92).

[99] Ting-Chun Wang, Alexei A. Efros, and Ravi Ramamoorthi. "Occlusion-Aware Depth Estimation Using Light-Field Cameras." In: *IEEE International Conference on Computer Vision*. IEEE. 2015, pp. 3487–3495 (cited on pp. 70, 92).

[100] Matthieu Hog, Neus Sabater, and Christine Guillemot. "Superrays for Efficient Light Field Processing." In: *IEEE Journal of Selected Topics in Signal Processing* 11.7 (2017), pp. 1187–1199 (cited on pp. 74, 75, 96).

[101] Radhakrishna Achanta, Appu Shaji, Kevin Smith, Aurelien Lucchi, Pascal Fua, and Sabine Süsstrunk. "SLIC Superpixels Compared to State-of-the-Art Superpixel Methods." In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 34.11 (2012), pp. 2274–2282 (cited on p. 74).

[102] Edsger W. Dijkstra. "A Note on Two Problems in Connexion with Graphs." In: *Numerische mathematik* 1.1 (1959), pp. 269–271 (cited on p. 75).

[103] Martin A. Fischler and Robert C. Bolles. "Random Sample Consensus: A Paradigm for Model Fitting with Applications to Image Analysis and Automated Cartography." In: *Communications of the ACM* 24.6 (1981), pp. 381–395 (cited on p. 75).

[104] Dion Eustathios Olivier Tzamarias, Pinar Akyazi, and Pascal Frossard. "A Novel Method for Sampling Bandlimited Graph Signals." In: *European Signal Processing Conference*. IEEE. 2018, pp. 126–130 (cited on p. 78).

[105] Ton Roosendaal. "Sintel." In: *ACM SIGGRAPH 2011 Computer Animation Festival*. 2011, pp. 71–71 (cited on p. 82).

[106] Ton Roosendaal. *Blender - A 3D Modelling and Rendering Package*. Blender Foundation. 1998. URL: http://www.blender.org (cited on p. 82).

[107] Thomas Brox and Jitendra Malik. "Large Displacement Optical Flow: Descriptor Matching in Variational Motion Estimation." In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 33.3 (2010), pp. 500–513 (cited on p. 91).

[108] Simon Baker, Daniel Scharstein, J.P. Lewis, Stefan Roth, Michael J. Black, and Richard Szeliski. "A Database and Evaluation Methodology for Optical Flow." In: *International Journal of Computer Vision* 92.1 (2011), pp. 1–31 (cited on p. 101).

[109] Karen Simonyan and Andrew Zisserman. "Very Deep Convolutional Networks for Large-Scale Image Recognition." In: *International Conference on Learning Representations*. 2015 (cited on p. 103).

UNIVERSITE BRETAGNE LOIRE / MATHSTIC

UNIVERSITÉ DE RENNES 1

Titre : Analyse de scène et rendu de vues à partir de champs de lumière

Mot clés : champ de lumière – estimation de mouvement – interpolation temporelle

Résumé : De nouvelles modalités d'imagerie ont récemment fait leur apparition. Parmi elles, les champs de lumière sont particulièrement intéressants car ils captent des rayons lumineux individuels provenant de différentes positions et directions. Cette caractéristique permet certains usages comme l'estimation de profondeur. Pour acquérir des champs de lumière, deux types d'appareils existent : les matrices de caméras et les caméras plénoptiques. Ces dernières nécessitent un traitement post-acquisition lourd pour récupérer un champ de lumière utilisable. Récemment, certains appareils ont même permis de capturer des vidéos de champs de lumière, ouvrant la voie à l'estimation de mouvement et l'interpolation temporelle. Le premier objectif de cette thèse est d'extraire des vues de champs de lumière de haute qualité à partir de caméras plénoptiques. Dans ce contexte, nous proposons de nouvelles méthodes de démosaïquage et d'interpolation qui prennent en compte la structure lenticulaire de l'image brute. Le second objectif du travail présenté est d'explorer de nouvelles approches pour l'analyse de scène, à savoir l'estimation du mouvement. Cela nous amène à concevoir un modèle pour représenter le flux de scène (ou mouvement apparent d'une scène). Couplé à des techniques d'apprentissage profond, nous pouvons alors effectuer un rendu temporel de vues du champ du lumière, angulairement et temporellement cohérent, ce qui constitue notre dernier objectif.

Title: Scene Analysis and View Rendering from Light Fields

Keywords: Light Field – Motion Estimation – Temporal Interpolation

Abstract: New imaging modalities have recently emerged and among them, light field imaging is especially compelling for virtual and augmented reality as they allow to capture individual light rays coming from different positions and directions. This characteristic enables various functionalities including depth estimation or digital refocusing. To acquire light fields, two types of devices exist: arrays of cameras and plenoptic cameras. The latter demands heavy post-capture processing to retrieve a usable light field from the raw data. Recently, some devices have been able to capture light field videos, thus adding a temporal dimension. This new feature allows new applications for light fields: motion estimation and temporal interpolation. The first aim of this thesis is to extract high quality light field views from plenoptic cameras. In this context, we propose new methods of demosaicing and interpolations which take into account the lenslet structure of the raw image. The second aim of the presented work is to explore new approaches for scene analysis, namely motion estimation. This leads us to design a model to represent scene flow which describes the apparent 3D motion of a scene. This model is used in the context of sparse-to-dense interpolation and dense regularization. Coupled with deep-learning techniques, we are also able to perform temporal light field view rendering that is angularly and temporally consistent.