Learning-based depth estimation from light field and view synthesis

Jinglei Shi

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 : *Informatique*

Par
Jinglei SHI

Learning-based depth estimation from light field and view synthesis

Thèse présentée et soutenue à Rennes, le 16 juin 2021
INRIA Rennes - Bretagne Atlantique

Rapporteurs avant soutenance :

Pascal MONASSE	Professor, Ecole des Ponts ParisTech
Marten SJOSTROM	Professor, MidSweden University

Composition du Jury :

Examinatrice :	Luce MORIN	Professor, INSA Rennes
Examinateurs :	Pascal MONASSE	Professor, Ecole des Ponts ParisTech
	Marten SJOSTROM	Professor, MidSweden University
	Atanas GOTCHEV	Professor, Tampere University
	Xiaoran JIANG	Researcher, INRIA Rennes
Dir. de thèse :	Christine GUILLEMOT	Research Director, INRIA Rennes

ACKNOWLEDGEMENT

I would like to thank all of you who helped me to write this thesis. This work could not have been finished without your support and assistance.

I'll firstly express my deepest gratitude to my thesis supervisor, Prof. Christine GUILLEMOT, for offering me the opportunity to work on this thesis topic, and for the inspiring discussions that we've had during these past few years. Her patience and effort with me have played a key role in the completion of this manuscript. Furthermore, her rigorous attitude towards academia and her passion for research have set a good example for my future research career.

Secondly, I would like to thank my dissertation board members, Prof. Luce MORIN, Prof. Pascal MONASSE, Prof. Marten SJOSTROM and Prof. Atanas GOTCHEV. It is based on your suggestions and comments that my thesis could be improved and finally finished. Thank you very much your effort and time.

I would like to give special thanks to my colleague Xiaoran Jiang. For me, he has not been merely an excellent collaborator during my PhD studies, but also a good friend in my daily life. I sincerely thank him, both for his guidance towards my research advancement and his encouragement when I reached a plateau.

Many thanks to the best teammates of Sirocco in this world. Navid, Pierre A., Nam, Mai, Fatma, Mira, Pierre D., Elian and Simon. Your warm welcome and interesting conversations made my every day in the team colorful and joyful. Thank you Maja, I really enjoyed our tea time in the office, and thanks for helping me with my English.

Finally, I would like to thank all the people who love me and whom I love: my parents, my friends, those who accompany me on my life journey and give me their endless love, unconditional support and selfless kindness.

Table of Contents

Résumé en français

Contexte

Les images 2D classiques, qui ont jusqu'à présent été largement utilisées en photographie et en vidéo, sont une représentation plate d'une scène 3D, de laquelle il est difficile d'extraire des informations de profondeur pour la reconstruction 3D de la scène. Ce manque d'informations sur la géométrie rend les photographies 2D inadaptées aux applications qui nécessitent une reconstruction réaliste de la géométrie 3D ou un sentiment d'immersion dans une scène 3D. Par conséquent, de nouveaux types d'imagerie capables de sauvegarder ou de reconstruire facilement les informations de géométrie sont maintenant étudiées plus attentivement, par exemple: les images RGBD utilisées dans les automobiles et les drones, les images stéréo utilisées en réalité augmentée et en réalité virtuelle, ainsi que les images produites par les caméras commerciales de nouvelle génération à champ de lumière. Les caméras à champ de lumière capturent une scène à partir de différentes perspectives, produisant ainsi une grille d'images qui représente la scène telle qu'elle peut être observée à partir de différentes positions. Par conséquent, à partir d'une seule exposition, il est possible de récupérer suffisamment d'informations pour reconstruire au moins partiellement la géométrie de la scène, en se basant sur les champs de lumière capturés. Cette propriété a fait des champs de lumière une technologie d'imagerie prometteuse, qui a suscité beaucoup d'intérêt au cours de la dernière décennie.

Outre l'imagerie 3D, une autre tendance apparue au cours des deux dernières décennies est le développement d'algorithmes guidés par les données. Cette évolution est très largement due aux progrès significatifs effectués dans le développement des cartes graphiques (GPU en anglais). La nouvelle technologie GPU est suffisamment puissante pour prendre en charge des calculs intensifs en très peu de temps. Par exemple, l'Intel Pentium sorti en 2000 peut effectuer 7 Giga de FLOPS, tandis que le dernier GPU Nvidia GeForce RTX 3080 sorti en 2020 peut manipuler 29,8 Tera de FLOPS. De plus, des systèmes GPU distribués à grande échelle peuvent être utilisés afin de tirer parti de leurs capacités de traitement conjointes. Grâce à ces deux facteurs, il est désormais possible pour les algorithmes de données de traiter d'énormes quantités de données et d'apprendre des

modèles complexes, ce qui est difficile voire impossible pour les méthodes classiques. Grâce à l'augmentation de la puissance de calcul, on assiste également à des évolutions significatives pour ces algorithmes d'apprentissage : initialement de simples régression linéaires et machine à vecteur de support, les méthodes modernes sont aujourd'hui devenues beaucoup plus intensives en calcul, comme le sont par exemple les méthodes d'apprentissage profond. Il a également été démontré que cet apprentissage profond est capable de traiter efficacement un large éventail de problèmes de vision par ordinateur classiques, tels que la super-résolution d'image, la classification, la détection des contours, la segmentation, etc., ce qui a donné lieu à une série d'innovations importantes pour des applications telles que la détection des piétons en temps réel, la voiture autonome, le transfert de style de peinture, ou encore la synthèse de visage, etc.

Il faut noter que les développements de l'imagerie 3D et de l'apprentissage profond ne sont pas déconnectés. Au contraire, de nombreuses percées récentes dans le domaine de l'imagerie 3D ont été réalisées à l'aide de méthodes basées sur l'apprentissage profond. L'imagerie par champs de lumière fait partie des modalités d'image pouvant bénéficier des avancées récentes en apprentissage profond.

Motivations et objectifs

Un champ de lumière peut être décrit comme une collection de rayons lumineux émis par une scène 3D et vus à partir de n'importe quel point de coordonnées (x, y, z), selon différentes orientations, pour différentes longueurs d'onde et à différents instants. Le champ de lumière peut ainsi être défini par une fonction de rayonnement 7D qui peut, sous certaines hypothèses détaillées dans le manuscrit, être simplifiée en une fonction 4D avec deux ensembles de coordonnées : les coordonnées angulaires qui peuvent être vues comme des points de vue différents, et les coordonnées spatiales des points sur la scène. Alors qu'un échantillonnage angulaire fin du champ de lumière peut nécessiter l'enregistrement d'un nombre infiniment élevé de directions ou de points de vue, en pratique le nombre de points de vue est limité et dépend du dispositif de capture. Le concept de champs de lumière est donc assez large et englobe des vues structurées sur un plan avec une disparité constante entre les vues, ainsi que des captures non structurées (par exemple, des captures effectuées par un téléphone portable avec un déplacement sans contrainte dans la scène). Dans cette thèse, nous nous concentrons sur les champs de lumière structurés, qui sont une collection d'images capturées à partir d'un certain nombre de points de vue structurés

(a) Champ de lumière représenté sous forme d'un tableau d'images 8×8

(b) Champ de lumière représenté sous forme d'image microlentilles

Figure 1 – Champ de lumière représenté sous forme de (a) Tableau d'image. (b) image microlentille. Le champ de lumière utilisé vient de [1]

.

(normalement plus de 2 points de vue), où deux vues adjacentes horizontalement et verticalement ont toujours une baseline égale. Afin d'apporter une meilleure compréhension de la notion de champ de lumière, nous montrons Fig. 1(a) un champ de lumière représenté sous la forme d'un tableau d'images, chaque image étant obeservée de points de vue différents. Dans la Fig. 1(b), nous montrons également la représentation microlentilles d'un champ de lumière, qui peut être considéré comme des données brutes capturées par le capteur de la caméra.

Les observations à partir de plusieurs positions de visualisation offrent suffisamment d'informations pour déduire la géométrie de la scène. Néanmoins, cela signifie que le champ de lumière est une modalité d'image riche en données, peu optimisée pour le stockage et la transmission, et elle contient naturellement beaucoup de redondances. L'utilisation en pratique des champs de lumière constitue donc un défi intéressant et important. D'une part, nous souhaitons reconstruire la géométrie de la scène 3D aussi précisément que possible à partir d'un champ de lumière capturé; d'autre part, nous espérons que cette reconstruction pourra utiliser le moins de vues de champ de lumière possible. Nous posons donc la question:

1.Comment estimer précisément et efficacement une carte de profondeur en utilisant un petit sous-ensemble de vues dans un champ de lumière?

Pour répondre à cette question, nous proposons un algorithme d'estimation de disparité prenant en entrée un ensemble parcimonieux de vues de champs de lumière. Les techniques d'estimation basées sur les EPIs et les "focal stacks" n'étant disponibles que

pour des champs de lumière densément échantillonnés, nous décidons d'utiliser une technique d'estimation par correspondance, couramment utilisée pour l'estimation de disparité dans les images stéréo. Cependant, appliquer naïvement des méthodes d'estimation de disparité pensées pour des images stéréo sur des données de champs de lumière ne peut que produire des estimations de faible qualité, avec des valeurs de disparité imprécises dans les régions occultées. Puisqu'elles ne font usage que de deux vues, les informations présentes dans les autres vues considérées ne sont pas exploitées durant l'estimation. Dans cette thèse, nous cherchons à étendre la technique de correspondance en stéréo pour des champs de lumière, afin de produire des prédictions de disparité à haute qualité avec moins d'erreurs dans les régions occultés, en se basant sur plusieurs vues. En plus de cela, nous souhaitons construire notre algorithme en utilisant des techniques d'apprentissage profond, afin d'avoir un temps d'exécution nettement moins important que les méthodes classiques.

Le deuxième problème abordé dans notre recherche peut être considéré comme une extension du problème mentionné précédemment:

2.Pouvons-nous récupérer un champ de lumière entier à partir d'un petit sous-ensemble de vues et les valeurs de profondeur estimées correspondantes?

Pour répondre à cette question, nous présentons un algorithme de synthèse de vues pour les champs de lumière. Concevoir en détail une méthode de synthèse pose plusieurs problèmes, qui se doivent d'être traités. 1) Bien qu'un algorithme d'estimation de profondeur ait été déjà proposé pour répondre à la 1ère question, il est important de réduire encore davantage la taille du modèle, et de s'assurer de sa différentiabilité avant de l'intégrer dans un modèle de synthèse de vues. 2) Les nouvelles vues sont synthétisées en explorant les informations présentes dans les vues d'entrée, il arrive qu'à cause d'imprécisions dans l'estimation de disparité, des pixels issus de différentes vues ne soient pas bien alignés, donnant lieu à des régions floues dans les nouvelles vues. En particulier, quand la scène comporte de nombreuses structures fines, et de nombreuses textures, ce flou peut dégrader significativement la qualité de synthèse finale. L'algorithme conçu doit pouvoir traiter ce problème de flou issu du mauvais alignement de pixels, et mieux reconstruire les structures fines, ainsi que les textures, dans les vues synthétisées. 3) Nous aimerions que l'algorithme soit en mesure de reconstruire les champs de lumière pour une résolution angulaire arbitraire.

Le deuxième travail dans cette thèse pose le problème de la synthèse de vues pour les champs de lumières, et vise à répondre à ce trois difficultés, et nous croyons que

cette méthode de synthèse de vues s'avèrera utile pour toute une série d'applications. En effet, premièrement, un procédé de reconstruction qui n'utilise qu'un petit sous-ensemble de vues pour la reconstruction de scène allégerait le stockage et la transmission que les champs de lumière denses imposent généralement aux systèmes de transmission et d'acquisition. Deuxièmement, certains champs de lumière capturés présentent une grande disparité entre les vues adjacentes, ce qui conduit à des discontinuités perceptibles lors du changement de point de vue. De tels champs de lumière sont difficilement utilisables pour des applications immersives. Pour de telles applications, la reconstruction serait nécessaire pour générer des perspectives plus denses ou pour synthétiser certaines perspectives qui n'étaient pas disponibles auparavant (par exemple, des perspectives situées en dehors de la grille de points de vue capturée). Un tel contexte fournit une motivation dans notre travail pour développer un algorithme qui peut nous permettre de récupérer des champs de lumière entiers à partir d'un échantillon clairsemé des vues capturées.

On fait l'observation que les séquences vidéo comme les séquences de champs de lumière peuvent être considérées comme des échantillonnages d'une scène, l'une dans la dimension angulaire et l'autre dans la dimension temporelle, et que des vues de champs de lumière peuvent mener à des discontinuités en naviguant au sein de la scène. De même, des séquences vidéo sous-échantillonnées peuvent mener à des mouvements saccadés. Face à cela, nous étudions si la méthode de synthèse de vues proposée comme réponse à la 2e question pour suréchantillonner la résolution angulaire du champ de lumière peut être étendue encore davantage pour suréchantillonner une séquence vidéo. Ceci introduit le 3e point de discussion de notre thèse.

3. Une architecture de synthèse de vue pour un champ de lumière peut-elle être généralisée pour résoudre un problème de synthèse de vue temporelle?

De la même manière qu'une méthode de synthèse de vues en champs de lumière peut être utile pour toute une série d'applications, un système d'interpolation vidéo est également utile dans de nombreuses situations. Par exemple, en utilisant une méthode d'interpolation vidéo, il est possible de générer des vidéos à grand nombre d'images par seconde, ce qui est positif pour la perception humaine. Cela peut également s'avérer utile pour des tâches de compression vidéo, où il est possible de stocker et transmettre les images clé d'une vidéo, puis de reconstruire la séquence vidéo dans son intégralité en interpolant les images intermédiaires. Ces applications importantes nous ont motivé à nous pencher sur la question de l'interpolation vidéo.

Concevoir un algorithme générique, applicable pour de la synthèse de vues en champ

de lumière et l'interpolation vidéo, n'est pas une tâche triviale. En effet, nous souhaiterions que l'architecture du réseau change aussi peu que possible quand la tâche à traiter est modifiée. D'un autre côté, l'algorithme proposé devrait être en mesure de produire des résultats de qualité élevée pour les deux tâches, et ce même si les vues en champ de lumière et les séquences vidéo ont des propriétés différentes. Il est important d'aboutir à une méthode capable de traiter efficacement ces deux types de données. De plus, nous avons identifié, dans notre système de synthèse de vues, des choix de conception proposés pour la 2e question qui peuvent limiter l'efficacité, pour l'entraînement comme pour le test. Dans ce travail, nous nous intéressons également à ces composants sous-optimaux, et cherchons à améliorer la performance globale en réalisant des nouveaux choix d'architecture.

En résumé, la réponse à chaque question posée jusqu'ici implique un problème beaucoup étudié (estimation de disparité, synthèse de vue et interpolation vidéo) dans le domaine de l'imagerie en champ de lumière ou du traitement vidéo. L'objectif global de notre travail est de construire une série d'algorithmes répondant à ces questions. Ces algorithmes peuvent être soit appliqués uniquement pour résoudre les problèmes susmentionnés, ou servir de sous-modules pour résoudre d'autres problèmes liés aux champs de lumière ou aux vidéos, par exemple, les méthodes pour le synthèse de vue de champ de lumière peuvent être intégrés dans un pipeline de compression. De plus, nous essayons de garder nos méthodes proposées suffisamment générales pour pouvoir s'adapter aux données de champ de lumière synthétiques et réelles avec une large gamme de disparité, voire applicables aux données vidéo. Grâce aux techniques d'apprentissage profond, nos méthodes basées sur l'apprentissage sont applicables pour certains usages industriels, avec une complexité de calcul raisonnable.

Résumé des contributions

La thèse est organisée de la manière suivante:

Le **Ch. 1** donne une introduction générale sur l'imagerie par champ de lumière. Dans la première section, nous donnons d'abord la définition des champs de lumière et introduisons quelques notations utilisées tout au long de la thèse. Ensuite, nous résumons les systèmes d'acquisition de champ de lumière, la visualisation des données et donnons des exemples d'applications courantes. Nous présentons également brièvement les ensembles de données de champ de lumière existants et leurs caractéristiques. Comme l'une de nos contributions est de créer un jeu de données synthétique, nous présentons quelques

paramètres de configuration et des détails techniques sur notre jeu de données créé. Dans la deuxième section, nous nous concentrons principalement sur l'estimation de la profondeur, la synthèse de vue et l'interpolation vidéo, ces trois problèmes liés à notre travail, et nous résumons les méthodes de l'état de l'art dans chaque domaine en introduisant les principes de ces méthodes.

Dans le **Ch. 2**, nous proposons un système d'estimation de la profondeur (ou de la disparité) basé sur l'apprentissage qui convient à la fois aux champs de lumière échantillonnés de manière dense et clairsemée. Le système proposé se compose de trois étapes de traitement: un module qui estime la carte de profondeur initiale en se basant sur un réseau d'estimation de flux optique, une opération de fusion avec traitement d'occlusion et enfin un réseau de raffinement. L'estimation peut être effectuée à partir d'un sous-ensemble flexible de vues d'entrée, qui utilise pleinement les informations à vues multiples dans un champ de lumière. La fusion des cartes de profondeur initiales, reposant sur deux types de mesures d'erreur de warping, nous permet d'avoir une estimation précise à la fois dans les régions occultées et le long des contours. De plus, contrairement aux méthodes nécessitant le calcul des volumes de coûts, notre approche ne nécessite pas de définir la gamme de profondeur au préalable. Les résultats expérimentaux montrent que la méthode que nous proposons a obtenu des performances supérieures à celles des méthodes de l'état de l'art avec un sous-ensemble plus petit et plus flexible de vues d'entrée, et génère des cartes de profondeur plus précises.

Dans le **Ch. 3**, nous présentons un système basé sur l'apprentissage profond pour la synthèse de vues de champ de lumière à partir d'un sous-ensemble de vues d'entrée. S'appuyant sur un réseau d'estimation de flux optique léger pour obtenir des cartes de profondeur brutes, notre méthode utilise deux modules de reconstruction dans les domaines ***pixel*** et ***caractéristiques*** respectivement. Pour notre reconstruction au niveau des pixels, les occlusions sont explicitement gérées par un filtre d'interpolation dépendant de la disparité, tandis que l'inpainting sur les zones désoccultées est appris par des couches convolutives. En raison des incohérences de disparité, la reconstruction basée sur les pixels peut donner lieu à du flou dans les zones hautement texturées ainsi que sur les contours des objets. Au contraire, la reconstruction basée sur les caractéristiques fonctionne bien sur les hautes fréquences, ce qui rend la reconstruction dans les deux domaines complémentaire. L'apprentissage de bout en bout est finalement effectué, notamment par le biais d'un module de fusion fusionnant des reconstructions basées sur des pixels et des caractéristiques. Les résultats expérimentaux montrent qu'il atteint des performances d'état de

l'art avec des ensembles de données synthétiques et réels. Et il est même capable d'étendre la baseline du champ de lumière en extrapolant des vues de haute qualité sans nécessiter d'entraînement supplémentaire.

Dans le **Ch. 4**, en se basant sur nos travaux précédents, nous proposons une architecture résiduelle profonde qui peut être utilisée à la fois pour synthétiser des vues angulaires de haute qualité dans des champs de lumière et dans des vidéos classiques. Le système proposé consiste en un estimateur de flux optique optimisé pour la synthèse de vues, un encodeur de caractéristiques pouvant être entraîné et un réseau convolutif résiduel pour la reconstruction de vues basée sur les pixels et les caractéristiques. Parmi ces modules, la mise au point de l'estimateur de flux optique spécifiquement pour la tâche de synthèse de vue donne des informations de profondeur de scène ou de mouvement qui sont bien optimisées pour le problème ciblé. En coopération avec l'encodeur pouvant être entraîné de bout en bout, le bloc de synthèse utilise à la fois une synthèse basée sur des pixels et des caractéristiques avec des blocs de connexion résiduels, et les deux vues synthétisées sont fusionnées à l'aide d'un masque souple appris pour obtenir la vue finalement reconstruite. Les résultats expérimentaux avec divers jeux de données montrent que notre méthode se positionne favorablement par rapport à d'autres méthodes, avec un gain important pour la synthèse de vue de champ de lumière. De plus, avec peu de modifications, notre méthode peut également être utilisée pour l'interpolation vidéo, générant des images de haute qualité par rapport aux méthodes d'interpolation. De plus, nous intégrons les méthodes proposées dans Ch. 3 et Ch. 4 dans un système de compression de champ de lumière, montrant ainsi que le nouveau système peut surpasser le codage HEVC inter pour le champ de lumière et JPEG-Pleno avec un gain important en débit pour la tâche de compression.

Enfin, nous concluons notre thèse en résumant nos contributions et en discutant des perspectives futures de nos travaux.

INTRODUCTION

Context

Nowadays, multimedia consumers are searching for more realistic immersive experiences in applications such as movies, visio-conferences, and video games. This rising demand indicates a large commercial market and promising investment prospects, hence draws a considerable amount of attention from both industry and academia.

Classic 2D images that have hitherto been widely used in photography and video (where each frame is treated as an image) are merely a flat representation of a 3D scene, in which depth information that is crucial for reconstructing the 3D scene is neglected. This lack of geometry information makes 2D photographs unsuitable for applications that require realistic 3D geometry reconstruction or a feeling of immersion in a 3D scene. Consequently, new types of imaging that can store or easily reconstruct geometry information are now coming into the spotlight, for example: RGBD images used in automobiles and drones, stereo images used in augmented reality (AR) and virtual reality (VR), and light field images such as those produced by new-generation commercial light field cameras. Light field cameras capture a scene from different perspectives, thereby producing an array of images that represents the captured scene as observed from different viewing positions. Therefore, from a single exposure, it is possible to retrieve enough information to at least partially reconstruct the scene geometry based on the captured light field. This property has made light fields a promising imaging technology that has gained a lot of interest over the past decade.

Besides 3D imaging, another trend that has appeared over the last two decades is the development of data-driven algorithms. This has largely been triggered by significant advancements in Graphics Processing Unit (GPU) technology. The new GPU technology is powerful enough to support intensive computation in a very short time. For example, the Intel Pentium released in 2000 can conduct 7 Giga floating-point operations per second (7 GFLOPS), while the newest GPU Nvidia GeForce RTX 3080 released in 2020 can manipulate 29.8 Tera FLOPS. Moreover, large-scale distributed systems of GPUs can be used to take advantage of their joint processing capabilities. Thanks to these two

factors, it is now feasible for data-driven algorithms to process massive amounts of data and learn complicated patterns, which is hard or even impossible for classical methods. Even for data-driven algorithms, one can witness that, as the computing power increases, methods evolve from simple Linear Regression and Support Vector Machine (SVM) to computationally intensive methods such as Deep Learning methods. Deep Learning has also been shown to be able to deal effectively with a wide range of classical computer vision problems, such as image super-resolution, classification, edge detection, segmentation, etc. This has given rise to a series of exciting innovations in applications such as real-time pedestrian detection, autopilot, painting style transfer, face synthesis, etc.

One should note that the developments of 3D imaging and Deep Learning are not separate. On the contrary, many recent breakthroughs in 3D imaging have been achieved with the help of Deep Learning-based methods. In the process, Deep Learning has also successfully found more practical usage scenarios, waging a technological revolution in fields like cinematography and computational imaging. Light field imaging is amongst the new computational imaging modalities that is benefiting from the recent advances of Deep Learning.

Motivation and Goals

A light field refers to the collection of light rays emitted by a 3D scene and seen from any point of coordinates (x, y, z), along different orientations, in different wavelength and at different time instants. The light field can thus be defined by a 7D radiance function which can, under some assumptions detailed later in the manuscript, be simplified into a 4D function with two sets of coordinates, the angular coordinates which can be seen as different viewpoints and the spatial coordinates of the points on the scene. While a fine angular sampling of the light field may require recording an infinitely high number of directions, or viewpoints, in practice the number of viewpoints is limited and depends on the capture device. The concept is therefore quite broad and encompasses structured view captures on a plane with constant disparity between views, as well as unstructured captures with e.g. a mobile phone with an un-constrained displacement in the scene. In this thesis, we focus on structured light fields, which are a collection of images captured from a number of structured viewpoints (normally more than 2 points of view), where both two horizontally and vertically adjacent views always have equal baseline.

The observations from multiple viewing positions offer enough clues to infer the ge-

ometry of the scene. Nevertheless, it means that the light field is a data-heavy image modality, being unfriendly to both storage and transmission, and naturally contains a lot of redundancies. This leads to an interesting and meaningful challenge for light field practical usage. On one hand, we wish to reconstruct 3D scene geometry as precisely as possible from a captured light field; On the other hand, we hope that this reconstruction can utilize as few light field views as possible. We therefore pose the question:

1.How to precisely and efficiently estimate a depth map using a small subset of views in a light field?

One straightforward solution is to apply a stereo image depth estimation algorithm by taking a pair of views out of a light field, but this is not sufficiently precise and is unable to handle occlusions due to the lack of multi-view information. This challenging issue motivates us to develop a new, light field-specific algorithm that estimates depth by making use of a limited number of perspectives. We would furthermore like to build this algorithm based on deep learning, in order to take advantage of deep learning's computational efficiency.

The second challenging issue addressed in our research may be considered as an extension of the previously-mentioned issue:

2.Can we retrieve the entire light field from a small subset of views and the corresponding estimated depth values?

The solution to this issue could be useful for a range of applications. This is because, firstly, a reconstruction method that utilizes only a small subset of views for scene reconstruction would alleviate some of the storage and transmission burdens that dense light fields typically impose on transmission and display systems. Secondly, some captured light fields have a large disparity between adjacent views, which leads to perceptible discontinuities when changing viewing perspectives. Such light fields are hardly applicable for immersive applications. For such applications, reconstruction would be required to generate denser perspectives or to synthesize some perspectives that were unavailable before (e.g., perspectives located outside the captured viewpoint grid). Such a context provides further motivation in our work to develop an algorithm that can enable us to retrieve entire light fields from a sparse sampling of the captured views.

Although the proposed architecture for the previous issue enables us to upsample a light field in the angular dimension based on given views, there are still some network design issues for both training efficiency and test effectiveness. Therefore, we firstly focus on improving the proposed scheme with novel designs. Then we take a fresh step from

light field view synthesis problem to classical video frame interpolation by posing the question:

3.Can a view synthesis architecture for light field be generalized to solve temporal view synthesis problem?

As both light field and video sequence can be considered as scene samplings, one in the angular dimension and the other in the temporal dimension. Undersampled light field views may lead to perspective discontinuities when navigating within the scene, and undersampled video sequence will likewise suffer from jerky motion. Therefore, the motivation of this work is to generalize a network which is proven effective for light field angular upsampling to solve a similar video temporal upsampling problem.

The overall goal of our work is to build a series of algorithms addressing some of the most common challenges in light field processing or even video processing domains, such as depth estimation, view rendering, frame rate up-conversion etc. These algorithms can either be applied solely to solve the aforementioned issues, or serve as sub-modules to address other problems related to light fields or videos, for example, methods for light field view synthesis can be integrated into a compression pipeline. In addition, we try to keep our proposed methods general enough to adapt to both synthetic and real-world light field data with a wide baseline range, or even applicable to video data. Thanks to deep learning techniques, our learning-based methods are practical for some industrial usage with a reasonable computational complexity.

Thesis structure & Contributions

The dissertation is organised in the following manner:

Chapter 1 gives an overall introduction about light field imaging. In the first section, we first give the definition of light fields and introduce some notations used throughout the thesis. Then we summarize light field acquisition systems, data visualization and give examples of typical applications. We also briefly present existing light field datasets and their characteristics. As one of our contributions is to create a synthetic dataset, we thus present some configuration settings and technical details about our created dataset. In the second section, we mainly focus on depth estimation, view synthesis and video frame interpolation these three problems related to our work, summarizing state-of-the-art methods in each domain and introducing principles of these methods.

In **Chapter 2**, we propose a learning-based depth (or disparity) estimation frame-

work that is suitable for both densely and sparsely sampled light fields. The proposed framework consists of three processing steps: an initial depth estimation module based on optical flow estimation network, a fusion operation with occlusion handling and finally a refinement network. The estimation can be performed from a flexible subset of input views, which makes full use of multi-view information in a light field. The fusion of initial depth estimates, relying on two types of warping error measures, allows us to have an accurate estimation in both occluded regions and along the contours. Additionally, in contrast with methods requiring computation of cost volumes, our approach does not need any depth range prior. Experimental results show that our proposed method achieved superior performance than state-of-the-art methods with a smaller and more flexible subset of input views, and generates depth maps that are more accurate.

In **Chapter 3**, we present a deep learning-based framework for light field view synthesis from a subset of input views. Building upon a lightweight optical flow estimation network to obtain raw depth maps, our method employs two reconstruction modules in ***pixel*** and ***feature*** domains respectively. For our pixel-wise reconstruction, occlusions are explicitly handled by a disparity-dependent interpolation filter, whereas inpainting on disoccluded areas is learned by convolutional layers. Due to disparity inconsistencies, the pixel-based reconstruction may lead to blurriness in highly textured areas as well as on object contours. On the contrary, the feature-based reconstruction well performs on high frequencies, making the reconstruction in the two domains complementary. End-to-end learning is performed finally including a fusion module merging pixel and feature-based reconstructions. Experimental results show that it achieves state-of-the-art performance with both synthetic and real-world datasets. And it is even capable to extend light field's baseline by extrapolating high quality views without additional training.

In **Chapter 4**, based on our previous work, we propose a deep residual architecture that can be used both for synthesizing high quality angular views in light fields and temporal frames in classical videos. The proposed framework consists of an optical flow estimator optimized for view synthesis, a trainable feature extractor and a residual convolutional network for pixel and feature-based view reconstruction. Among these modules, the finetuning of the optical flow estimator specifically for the view synthesis task yields scene depth or motion information that is well optimized for the targeted problem. In cooperation with the end-to-end trainable encoder, the synthesis block employs both pixel-based and feature-based synthesis with residual connection blocks, and the two synthesized views are fused with the help of a learned soft mask to obtain the final

reconstructed view. Experimental results with various datasets show that our method performs favorably against other methods with a large gain for light field view synthesis. Furthermore, with a little modification, our method can also be used for video frame interpolation, generating high quality frames compared with interpolation methods. Additionally, we integrate the methods proposed in Ch. 3 and 4 into a light field compression pipeline, showing that the new pipeline can outperform light field HEVC inter coding and JPEG-Pleno[2] with a large gain in terms of bitrate for compression task.

Finally, we conclude our thesis by summarizing our contributions and discussing the future perspectives of our work.

Chapter 1

Background

1.1 Light field imaging

1.1.1 Definition for Light Field

The term 'Light Field' was first used by Gershun [3] to describe a collection of light rays that pass through every point in 3D space along different directions. Adelson and Bergen proposed the *plenoptic function* as a complete representation of a light field [4]. A plenoptic function is usually represented in 7-dimensional form as follows:

$$L(\theta, \phi, \lambda, t, V_x, V_y, V_z), \tag{1.1}$$

in which variables V_x, V_y and V_z can pinpoint the position in 3D space. Variables θ and ϕ are polar coordinates indicating light ray direction. Finally, the variable t denotes the moment of observation, and λ is the wavelength of a certain light ray. In the case of a static light field, the dimension t can be excluded, and λ can be also replaced by its RGB components in a color system.

A simplified light field representation termed the '***Lumigraph***' was proposed by Gortler *et al.* [5], assuming that the rays traverse a space with no occlusion and that the radiance stays constant along a light ray. The plenoptic function can consequently be represented by the values of the radiance of rays intersecting two parallel planes. This model based on two parallel planes is normally termed the '***Two-plane parameterization***'[6] and is illustrated in Figure 1.1.

In Figure 1.1, the two intersection points are denoted (x, y) and (u, v) respectively. The plane of (x, y) is termed the ***spatial plane*** as it describes the location of light rays when passing through a single lens (if all the light rays traverse the same point (u_0, v_0), it models a pinhole camera). The plane of (u, v) is termed the ***angular plane*** as it defines the perspective (angle) of observation. A static light field is then represented as a 4D

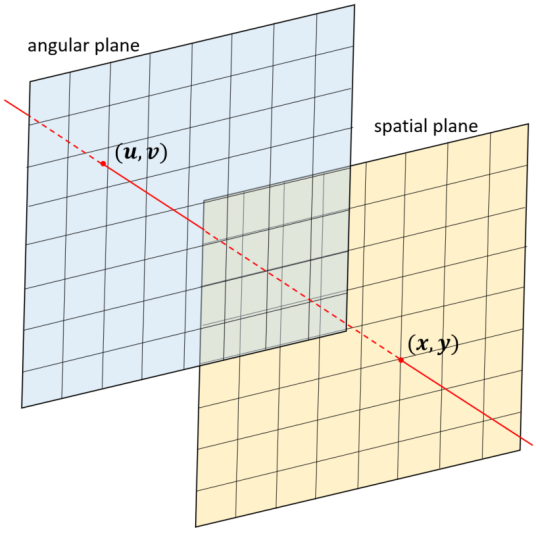


Figure 1.1 – Two-plane parameterization of the light field function.

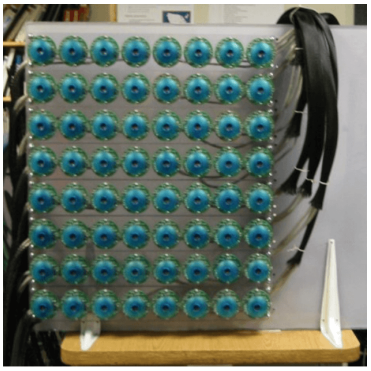
function $L(x, y, u, v)$ based on these two-plane parameterization. For the sake of notation consistency, we adopt this 4D function to represent a light field in the rest of this thesis.

1.1.2 Light Field Acquisition Devices

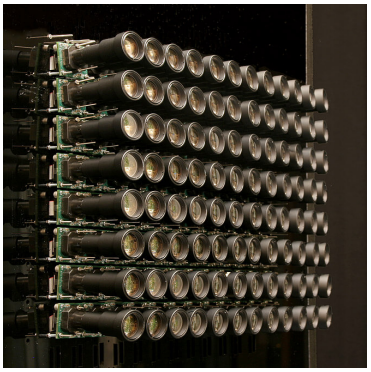
Light fields sample a scene in both the spatial and angular dimensions. When a conventional camera captures a scene, the light rays that come from different directions and reach the same spot on the sensor are integrated, so important angular information is consequently neglected. Therefore, this type of camera isn't suitable for the acquisition of light fields. Considering this limit of conventional cameras, more specialized devices for light field acquisition have been proposed in recent decades. These light field capture systems record the light rays' intensities while also preserving their direction information. Existing light field capture equipments can be roughly classified into four categories: camera arrays, plenoptic cameras, camera with modulation masks and others.

Camera arrays

The key idea in the design of camera arrays is to employ multiple cameras placed at different perspectives to increase the angular sampling. Based on this idea, both physical equipments and 3D graphics software plugins have been created for capturing a light field.



(a) MIT camera array



(b) Stanford multi-camera array

Figure 1.2 – Camera arrays for light field acquisition. (a) MIT camera array with 64 cameras. (b) Stanford camera array with 100 VGA video cameras. Figures are respectively from [7] and [8]

Physical equipment The first camera array was designed by Yang *et al.* [7] in 2002, which is shown in Fig. 1.2(a). This camera array is composed of 64 cameras placed in a 8×8 grid, with resolution of each camera being 320×240. Additionally, this array can capture light fields with a sampling rate of 18 FPS and a delay time of 80ms, meaning that it supports real-time light field acquisition. Then in 2004, Wilburn *et al.* propose a dense camera array [9] using 53 CMOS image sensors, which is able to capture light fields with a higher sampling rate (30 FPS) and a higher resolution (640×480). They further improve their work by proposing a larger-scale of camera array in 2005, the Stanford Multi-Camera Array [8](shown in Fig. 1.2(b)), where 100 VGA video cameras placed on a 2D grid can record live, synchronized video from all viewpoints at once, and light fields are later obtained by post-processing these recorded video streams. There are also some other camera arrays that have been built in recent years, both by academia and industry (such as Technicolor's camera array [10] for light field acquisition).

Since a camera array is composed of a set of synchronized cameras, the spatial resolution of a captured light field is dependent on the resolution of each single camera, and the angular resolution is dependent on the total number of cameras used. It can hence capture light fields with higher spatial resolution than plenoptic cameras (introduced in the following section), and also with reasonable angular resolution if there are enough cameras installed and rectified. In comparison with plenoptic cameras, camera arrays have a larger parallax or baseline, making captured light fields have larger disparity values between angular views. High spatial resolution and large parallax allow camera arrays to support many interesting applications, such as 3D reconstruction. Although camera arrays have

such advantages, their shortcomings are likewise obvious: a complicated camera array is not only expensive, energy-intensive and bulky, but also requires extra calibration and synchronization procedures after installing all cameras, to make sure that every view is well rectified. Such work is not trivial and sometimes necessitates engineering expertise. A badly configured camera array tends to produce artifacts in captured views, or even fails to generate usable light fields. Since image rectification and synchronization is not a research focus in this thesis, we assume that all the light fields used in our work are well rectified and synchronized.

Efforts are made by industries in recent years to reduce the cost and volume of camera arrays, in 2013, *Pelican* proposed an ultra-thin monolithic camera array (shown in Fig. 1.3) [11] designed for portable devices. Thanks to optical design optimization [12] and applications of light field super-resolution algorithms [13, 14], it can record a 4×4 views, with each view being an 8 Megapixel JPEG image, or record light field video in 1080p with frame rate 30fps.

(a) Pelican camera array on a chip

(b) Nokia Lumia powered by Pelican camera array

Figure 1.3 – Pelican camera array on chip and corresponding portable equipment. Figures are from website *light-field forum* [15].

3D graphics software As installing a physical camera array is always expensive and complicated, it is not always available for some research usage. Here we introduce a novel light field acquisition approach using camera arrays in 3D graphics software, which is adopted in our research as well. This approach is based on the open source software Blender [16] generally used for creating animation films, virtual games, 3D printed models etc. After creating a virtual scene and setting up a camera array with proper configuration, photorealistic light fields will be synthesized by a rendering engine after simulating all lighting conditions and physical effects. Fig. 1.4 illustrates some examples from the user interface of Blender, for rendering a light field. With Fig. 1.4(a) a raw model and camera

(a) Blender model and camera array setting (b) Blender rendered scene (c) Rendered sub-aperture image

Figure 1.4 – Blender user interface, with (a) Model and camera array setting page. (b) Rendered entire scene. (c) Central sub-aperture image captured by camera array.

array grid, Fig. 1.4(b) the entire scene rendered using its Cycle engine, and Fig. 1.4(c) the rendered central sub-aperture view. Let us note that in Fig. 1.4(c), the object materials, shadows, reflections and other physical effects are well simulated by the rendering engine, yielding realistic light fields applicable in research. Furthermore, as Blender supports film animation with a variable frame rate, dynamic light fields generated by Blender with a high frame rate will be useful for researches on light field video. Besides Blender, there also exist some other graphics software that can be used for generating synthetic light fields, such as Maya [17] and 3ds Max [18] with a commercial license.

Compared with a physical camera array, the largest advantage of using 3D graphics software is its flexibility. Parameters such as camera baseline or sensor size, that are almost impossible to be changed after installation of physical camera arrays, are easily re-configured in 3D software.

Plenoptic cameras

The second category of light field acquisition devices is the plenoptic camera. The main idea for its design is based on the work of Adelson et al.[21], where a single main lens is incorporated along with a lenticular array placed at the sensor plane. Light rays bounded by the main lens aperture will be re-organized by the lenticular array, allowing the camera to provide information about how the scene would look from various perspectives along a continuous trajectory. Thanks to the usage of microlenses, plenoptic cameras are compact and low-cost. So far, there has been two kinds of plenoptic cameras on the market: the

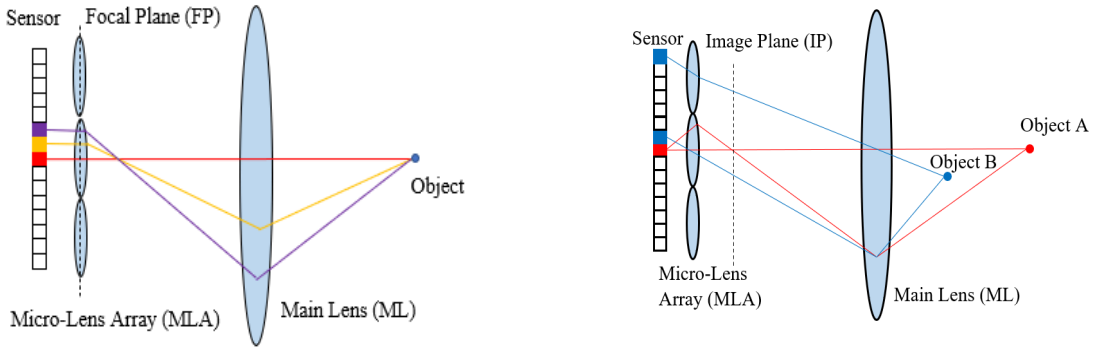
(a) Lytro 1st generation camera (b) Lytro Illum camera (c) Raytrix camera

Figure 1.5 – Commercial plenoptic cameras (unfocused 1.0 camera Lytro and focused 2.0 camera Raytrix). (a) Lytro 1st generation camera. (b) Lytro Illum camera. (c) Raytrix camera. Figures are from websites *CLIM* [19] and [20].

Lytro camera based on unfocused 1.0 design, which targets consumer market, and Raytrix camera based on focused 2.0 design, which aims at industrial customers.

Unfocused 1.0 cameras The first commercial hand-held plenoptic camera has been proposed by Ng *et al.* [22] in 2005 and later marketed as Lytro (shown in Fig. 1.5(a) and 1.5(b)) respectively in 2011 (Lytro 1st generation) and 2014 (Lytro Illum). The imaging principle of Lytro (unfocused 1.0 camera) is illustrated in Fig. 1.6(a), a micro-lens array is placed between the main lens and sensor, in the focal plane of the main lens. A bundle of light rays passing through the main lens and focused on the image plane will simultaneously traverse the focal plane of micro-lens. And after passing through the micro-lens, these light rays will be re-directed and will converge at infinity (i.e. never focused). Finally, on the sensor area right behind each micro-lens, incoming light rays from different directions will be detected as different pixels. We name this collection of pixels behind each micro-lens as a micro-lens image. By this design, the spatial resolution of captured light fields is dependent on the number of micro-lenses in the array and the angular resolution depend on the number of pixels within each micro-lens image. The micro-lens size (or micro-lens number) of such a plenoptic camera is always determined by the aperture or f-number of the mains lens, a too small micro-lens size will result in overlap of micro-lens image, while a too large size will consequently leads to waste of sensor area.

Let us note that light fields captured by a Lytro is a collection of micro-lens images, whose visualization will be introduced later in Ch. 1.1.3. The conversion from micro-lens images to sub-aperture images necessitates a rearrangement of pixels by extracting co-located pixels from each micro-lens image and placing them within an image, at spatial



(a) Unfocused 1.0 camera design (Lytro)

(b) Focused 2.0 camera design (Raytrix)

Figure 1.6 – Optic designs of unfocused 1.0 camera and focused 2.0 camera. In (a), the focal plane of micro-lens array overlaps the image plane of the main lens, hence a bundle of light rays focused on the image plane will be re-organized into parallel rays after micro-lens array. In (b), micro-lens array will focus on the image plane of the main lens, light rays focused on the image plane of the main lens will be focused again by micro-lens array.

coordinates that correspond to their source micro-lens images.

In comparison with camera arrays, plenoptic camera is a much more compact piece of equipment in lower price. Nevertheless, since spatial sampling and angular sampling of scene share the same sensor, a denser angular resolution will inevitably lead to a sparser spatial resolution. Limited by its aperture size, plenoptic camera has a smaller FOP than a cameras array, which means a smaller disparity value between adjacent angular views.

Focused 2.0 cameras One can increase spatial resolution of captured light fields by slightly changing optical design of plenoptic camera, this is the case of focused 2.0 cameras (Raytrix [20], shown in Fig. 1.5(c)). The main difference in the optical setup between unfocused 1.0 camera and focused 2.0 camera is the position of micro-lens array. The micro-lens array is placed farther away from main lens, with whose distance to the image plane of the main lens being larger than f_{MLA}. This design makes the micro-lens array focus on the image plane of the main lens (like shown in Fig. 1.6(b)). Or we can equivalently consider each micro-lens as a pin-hole camera, each camera captures a fraction of virtual image inside the camera. Objects that are in focus of the main lens (like object A in Fig. 1.6(b)) will be mapped onto the sensor with a high spatial resolution, points of objects that are out of focus of the main lens (such as object B in Fig. 1.6(b)) will be captured by several micro-lens images on the sensor, hence encoding angular information over these micro-lens images. The decoding of light fields captured with Raytrix neces-

sities a pre-estimated scene depth and complicated processing algorithms. This type of camera and bundled software are generally in high prices and mostly ordered by industry communities.

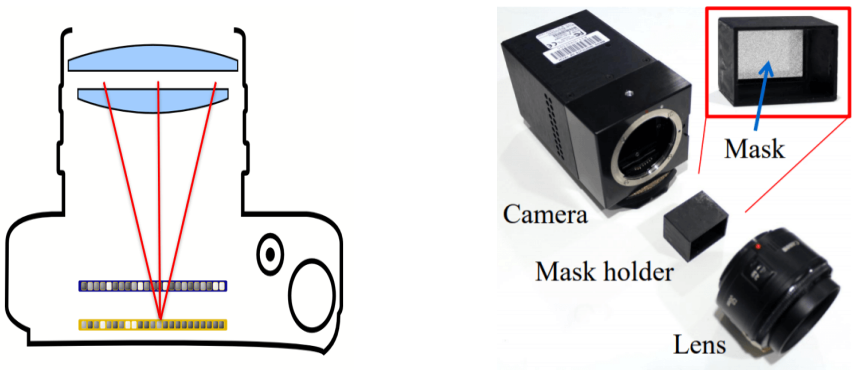
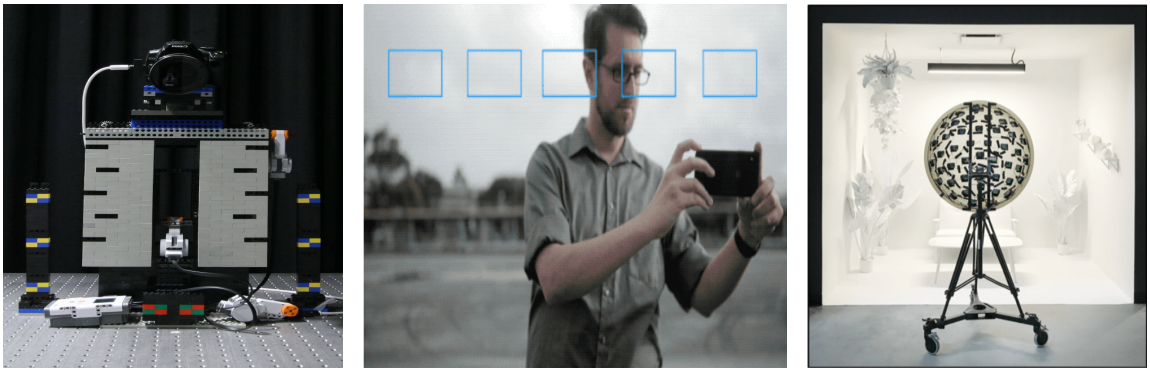


Figure 1.7 – Camera with modulation mask. Light rays traversing the main lens will be modulated by a translucent mask before being mapped onto the sensor. Figure on the left is schematic diagram, and figure on the right is a realized camera. Figures are from the project website [23].

Cameras with modulation mask

Coded mask designs have been considered instead of micro-lens arrays to modulate 4D light fields into 2D projections on the sensor, as shown in Fig. 1.7. Different reconstruction algorithms can then be used to restore the light field from its 2D projection [24, 25, 26], and the coded mask. The light field acquisition and reconstruction problems are formulated by a compressive sensing framework, in which the sensing matrix is materialized by a coded physical mask. Thanks to the use of a coded mask, instead of recording a spatial multiplex of 2D slices of the light field, as in micro-lens based camera architectures, the photosensor records a set of linear measurements from which a higher resolution light field can be reconstructed. Babacan *et al.* [27] place a randomly coded mask on the aperture plane to obtain incoherent measurements of the light field. Multiple shots are captured as random linear combinations of angular images by separately opening one region of the aperture and blocking the light in the others. The authors in [25] propose a camera architecture that records optically coded projections on a single image sensor, while the authors in [28] and [26] use respectively a random binary mask or a moving color coded mask affixed to the sensor to extract incoherent measurements. The authors in [29] propose a multi-mask camera model encompassing the colour filter array as well as the previous designs.



(a) Stanford light field lego gantry (b) Smartphone application (c) Light field capture rig

Figure 1.8 – Light field acquisition systems. (a) Light field gantry with lego. (b) Unstructured light field acquisition with a smartphone. (c) Spherical light field capture rig. Figures are respectively from [34], [35] and [36].

While the previous methods rely on patch-based dictionaries with a sparsity assumption for light field reconstruction from the set of sparse measurements, Choudhury *et al.* [30] employ learned convolutional sparse coding features to synthesize light field views. Light fields can also be reconstructed from a set of measurements using convolutional neural networks (CNN) [31, 32, 33], given a pre-defined coded mask.

The advantage of cameras with modulation mask is that they can increase angular resolution without decreasing spatial resolution. However, as the modulation mask is not a transparent medium, light efficiency is decreased, which may lead to a lower signal-to-noise ratio for the reconstructed light fields.

Others

In this thesis, we classify the acquisition methods that do not fall into the previous categories as the fourth class, including the light field gantries, and apparatus for capturing unstructured light fields. A light field gantry [34] is a specialized equipment to capture light fields, where a digital camera is mounted on a gantry controlled by a computer to move on a 2D plane (like shown in Fig. 1.8(a)). Although a gantry is easy to install and less expensive than a camera array, it is only suitable for capturing static light fields. Recently, applications of unstructured light fields have started becoming a hot topic in industry. Views captured on an irregular grid can enable the reconstruction of more complex 3D scenes. Several types of acquisition systems have been proposed. For example, in [37], the

(a) Lumigraph obtained with a plenoptic camera (b) 8 × 8 sub-aperture views obtained by re-organizing pixels of a lumigraph

Figure 1.9 – Visualization of a light field, with (a) lumigraph captured by Lytro Illum. (b) 8 × 8 subaperture views obtain by pixel re-organization. The light field used is from [1].

authors take advantage of the augmented reality toolkit available on a smartphone, to develop an application for sampling unstructured light fields of a real scene such as shown in Fig. 1.8(b). The authors in [36] propose a light field capture rig composed of 46 action sports cameras mounted to an acrylicdome as shown in Fig. 1.8(c), to record immersive light fields. Other equipments, such as the 16-camera rig in [38], have also been invented to capture unstructured light field views.

1.1.3 Light Field Visualization

Lumigraph

As introduced in Sec. 1.1.2, by placing a micro-lens array between the main lens and the sensor, a plenoptic camera captures a lumigraph, with the output of the sensor being a matrix of microlens images. Pixels within the same microlens image are derived from light rays having different orientations. Fig. 1.9(a) shows a lumigraph obtained with a plenoptic camera. We can see that this lumigraph is not in focus, as the micro-lens array, and not the sensor is placed in the focal plane of the main lens. Therefore, we need a post-refocusing process to retrieve focused images at different focus planes.

Sub-aperture image

A lumigraph captured by a plenoptic camera places pixels from all angular views within a single 2D plane. By re-organizing pixels from different angular viewpoints, we

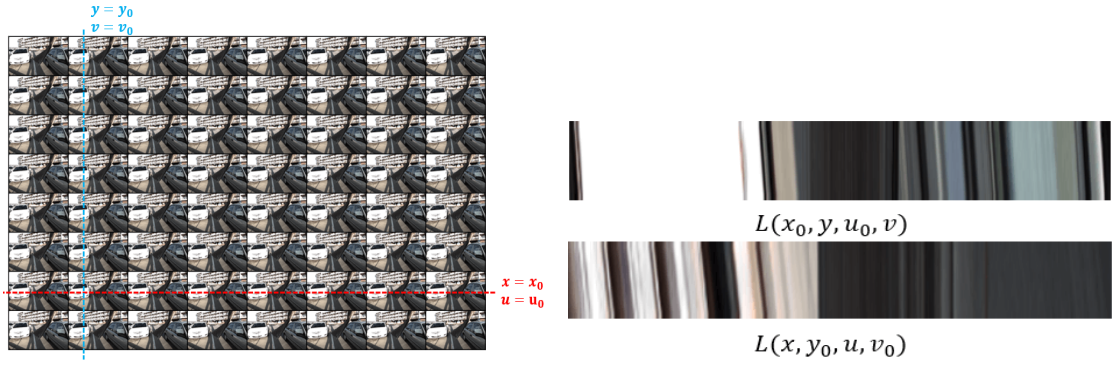


Figure 1.10 – Epipolar Plane Images (EPI) can be obtained by fixing one spatial and one angular dimension like shown in the left figure. Corresponding horizontal and vertical EPIs are shown in the right figure. The light field used is from [1].

can obtain a sub-aperture image array, where each image can be considered as an observation from a different perspective (e.g., as shown in Fig. 1.9(b)). Unlike a plenoptic camera, which requires pixel re-organization to obtain sub-aperture images, camera arrays and 3D graphics software can directly produce an array of sub-aperture images after synchronization and rectification of cameras.

Epipolar Plane Image

Another way of visualizing a light field is an Epipolar Plane Image (EPI), which is obtained by fixing one spatial and one angular dimension as shown in Fig. 1.10. Compared with a lumigraph or sub-aperture image array, with an EPI it is difficult to have the whole scene geometry information, as an EPI is merely composed of pixel vectors from a single row or column of views. However, the slopes of lines on the EPI are related to the depth values of corresponding pixels, so one can infer the geometry of the scene EPI by EPI. Occlusion regions can also be identified by intersections of those lines. EPIs are widely used to estimate the depth of a scene or to interpolate intermediate angular views. However, EPIs can be used for depth estimation, by analyzing the slopes of 1D structures only in the case of dense light fields. In the case of sparse light fields, large disparities between views may cause discontinuity of the corresponding pixels in each angular view on the EPI lines, hence leading to aliasing artifacts on the EPIs.

1.1.4 Examples of applications

A light field samples a scene in both the spatial and angular dimensions, and the angular information is important for a variety of applications such as, to give only a few examples, computational photography with digital re-focusing, face recognition and reconstruction, fluid flow reconstruction, object or material recognition, saliency detection.

Digital refocusing for computational photography

One typical application of light field data is computational photography with the digital refocusing capability [22]. Images captured with a conventional camera always have a determined focus plane after exposure, while with a light field, we can compute photographs with a focus on different focus planes from a single exposure. The refocused image I^p with the focus plane p can be obtained through a 'shift-and-average' algorithm as follows:

$$I^p_{(u_0,v_0)}(x,y) = \frac{1}{N} \sum_{(u,v)\in V} L(x - p(u-u_0), y - p(v-v_0), u, v), \tag{1.2}$$

where (u_0, v_0) denotes the angular position of the reference view, V is a set of angular positions around (u_0, v_0) and N is the total number of views in V. In Fig. 1.11, we show a set of photographs focused on different depth planes after digital refocusing.

Figure 1.11 – Light field refocusing after a single exposure. From left to right, photographs are focused from foreground to background. Figures are courtesies of [22].

Face reconstruction and recognition

Light fields can also be applied in face reconstruction and recognition problems. 2D facial images have been widely applied in the fields like security, mobile payment etc. In this process, relevant methods for facial image super-resolution [39], feature point detection [40], facial expression recognition [41] are correspondingly proposed to solve problems in these applications. 3D face model has advantages over its 2D counterpart in tackling challenges like variations in illumination, pose and scale. Reconstructing 3D face from 2D image is an ill-posed problem as it lacks the important depth prior for pixel projection. Existing methods such as Shape-from-shading (SfS) [42] or 3D Morphable Model (3DMM) [43] either rely on priors of reflectance properties and lighting conditions, but usually fail in retrieving facial details. In [44], the authors exploit the EPIs of captured light fields, and employ a CNN to estimate facial geometry from the extracted EPIs for further reconstructing corresponding 3D point cloud and surface mesh. Thanks to light field images, they finally reduce construction errors by over 20% against state of the arts on public datasets.

Light fields offer a new perspective for Presentation Attack Detection (PAD) of a face recognition system. The presentation of biometric artefacts at the sensor level has been one of the main dangerous attack forms for the biometric system, where an unauthorized person can deceive the security system by presenting information of the genuine user (e.g. by photos pre-printed or shown on a screen) to the sensor. And face recognition system based on a single RGB image has quite limited discriminating capability for these attacks. In [45], the authors apply light field cameras to replace conventional sensors in a face recognition system, where multiple focus images rendered from each capture instead of classical RGB images are analyzed by the system. The extracted focus variation information is proven effective for detecting the presence of spoof.

LF PIV-based 3D fluid flow reconstruction

3D fluid flow reconstruction has been a widely investigated problem in the engineering and scientific fields. Particle Imaging Velocimetry (PIV) that estimates the motions of particles injected inside the fluid is one of the standard solutions for this flow reconstruction problem. However, challenges still exist in the usage of PIV, since particles have similar appearances and are distributed at different depths, occlusions among them increase the tracking difficulty as well. Light field imaging is shown to be an effective solution for this

application, this is the work in [46]. The authors rely on the post-capture refocusing capability of light fields, using focal stacks to establish correspondences between particles at different depths. The focal stack symmetry property is also exploited to estimate accurate depths for those heavily occluded particles. Experimental results show that, in cooperation with other processing steps, this LF PIV-based solution finally yields an accurate 3D fluid flow reconstruction.

Material and object recognition applications

There are some other applications that make use of the angular information provided by light fields, and yield promising results. For example, the authors in [47] carry out a material recognition task on light field data by measuring the intensity variation under different viewing angles, and they gain 7% better recognition compared to using classical images. In [48], the authors propose a novel scale-invariant feature detector and descriptor that is built upon 4D light field data, which shows its usefulness for structure from motion and other tasks that match features across viewpoints of a scene. Light field imaging having additional angular patterns is also an effective alternative to RGB imaging in the detection of salient regions. The authors in [49] design a CNN dedicated to light field microlens image data, and show that it outperforms other, regular 2D image-based saliency detection methods.

In this section, we have summarised just a few applications of light fields, but there are many more computer vision applications that can also benefit from using light fields, e.g. , image segmentation [50].

1.1.5 Light field datasets

In this chapter, we will briefly introduce some public light field datasets that are commonly used by the research community and also in our work. Moreover, as a part of our contributions, we have also created our own synthetic light field dataset with the help of 3D graphics software Blender.

Real-world LF dataset

UCSD Illum dataset The UCSD Illum dataset was first proposed in the work of Kalantari *et al.* in [1], where they constructed a light field dataset captured with a Lytro Illum camera. The training set for this dataset contains 100 scenes (28 scenes are imported

from the Stanford Light Field Archives [51]), and the test set includes 30 scenes. All the scenes were with resolution $376 \times 541 \times 14 \times 14$. As the border views of a decoded light field are usually black, we use the central 8×8 views for training and for testing.

INRIA Illum dataset Like the UCSD Illum dataset, the INRIA Illum dataset is another real-world light field dataset captured with a Lytro Illum camera. It is composed of 58 light field scenes with a resolution of $376 \times 541 \times 14 \times 14$ without gamma correction. In our work, we use only the central regions of captured light fields, due to the same reason as with the UCSD Illum dataset.

There also exist other real-world light field datasets captured with a plenoptic camera, such as the EPFL LF dataset [52], the Stanford Light Field Archives [51] captured with a Lytro camera, the Light Field Video dataset captured with Raytrix [53] etc. We will not make an exhaustive introduction of these datasets here, as we have not used them for the work in this thesis.

Synthetic LF dataset

HCI synthetic dataset As mentioned earlier, 3D graphics software is a powerful tool for generating photorealistic light fields. One well-known synthetic light field dataset created with Blender is the HCI dataset (HCI old) proposed in [54]. This dataset contains 5 light fields with resolution $768 \times 768 \times 9 \times 9$ (*stilllife, buddha, buddha2, butterfly, monasRoom*), 1 light field with resolution $1024 \times 768 \times 9 \times 9$ (*medieval*) and 1 light field with resolution $1024 \times 576 \times 9 \times 9$ (*horses*). The disparity maps for all the views are also provided.

Another synthetic light field dataset, proposed in [55], is the 4D Light Field Benchmark (or HCI new). This dataset includes 28 light fields with resolution $512 \times 512 \times 9 \times 9$. Among them, 16 scenes in the 'Additional' category are provided with disparity maps for all the views, whereas for the 8 others in the 'Stratified' and 'Training' categories, the disparity information is available only for the central view. Disparity maps for the 4 scenes in the 'Test' category are not accessible, for the sake of online competition.

INRIA synthetic dataset The supervised learning of CNN models for depth or disparity estimation [56, 57] requires large datasets with ground truth disparity information. However, for existing datasets with ground truth disparity, the quantity and scene

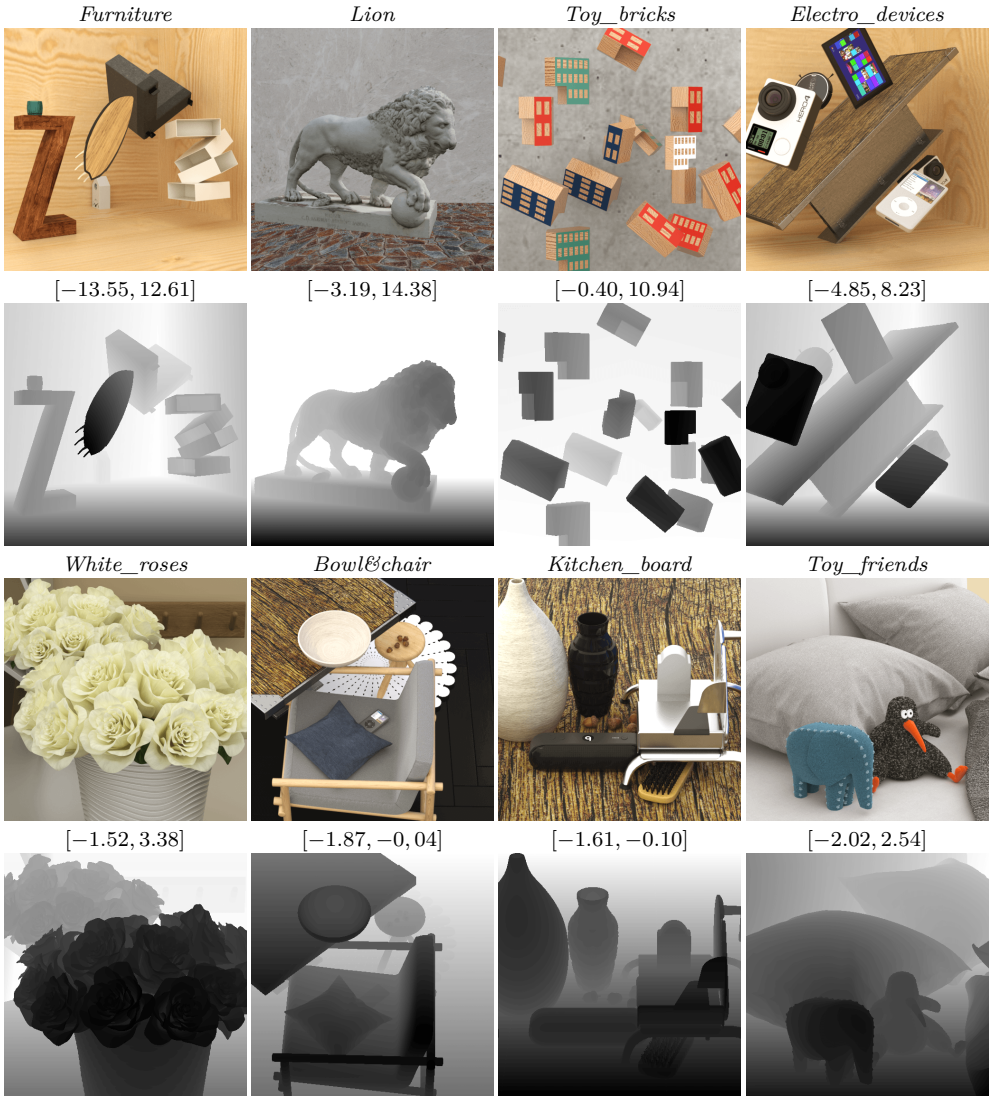


Figure 1.12 – Examples of scenes from our two datasets. The 1st and 2nd rows show four scenes and the corresponding disparity maps from SLFD, while the 3rd and the 4th rows show four examples of scenes and the corresponding disparity maps from DLFD.

variety of light field data are not sufficient for training a model. In addition, these datasets are limited to densely sampled light fields with narrow baselines.

Considering these limitations of light field datasets, we have created two synthetic datasets: a Sparse Light Field Dataset (SLFD) including 53 scenes with disparity range [-20,20] pixels between adjacent views, and a Dense Light Field Dataset (DLFD) containing 43 scenes with disparity range [-4,4] pixels. Each light field has the same resolution of $512 \times 512 \times 9 \times 9$ as the light fields in the HCI new dataset. Both SLFD and DLFD are provided with the disparity and depth maps for every viewpoint in the light fields. To the

best of our knowledge, SLFD is the first sparsely sampled light field dataset that provides ground truth depth and disparity information for every light field view.

The rendering of the light field scenes is performed with the open source software Blender and a light field generation addon, whose installation and usage are detailed in `https://github.com/lightfield-analysis/blender-addon` . The elementary models are downloaded from the websites Chocofur [58] ad Sketchfab [59] with a non-commercial CC license, and are assembled to create various 3D, mostly indoor scenes. The scenes contain textureless background, specular reflection, diffusion and object occlusion, which makes our dataset useful to measure the effectiveness of depth estimation algorithms. The 3D scene models in SLFD and DLFD are partly shared, but they are rendered with different camera baselines. Fig. 1.12 shows some examples of light field scenes and their corresponding disparity maps. As two HCI light field datasets have same disparity range as our DLFD, we can mix the HCI light field scenes with our DLFD when training a network.

Other datasets

Apart from light field datasets captured with a plenoptic camera or rendered with Blender, datasets captured with camera arrays are also available. The latter include, for example, the Google camera array dataset in [60], which contains 6 calibrated light fields of 5×5 views, or the dataset '*Space*' proposed in [38], which contains unstructured light fields captured with a 16-camera array etc. Here we will not detail much these datasets, as they are not the focus of our work.

1.2 Scene depth from light field and view synthesis: State of the art

In Ch. 1.1.4, we listed some novel applications powered by light fields, in which vision problems that are usually hard for classical images to handle are well tackled thanks to directional information of light fields. Albeit useful in such applications, light fields still need to face many challenges caused by their acquisition hardware, data redundancy etc. before a larger-scale adoption. Depth (or disparity) estimation and view synthesis from light fields, amongst these challenges, are two widely investigated topics and involve a large variety of practical usages. For example, the aforementioned facial reconstruction

and fluid flow reconstruction both involve estimating scene depth from captured light fields, and accurate estimates will certainly favor the following reconstruction procedures. View synthesis shows its importance in solving the spatio-angular trade-off of plenoptic cameras, and in applications like augmented and virtual reality which necessitate continuous perspectives when navigating within the scene. Let us note that depth estimation and view synthesis are tightly related problems, since synthesizing novel views sometimes relies on scene geometry as a prior for image warping.

Our work in this thesis mainly encompasses the problems of depth estimation and view synthesis from light fields. Additionally, although originally designed for light fields, our view synthesis solution is further extended to fit video sequences, as methods for light field view synthesis and video frame interpolation follow very similar principles. In the following sections, we will give an overview of prior work on depth estimation from light field and on view synthesis (from both light field and video sequence).

1.2.1 Light field depth estimation

Scene depth estimation is a highly-studied problem in the computer vision domain, and it is indeed needed for a variety of processing problems such as 3D reconstruction and view synthesis. Approaches using monocular image [61, 62, 63, 64, 65] or stereo image pairs [66, 67, 68, 69, 70, 71, 72, 73, 74, 75] for scene depth estimation have been widely investigated in recent decades. In comparison with those using a single RGB image or stereo image pair, methods using a light field have more available directional information of the scene to yield accurate predictions, especially in the occluded regions. As a consequence, depth estimation from light fields has become a novel active field with the emergence of commercial light field cameras.

We can classify the depth estimation methods into three categories according to the type of data used (EPI, sub-aperture view and focal stack), i.e. *EPI-based approaches*, *pixel matching-based approaches* and *focal stack-based ones*.

For those densely sampled light fields with small baselines, the corresponding pixels in different views of a light field form a line in an EPI, whose slope is proportional to the disparity value [76]. This property is employed to estimate depth by analyzing the specific linear structures in the EPI. For example, the authors in [77] use structure tensors to locally estimate these slopes, then this local estimation is placed in a global optimization framework using a variational approach. The authors in [78] propose a spinning parallelogram operator for disparity estimation in the EPI, accompanied by a confidence

measure to handle ambiguities and occlusions. Recently, classical analysis of EPI structure is replaced with deep neural networks, where EPI stacks are taken by a network to estimate a disparity map of the reference view. In [57], the authors concatenate the views in the same row to form an EPI volume, and feed it to a 3D convolution-based U-Net architecture [79] to estimate disparity maps for all input views. Albeit more efficient than non-learning-based approaches, it predicts disparity maps containing severe artifacts on the object contours. While in [80], instead of using EPI volume formed in one single direction, the authors propose a multistream architecture that takes EPI volumes formed in four directions (horizontal, vertical, left and right diagonal) and outputs a disparity map for the reference view. The exploitation of multi-directional EPI volumes deters contour artifacts in disparity maps, and achieves state-of-the-art performance. However, the architecture in [80] is constructed using 2D convolutional layers, which means the number of views used to form EPI volumes is fixed, users have to train several models for light fields in different angular resolutions. EPI-based methods depend on the linear structure in the EPI, hence are well suited for densely sampled light fields with small baselines, but fail for light fields with large baselines, as the linear structure is destroyed due to severe aliasing artifacts.

In the case where light fields are with large disparity values, pixel matching-based approaches show their superior performance in estimating accurate disparity maps. In contrast to EPI-based methods, pixel matching-based schemes adopt techniques from classical stereo reconstruction, i.e. , matching corresponding pixels in all sub-aperture images of the light field, essentially using robust patch-based block matching. To give a few examples, the authors in [81] estimate disparity by computing a matching cost volume between the central sub-aperture image and sub-aperture images warped using the phase shift theorem. Besides matching pixels across multiple views, the matching procedure can also be carried out on several stereo image pairs. In [82], after dividing a light field into several stereo image pairs, the authors estimate corresponding disparities through a multiscale and multiwindow (MSMW) stereo matching method, and then process them with an optical flow-based interpolation. The final disparity is obtained by a median fusion of the initial disparities. The approach in [83] consists of estimating disparities between the four corner views, then propagating them to the target viewpoint. Correlation between viewpoints is exploited by a low rank approximation model to cope with occlusions. The authors in [84] employ an empirical Bayesian framework to estimate scene-dependent parameters for inferring scene disparity. This algorithm is free of additional cues exploiting

dense view sampling, hence it is relevant for both dense and sparse light fields. Methods relying on stereo matching in their pipelines may adopt state-of-the-art disparity estimation methods [56, 85] or optical flow estimation methods [86, 87, 88, 89, 90] to ensure accurate initial disparities. This is the case of [91], where the authors employ a state-of-the-art optical flow estimation network *FlowNet 2.0* [87] to predict accurate disparity candidates, then propagating them from input viewpoints to the target position and performing an occlusion-aware soft 3D reconstruction. In comparison with EPI-based approaches, pixel matching-based approaches are more suitable for light fields with large baselines, e.g. light fields captured using camera arrays, but are more sensitive to lighting conditions variations. The change of lighting from one view to another may lead to massive mismatched pixels.

The last category of depth estimation methods is based on focal stack. A focal stack refers to a collection of images focused at different depths. Scene depth can be estimated by analyzing either focus variation (depth from defocus) or sharpness of pixels in a sequence of images (depth from focus). By digital refocusing, a light field can be used to produce a synthetic focal stack for scene depth estimation. Here, we take the work in [92, 93] as examples. The authors in [92] propose a pipeline that combines both defocus cue and correspondence cue to estimate scene depth. They first construct 2D slice of EPI by considering horizontal spatial dimension and vertical angular dimension. Then the defocus cue is obtained by computing the horizontal spatial variance after vertical angular integration, and the correspondence cue by computing the vertical angular variance. To exploit the strengths of both cues, they finally fuse two local estimates into a high quality depth map using Markov Random Fields (MRFs). While in [93], the authors propose a depth estimation pipeline by exploiting two features of the focal stack: one is the symmetry of the non-occluded pixel along the focal depth dimension centered at the in-focus slice. The other is the data consistency between synthesized focal stack using estimated depth and that computed directly from the light field. And an iterative optimization procedure is finally carried out to incorporate both features for obtaining better depth maps. Although a focal stack is easy to be produced from light field captures, the depth estimated from focal stack, when compared with those from the other two types of approaches, is still prone to artifacts and less accurate. In addition, focal stack-based approaches are generally used for densely sampled light fields with small baselines, since sparsely sampled light fields with large baselines will generate focal stacks full of artifacts.

1.2.2 Light field view synthesis

Like we have introduced previously, light fields captured with plenoptic cameras suffer from a spatio-angular trade-off, this problem is addressed by applying super-resolution (SR) algorithms to enhance the resolutions in the spatial (spatial SR) [94, 95, 96, 97], angular (angular SR or view synthesis) [1, 60, 37, 98] or both (spatio-angular SR) dimensions [77, 99]. One main aspect of our work in this thesis is about light field angular SR or view synthesis. The problem of view synthesis consists of generating novel views based on a set of camera captures. A straightforward solution to this problem is to first estimate the scene's 3D geometry as a prior, novel views are then obtained by projecting given views to the desired viewpoints. However, 3D modeling of the scene is not trivial, in the case where the scene contains complex structures, a poorly constructed 3D model may lead to synthesis failures. We put this type of global geometry-based methods aside from our discussion since accurate 3D reconstruction appears to be another long story.

Depth-dependent schemes

Instead of using global geometry of the scene, a more practical manner to realize view projection is to employ local geometry i.e. scene depth, in an explicit form of depth map, or implicit forms of Plane Sweep Volume (PSV), Multi-Plane Image (MPI) etc. We classify this family of synthesis methods using scene depth into the first category as *depth-dependent schemes*.

A typical depth-dependent synthesis pipeline involves two-stage processing: first estimating a depth map, then warping the source views based on the estimated depth and blending them into the target view. Efforts to improve such a Depth Image-Based Rendering (DIBR) pipeline consist in reducing depth errors and in handling non-Lambertian surfaces in the scene. Taking the work in [100, 77] as examples, the authors in [100] adopt the superpixels augmented with synthesized depth to operate a local shape-preserving warping which compensates for depth inaccuracy. The warped views are merged, with weights specified by camera orientation but also the reliability of depth information in each warped superpixel. While the authors in [77] impose some smoothness constraints in EPIs when synthesizing novel views. They first propose a 2D structure tensor to estimate disparity values in EPIs, then based on the estimated disparity map, they warp views to the target position and optimize final synthesis results under a smoothness constraint in EPIs.

In recent years, deep neural networks have shown great success in coping with various computer vision tasks, and learning-based DIBR pipeline has also been investigated. In [1], the authors cascade two convolutional networks to synthesize the intermediate views with given four corner views, whose input is the concatenated means and standard deviations computed among four corner views warped from source positions to the target position using a series of discretized disparity values. The first network analyzes the mean and standard deviation features to predict a depth map. After backward warping four corner views to the target position with the learned depth map, the second CNN blend these warped views to output the color of the target view. In comparison with a non-learning-based DIBR approach, end-to-end training such a learning-based pipeline merely requires minimizing synthesis error between the final output and ground truth, without paying much attention to the intermediate depth accuracy. However, this network, mostly designed for dense structured light fields, fail in occluded regions, especially for sparse light fields having large parallax. Due to the depth imprecision, the synthesized views also suffer from blurriness, tearing and ghosting effects. The authors in [101] construct a similar architecture as in [1] to retrieve a densely sampled light field from a single image, where the first network estimates the scene depth, and the second one tackles ray occlusions. This scheme is only suitable for retrieving densely sampled light fields having simple scenery.

Both of the work in [100, 1, 101, 77] involve predicting an explicit depth map for the following image warping procedure. Recently, methods using implicit depth priors such as PSV, MPI, sheared EPIs and multiple depth candidates have likewise yielded promising synthesis results. Instead of using a depth map, the authors in [60] propose a soft model of scene geometry that retains uncertainty in form of a distribution over depth values, from which they further formulate soft visibility functions used for depth refinement as well as for view synthesis procedures. Final synthesis is a blending of K nearest views with weights based on consensus and visibility scores, where consensus scores give the probability for a pixel coordinate (x, y) of a given camera that the input views agree for sampled depths defined within the interval $[z_{min}, z_{max}]$, and visibility scores measure how much a pixel coordinate can still see through the scene from a sampled depth within the interval $[z_{min}, z_{max}]$. Although being able to generate smooth and plausible renderings of difficult content, textureless regions and soft object edges, the iterative optimization and blending score computation make this algorithm computationally intensive and memory-consuming for its practical application.

More efficient algorithms are proposed by combining deep learning techniques and implicit depth priors. In [98], the authors shear EPIs with different disparity values, and train a CNN to score each sheared EPI, the final synthesized view is a blending of sheared EPIs according to their confidence scores. Like EPI-based solutions proposed for other vision tasks, this EPI-based synthesis pipeline is only suitable for densely sampled light fields. Furthermore, one can observe a synthesis quality degradation when the target view is distant from the source views. Besides sheared EPIs, MPI is also proposed for view synthesis problem, especially for those with large baselines. The MPI representation is generally composed of color images and corresponding alpha maps, the multiplication of both will produce a set of scene slices, with whose pixels belonging to a certain depth range. The authors in [102] employ MPI representation to realize baseline magnification by extrapolating novel views from narrow-baseline captures. In the same vein, the authors in [37] adopt MPIs for light field view interpolation task. They construct a 3D convolution-based network that takes PSV to predict MPI (including color images and transparency maps) for each input view, then warp these MPIs to the target position and merge them via a weighted combination. The usage of 3D convolutions in the architecture makes it capable to support a variable number of input views, and the usage of MPI has shown its power for wide-baseline view synthesis task. Nevertheless, as this method is designed to synthesize novel views from unstructured inputs, it is necessary to estimate the camera pose for each view. The camera pose estimation is always time-consuming and sometimes fails due to insufficient input views and small parallax. In comparison with MPI, of which each slice corresponds to a certain depth, the authors in [103] propose the notion of depth probability volume, for image warping procedure and selection of image patches for view refinement. This method shows better view extrapolation effectiveness than the work in [102].

Here, we just want to remind readers that methods in [1, 98, 37] all involve the applications of PSV, but in different manners: the authors in [37] directly use PSV as input of their CNN to learn MPIs, while the authors in [1] calculate means and standard deviations of the PSV as input features. The authors in [98] use sheared EPIs, which can be considered as a variant of PSV, as EPI is sheared with different depth values. The form of EPI used is always related to the algorithm designs.

Depth-independent schemes

The methods mentioned above depend on the estimated scene geometry. While in parallel, methods exploiting signal priors, e.g. sparsity priors in the 4D Fourier domain, sparsity in the shearlet transform domain, smoothness on EPI have also been proposed for view synthesis. We classify this category of synthesis methods as *depth-independent schemes*.

The sparsity of Fourier domain is an important property of densely sampled 4D light fields, which has been exploited to reconstruct entire light fields from a small set of samples. The authors in [104] propose a linear view synthesis pipeline using the dimensionality gap of light fields as a prior. This dimensionality gap states that the 4D Fourier transform of light fields includes only a 3D subset of entries whose energy is significantly higher than zero. In the pipeline, they first construct a focal stack by performing a 'shift and average' operation on a set of views sampled on 1D circular or grid trajectory. Novel views are then rendered by applying a spatially uniform, depth-invariant deconvolution to the focal stack. Observing that much of sparsity in continuous spectra is lost in the discrete domain due to the windowing effect, and this loss of sparsity can severely limit reconstruction quality, the authors in [105] propose a synthesis algorithm that recovers a discrete signal by optimizing for sparsity in the continuous domain. More specifically, under a *k-sparsity* assumption, the Continuous Fourier Transform (CFT) of light fields can be decomposed of k continuous positions of frequencies ω_i and corresponding coefficients a_i. The synthesis step consists of optimizing the set of $\{\omega_i, a_i\}$ by minimizing the reconstruction errors with a gradient descent approach. The initialization of the algorithm requires sampling light field views in a specific 1D pattern, for example, box and two diagonals, which in general limits its practical usage.

Light field view synthesis problem can also be tackled using EPIs in different manners. In [106], the authors model the light field view synthesis as a learning-based detail restoration in the EPIs. They first apply a bi-cubic angular interpolation on input EPIs from which spatial high frequencies have been removed and use a CNN to restore details in the angular domain of the interpolated EPIs. The spatial details are then recovered by a non-blind deblur operation. While the authors in [107] consider the view synthesis problem as inpaintings in EPIs, they aim at reconstructing EPIs of densely sampled light fields from its sparse counterpart, by performing regularization in discrete Shearlet transform domain. It shows superiority in coping with semi-transparent objects. Note that both of [106] and [107] realize view synthesis EPI by EPI, hence are naturally time-consuming.

More efficient EPI-based synthesis can be achieved by operating on EPI stacks. The authors in [108] build a pseudo 4D CNN for reconstructing densely sampled light field from sparse samples, where 4D neural network is composed of 2D strided convolutions for upsampling EPI stacks, and a 3D CNN connected with angular conversion operation for recovering details.

Methods exploiting spatial-angular cues for light field reconstruction are proposed recently, this type of methods is able to handle both angular and spatial super-resolution at the same time. To give a few of examples, the authors in [109] describe a spatio-angular restoration network using 4D convolutions and high-resolution residual blocks, in order to extract spatial features that preserve geometrical properties. And the authors in [110] propose a two-stage pipeline which first generates the whole set of novel views, and then refines spatial texture details. However, these methods can synthesize novel views with a pre-defined angular upsampling rate (e.g. 2×2 to 7×7), changing upsampling rate necessitates to training a new model from scratch.

1.2.3 Video frame interpolation

Light field view synthesis and video frame temporal interpolation methods follow very similar principles. Although developed in parallel with methods specific to one problem or to the other one, like those proposed for light field view synthesis, video frame interpolation methods can be similarly classified into two categories: one category of methods which first estimate optical flows later used to warp input frames. The second category of methods does not rely on explicit estimated motion but instead relies on signal priors that are specific to the input data characteristics.

Flow-based interpolation

Flow-based methods consist in first estimating optical flows between given frames, and then in interpolating target frames along the motion vectors. This problem of video frame interpolation can be solved in a multiple frame context for better handling occlusions. To give an example, the authors in [111] propose a frame rate up-conversion framework using four consecutive frames as inputs. It first estimates four initial motion vectors (MVs) for given frames. The four initial MVs are then used to temporally interpolate the intermediate MVs, and in the determination of the interpolated MVs, the authors employ temporal matching error and spatial variation of MVs to define a reliability measure for handling

occlusion areas. Finally, frames warped with the interpolated MVs are fused into a single frame via a variational image fusion process. Interpolation methods using multiple frames on one hand is able to better handle occlusions and motion ambiguities, on the other hand, is more computationally intensive than those with two frames.

More work using two consecutive frames is proposed for the frame interpolation task. Observing that frame interpolation implicitly solves for dense correspondences between the input image pair, the authors in [112] propose an approach called Mind (matching by inverting a deep neural network) to trace back pixel matches from the interpolated frame generated during the forward-propagation through the trained network. They realize the frame interpolation but with more focus on inferring correspondences between input frames, the generated frames contain lots of artifacts such as ghosting effects and blurriness.

The authors in [113] were the first to introduce an architecture with an unsupervised learning of the optical flow. Similarly to the classical fully convolutional approaches, their Deep Voxel Flow (DVF) network is trained end-to-end to interpolate an intermediate frame from two consecutive input frames. The last layer of the network is a non-trainable interpolation function interpolating the previous layers output and the input frames. The network thus learns to estimate a kind of optical flow and a temporal mask for the trilinear interpolation layer. The network outputs a voxel flow which represents the per pixel displacement (dx, dy) and a temporal mask dt weighting the trilinear blending. Similar to [112, 113] where the optical flow is learned in the frame interpolation context, the authors in [114] couple a flow estimation network to a video processing component, then jointly train them to obtain a task oriented flow (TOFlow) and processed video frames.

The authors in [115] propose a double U-net pipeline where the first U-net takes two input frames to predict bi-directional optical flows. The predicted flows are then linearly combined to approximate the intermediate bi-directional optical flows. The second U-net is used to refine the intermediate optical flows as well as to predict soft visibility maps in order to handle motion boundary artifacts. Based on visibility maps and refined flows, two input frames are warped and linearly merged to get intermediate frames. Since none of their learned network parameters is time-dependent, their approach is hence able to synthesize as many intermediate frames as possible, producing a *'slow motion version'* of the input video.

Most of the above interpolation methods assume that brightness is constant and mo-

tion is linear. However, for videos containing complicated motions and brightness, synthesis based on this assumption may lead to imprecise results or even artifacts. Considering this limitation, the authors in [116] describe brightness and position with high-order model, in order to predict video content more accurately. More precisely, they approximate the pixel intensity with a 1st-order polynomial composed of a constant part and an inter-frame variation part, and the position with a 2nd-order motion trajectory including an acceleration term and a linear motion term. The coefficients are optimized through the minimization of an auto-regressive prediction error energy term and a reconstruction error energy term. More visually pleasant interpolated frames are then produced with the help of these two polynomials. The final frame is synthesized by performing a motion-compensated interpolation by pixel-wise intensity variation and motion trajectory. However, optimization of two energy terms makes this scheme time-consuming and less practical for usage.

For flow-dependent schemes, the precision of the estimated optical flows can determine directly the quality of the final synthesis. To better predict optical flows between input frames, in recent years, some frame interpolation methods adopt off-the-shelf state-of-the-art flow estimation networks in their pipelines. For example, the work in [117, 118, 119] all employ a state-of-the-art flow estimator 'PWC-Net' [90] to predict motion vectors between frames, the advanced PWC-Net shows its effectiveness in estimating more accurate optical flows, especially for those with large motions. The authors in [117] use optical flows predicted with PWC-Net to warp input frames as well as their deep feature maps (named contextual information in their work) to the target instant and blend them with another synthesis network. The application of contextual features improves the quality of details in the synthesized views. Since this scheme is not end-to-end trainable, the flow estimation and the view synthesis step are carried out one after another, predicted motions are thus not optimized for the synthesis task. Besides, occlusions are tackled with the help of contextual information in the synthesis procedure, but not explicitly handled in the warping procedure.

Observing that pixels of a closer object should cover those of a further object in the case of occlusions (pixel overlaps), the authors in [118] propose a depth-aware video frame interpolation architecture that explicitly copes with occlusions using scene depth, where they employ an off-the-shelf optical flow estimation network [90] and a single image depth estimation network [120] respectively for flow and depth inferences. An adaptive warping layer controlled by optical flow and depth is then used to warp the input frames, depth

maps, contextual features and optical flows to the target instant. A synthesis network finally takes these warped tensors and local interpolation kernels as inputs to produce the output frame. Nevertheless, as we all know, single image depth estimation is a tough task. The trained depth estimation model can not always produce reliable depth maps for all types of scenes, which has consequently become a drawback of the proposed frame interpolation method.

Warping errors are used instead of depth information to handle occlusions in [119]. This is based on the observation that pixels having larger brightness errors are more likely to be overlaid after warping. The authors propose a differentiable forward warping operation named '*softmax splatting*' to warp input frames and corresponding deep features without predicting flows at the target instant. Warping errors are involved in the interpolation of pixel values to tackle overlaps. In comparison with their previous work in [117], occlusions are better handled both with deeper contextual features and error-based warping operations. Additionally, the whole pipeline is end-to-end trainable, which means flows estimated by PWC-Net are optimized for the following synthesis process.

Most of the above methods focus on predicting optical flows between frames, while the authors in [121] use a multi-flow multi-attention generator to estimate flows between features. The feature flows are then used to warp and interpolate feature maps of the input frames to the target position. And the output frame is reconstructed from these warped deep features. To our best knowledge, this is the first frame interpolation method using feature flows, which shows its effectiveness in the case of severe occlusions.

Flow-free interpolation

The second category of methods can be referred to as flow-free interpolation methods. This is the case of kernel-based methods and phase-based ones. kernel-based methods such as AdaConv [122] and SepConv [123] based on a convolutional neural network estimate a series of spatially-adaptive interpolation kernels. The synthesized views are obtained through the convolutions between input frames and these learned kernels. More precisely, the authors in [122] proposed an adaptive convolution approach for video frame interpolation (AdaConv). For each pixel, parameters for a 2D convolutional kernel are predicted, then the input frames are convolved at the current pixel location to predict its interpolated value. The method is able to handle occlusions, blur and brightness changes. However, a large kernel is required to handle large motion and would be very computationally expensive. This drawback is addressed in [123] by introducing separable convolutions (SepConv)

using 1D kernels for faster processing. Please note that such spatially-adaptive filters and optical flows can nevertheless be combined as in [124], where the authors propose an architecture in which motion vectors and compensation filters are estimated using convolutional neural networks. The architecture includes an adaptive warping layer based on optical flow and compensation filters for synthesizing the target frame.

Phase-based methods[125, 126] are another kind of motion estimation-free interpolation solutions. They consist of representing motion with per pixel phase shift. Hence the intermediate frame is a result of operating a phase modification on each pixel. The authors in [125] proposed an architecture that propagates phase information across oriented multi-scale pyramid levels using a bounded shift correction strategy. This method yields robust results in the case of strong illumination changes, where flow-based methods often fail to work. The authors in [126] combine the phase-based approach with a learning framework to propose a novel architecture named PhaseNet. PhaseNet mirrors the hierarchical structure of the input phase decomposition and predicts the phase and amplitude of the in-between frames. The final frame is reconstructed based on these predictions. Although this work tackles large motion and higher frequency content better than [125], it still can't reach the same level of details as flow-based methods do.

Chapter 2

Depth estimation in light field

2.1 Introduction

Light fields allow estimating 3D scene geometry from a subset of views. As described in Ch. 1.2, estimation methods can be broadly broadly categorized into EPI-based schemes [77, 78] or pixel matching-based ones [81, 84, 83]. EPI based methods are only suitable for densely sampled light fields. Although a few learning-based architectures have been recently proposed for scene depth (or disparity) estimation from dense light fields based on EPI[57, 80], yielding promising performances, very few methods have been proposed so far for sparse light fields. While stereo matching methods allow estimating large disparities, they need to discretize the disparity space to compute cost volumes. A high discretization level improves estimation accuracy but yields a heavy computational cost, leading often to estimating depth at the central viewpoint only. Besides the above disparity estimation methods, a disparity map can be also estimated with an optical flow estimator, as both disparity and optical flow measure pixel displacement between two images. Recently, the field of optical flow estimation has known significant advances thanks to the use of deep neural networks trained in a supervised or unsupervised manner.

In this chapter, we propose a supervised deep learning framework for estimating scene geometry, taking at the input a flexible subset of light field views. In order to compute scene depth, the proposed approach estimates disparity maps for every viewpoint of the light field. The use of subsets of input views allows us, compared to stereo estimation methods, to increase the estimation accuracy, while limiting computational complexity. Initial disparity estimates are computed between aligned stereo pairs using the *FlowNet 2.0* optical flow estimation architecture that we fine-tuned to be suitable for disparity estimation in dense and sparse light fields. These initial estimates are used to warp a flexible set of anchor views onto a target viewpoint. The fusion of these initial estimates relying on a *winner-takes-all* (WTA) strategy with two measures of warping errors reflecting disparity inaccuracy in occlusion-free and occlusions respectively, allows us to have an accurate

disparity estimation in occluded regions and along the contours. A refinement network is then proposed to learn the disparity maps residuals at different scales. The proposed new architecture relies in part on the one considered in [91], by however extending the approach into a more general framework, which enables to perform estimation from a flexible subset of input views.

According to the metrics defined in [55][127], experimental results show that the proposed approach outperforms state-of-the-art light field disparity estimation methods for both densely and sparsely sampled LF. In addition, it does not require any prior information on disparity range as in [84, 78, 81] for example.

2.2 Methodology

2.2.1 Architecture overview

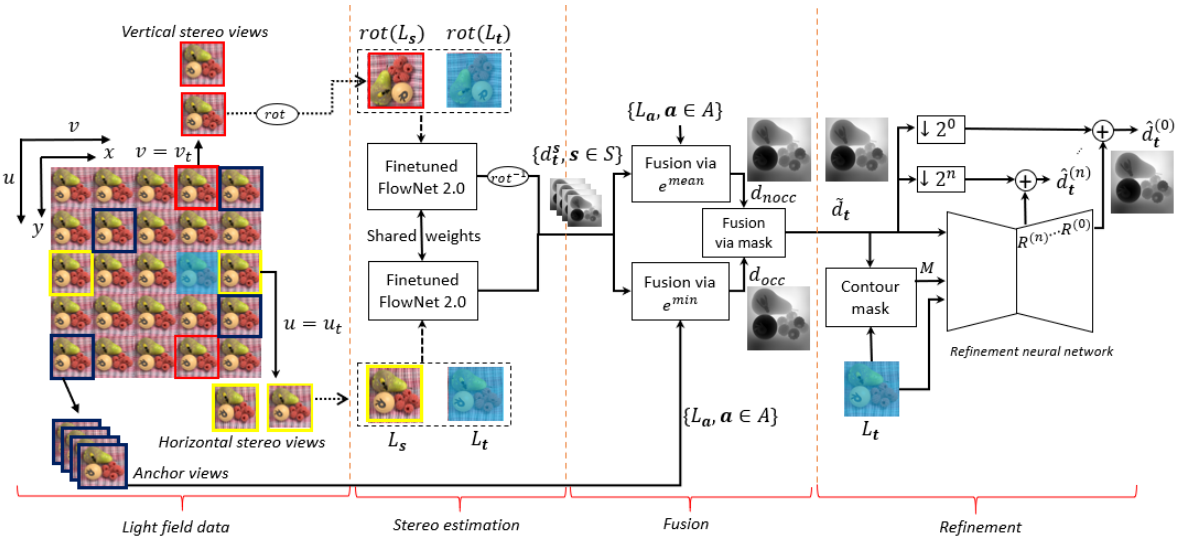


Figure 2.1 – Overview of proposed framework. We take a 5×5 LF as example. The blue masked view, called ***target view*** $L_\mathbf{t}$, is the view for which we search to estimate the disparity. Views in the yellow and red rectangles are respectively horizontal and vertical ***stereo views*** denoted $L_\mathbf{s}$. Target and stereo views are used to compute the initial disparity maps d_i using a fine-tuned *FlowNet 2.0* model. ***Anchor views*** $L_\mathbf{a}$ (in dark blue rectangles) can be any subset of views, except the target view, that are used to compute the warping error for the fusion of initial estimates. A multi-scale residual learning network corrects fusion artifacts and smooths the final disparity map in a last refinement step.

Given a 4D light field representation $L(x, y, u, v)$, we will refer to a light field view by the index of its angular position, e.g., $L_\mathbf{i}$ where $\mathbf{i} = (u_i, v_i)$, and denote $d_\mathbf{i}^\mathbf{j}$ the disparity

between two views $L_{\mathbf{i}}$ and $L_{\mathbf{j}}$ normalized by the distance between the two views.

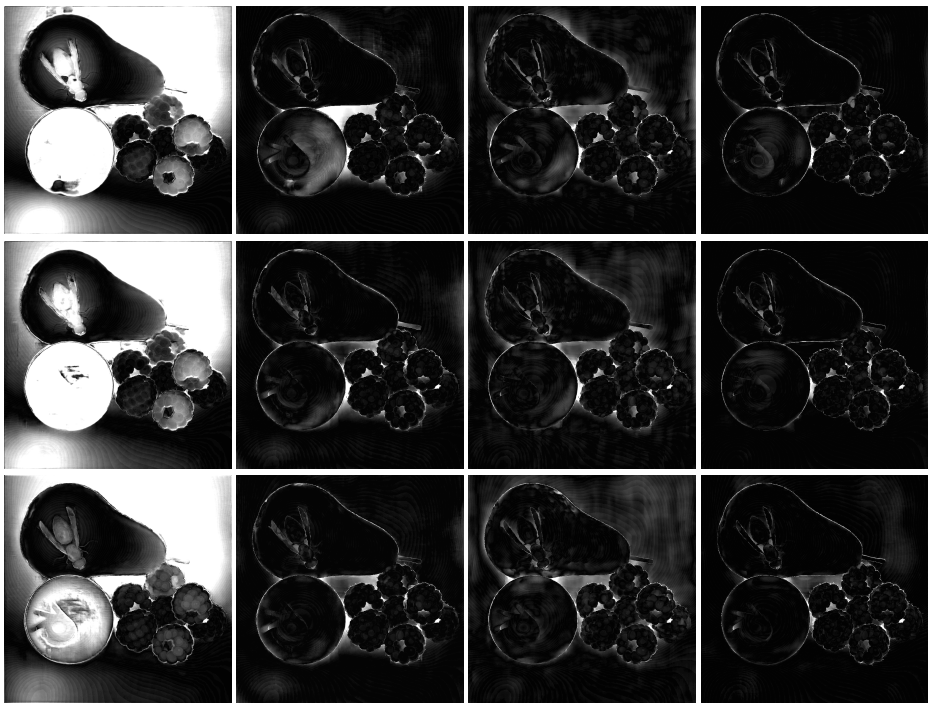
The proposed learning framework to estimate depth (or disparity) for any light field viewpoint, from a subset of input views is depicted in Fig. 2.1. The approach is composed of three main steps, i.e. *stereo estimation*, *fusion* and *refinement*. We denote $L_{\mathbf{t}}$ the ***target view*** for which the disparity map is to be estimated. Multiple coarse disparity maps on this target position are first estimated by a convolutional network trained for stereo estimation. The model, that we call *FN2-ft-stereo*, is obtained by fine-tuning a pre-trained *FlowNet 2.0* network with light field stereo pairs. Each of these disparity maps is computed between $L_{\mathbf{t}}$ (shadowed in blue) and a ***stereo view*** $L_{\mathbf{s}}$ located on the same row (framed in yellow) or on the same column (framed in red). For vertical image pairs, a rotation of 90° is applied such that the displacement flow between these two images only contains the horizontal component. Accordingly, the obtained disparity map should be also rotated by 90° in the reversed direction. The disparity maps between the target and the stereo views are denoted $d_{\mathbf{t}}^{\mathbf{s}}$, $\mathbf{s} \in S$, with S being the set of used stereo view positions.

These multiple estimates of the disparity information on the target view $L_{\mathbf{t}}$ should be fused to a single disparity map. To achieve this, we leverage the warping error from a set of ***anchor views*** (framed in blue) $L_{\mathbf{a}}$, $\mathbf{a} \in A$, with A denoting the set of anchor view positions. The disparity value corresponding to the smallest error per pixel is selected for the fused disparity map. In order to better handle the object boundary, the warping error is computed differently for pixels within occlusion areas or those within occlusion-free areas.

This fusion is simple and efficient, but is prone to noise and discontinuity because the decision is made pixel by pixel. Further refinement is realized by a second CNN which learns in a supervised fashion the residual signals of the disparity at multiple scales by an encoder-decoder architecture. Our network structure is flexible with respect to the anchor views, i.e. anchor views can be located at any viewpoint of the light field, and no additional training is required if the anchor view positions are changed.

2.2.2 Fine-tuned FlowNet 2.0 for disparity estimation

FlowNet 2.0 (*FN2*) [87] is an efficient CNN-based optical flow estimator. Two parallel branches of sub-networks are combined, the first specialized in large displacements estimation and the second in small displacements. The final stage of the network merges the two previously estimated flows taking into account the flow magnitude. Thanks to this structure, it is relevant to apply *FN2* to estimate disparity for light field views with



(a) *FN2* (b) *FN2-ft* (c) *DispNet-ft* (d) *FN2-ft-stereo*

Figure 2.2 – Disparity estimation errors (display range between 0 and 1) using different models: (a) *FN2*, (b) *FN2-ft*, (c) *DispNet-ft* and (d) *FN2-ft-stereo*. The first row corresponds to the estimation errors using a stereo pair $L_{2,2}$ and $L_{2,8}$. On the second and third row, for *FN2* and *FN2-ft*, the estimation has been done between the views $L_{5,5}$ and $L_{8,8}$, and the horizontal (second row) and vertical (third row) flow components are shown. Since *DispNet-ft* and *FN2-ft-stereo* only take stereo pairs, the horizontal flows are estimated between $L_{5,5}$ and $L_{5,8}$ (the second row), and the vertical flows are estimated between $L_{5,5}$ and $L_{8,5}$ (the third row).

variable disparity ranges.

Let us denote $f(L_{\mathbf{i}}, L_{\mathbf{j}}) = (f^x_{\mathbf{i}\to\mathbf{j}}, f^y_{\mathbf{i}\to\mathbf{j}})^\top$ the flow estimation operator between the views $L_{\mathbf{i}}$ and $L_{\mathbf{j}}$. Assuming that the light field is well rectified and regularly spaced in both angular directions, the disparity map between the view $L_{\mathbf{i}}$ and $L_{\mathbf{j}}$, normalized by the distance between the views, can be computed as

$$d^{\mathbf{j}}_{\mathbf{i}} = \frac{f^x_{\mathbf{i}\to\mathbf{j}}}{v_i - v_j} = \frac{f^y_{\mathbf{i}\to\mathbf{j}}}{u_i - u_j}. \tag{2.1}$$

In order to well adapt the operator $f(\cdot,\cdot)$ to disparity estimation between two light field views, we fine-tune the pre-trained *FN2* model. Two strategies have been considered. The first one feeds the model with pairs of light field views $L_{\mathbf{i}}$ and $L_{\mathbf{j}}$ with no constraint on view positions, and the model learns dense optical flows both on the horizontal and vertical directions. The obtained model is denoted *FN2-ft*. Another approach is to learn

the model using image pairs $L_{\mathbf{i}}$ and $L_{\mathbf{j}}$ on the same row ($u_i = u_j$) or on the same column ($v_i = v_j$). Note that images on the same column are rotated 90° to become a horizontal stereo pair $\Big(rot(L_{\mathbf{i}}), rot(L_{\mathbf{j}})\Big)$. The obtained model is thus named *FN2-ft-stereo*. Formally, with *FN2-ft-stereo*, the disparity map for the view $L_{\mathbf{i}}$ is computed as

$$d_{\mathbf{i}}^{\mathbf{j}} = \begin{cases} \frac{f^x_{\mathbf{i}\to\mathbf{j}}}{v_i - v_j}, & \text{if } u_i = u_j \\ \frac{rot^{-1}(f^x_{\mathbf{i}^*\to\mathbf{j}^*})}{u_i - u_j}, & \text{if } v_i = v_j \end{cases} \tag{2.2}$$

where $f^x_{\mathbf{i}^*\to\mathbf{j}^*}$ denotes the horizontal flow component between $L_{\mathbf{i}^*}$ and $L_{\mathbf{j}^*}$ with $L_{\mathbf{i}^*} = rot(L_{\mathbf{i}})$. The symbols $rot(\cdot)$ and $rot^{-1}(\cdot)$ are counterclockwise and clockwise rotation of 90°.

In Fig. 2.2, we compare the estimation accuracy of the *FN2-ft-stereo* model against three other models: *FN2* (the pre-trained *FlowNet 2.0* model), *FN2-ft* and *DispNet-ft*, which is obtained by finetuning a pre-trained *DispNet* model [56] with our stereo light field views. The models *FN2* and *FN2-ft* can estimate displacement both in x and y dimensions, whereas *DispNet-ft* and *FN2-ft-stereo* focus on 1D (horizontal or vertical displacements) estimation. On one hand, *FN2-ft-stereo* performs better than *FN2* and *FN2-ft*, which shows the necessity of concentrating on 1D estimation. In addition, *FN2-ft-stereo* is significantly better than *DispNet-ft*, both being finetuned using the same training set of stereo light field views.

Therefore, we choose to use *FN2-ft-stereo* for computing a set $D_{\mathbf{t}}$

$$D_{\mathbf{t}} = \{d_{\mathbf{t}}^{\mathbf{s}}, \mathbf{s} \in S\} \tag{2.3}$$

of multiple estimates of disparity $d_{\mathbf{t}}^{\mathbf{s}}$ between the target view $L_{\mathbf{t}}$ and one of the stereo views $L_{\mathbf{s}}$. As each of the candidates $d_{\mathbf{t}}^{\mathbf{s}}$ is normalized by the distance between the views in the considered pair, it represents the amount of disparity between the view and its immediate neighboring views. In the sequel, we will denote this set of normalized disparity maps as $D_{\mathbf{t}} = \{d_k, k = 1..K\}$, with K the number of candidate maps.

2.2.3 Fusion based on warping error maps

Although our *FN2-ft-stereo* model provides satisfying results for disparity estimation with stereo pairs, information in other available views of the light field is not exploited. In this subsection, we propose to fuse the candidate maps in $D_{\mathbf{t}}$ into a single disparity

map based on the error of warping the anchor views $L_{\mathbf{a}}$, $\mathbf{a} \in A$ onto the target view.

Based on one of the disparity maps $d_k \in D_{\mathbf{t}}$, backward warping is applied to project the anchor view $L_{\mathbf{a}}$ to the target position $\mathbf{t}$. The warped view is denoted $\widetilde{L}^k_{\mathbf{a}\to\mathbf{t}}$. The corresponding warping error $e^{\mathbf{a}}_k$ is computed by summing on the three R, G, B color channels:

$$\forall \mathbf{a} \in A, e^{\mathbf{a}}_k = \sum_{R,G,B} (L_{\mathbf{t}} - \widetilde{L}^k_{\mathbf{a}\to\mathbf{t}})^2 \tag{2.4}$$

Warping errors are then fused by taking into account all the warped views $\widetilde{L}^k_{\mathbf{a}\to\mathbf{t}}$ with $\mathbf{a} \in A$. The fusion is performed either by the 'average' or by 'min' operations as

$$e^{\text{mean}}_k(\mathbf{p}) = \text{mean}_{\mathbf{a}} e^{\mathbf{a}}_k(\mathbf{p}), \mathbf{a} \in A \tag{2.5}$$

$$e^{\text{min}}_k(\mathbf{p}) = \text{min}_{\mathbf{a}} e^{\mathbf{a}}_k(\mathbf{p}), \mathbf{a} \in A. \tag{2.6}$$

Both the error maps e^{mean}_k and e^{min}_k suggest the reliability on values in the disparity map d_k, but possess complementary properties. The error map e^{mean}_k reflects well the disparity inaccuracy in the occlusion-free zones, since it averages the contribution from all the warped views. Nevertheless, in occluded areas, interpolation in large holes becomes the main cause of warping errors instead of disparity inaccuracy. In this case, e^{min}_k turns out to be a more relevant measure. Indeed, at a pixel $\mathbf{p}$ that can be seen in the warped view $\widetilde{L}^k_{\mathbf{a}'}$, but not in another warped view $\widetilde{L}^k_{\mathbf{a}}, \mathbf{a} \in A, \mathbf{a} \neq \mathbf{a}'$, the value $e^{\text{mean}}_k(\mathbf{p})$ is misleading because of the high contribution of the error $e^{\mathbf{a}}_k(\mathbf{p})$. On the contrary, the 'min' operation gets rid of the perturbation from the occluded views. However, if a pixel $\mathbf{p}$ is occluded in all the warped views, neither $e^{\text{mean}}_k(\mathbf{p})$ nor $e^{\text{min}}_k(\mathbf{p})$ gives a reliable measure of disparity inaccuracy. It is preferable that the anchor view positions $\mathbf{a}$ are dispersed in the light field such that a pixel occluded in one view may be seen in another view. The impact of anchor view positions on the quality of the final disparity map will be discussed in Ch. 2.4.2.

To fuse at each pixel $\mathbf{p}$ the candidate disparity values $d_k(\mathbf{p}), k = 1..K$, a *winner-takes-all* strategy is employed according to error values $e^{\text{mean}}_k(\mathbf{p})$ and $e^{\text{min}}_k(\mathbf{p})$:

$$k' = \arg\min_k e^{\text{min}}_k(\mathbf{p}) \tag{2.7}$$

$$d_{\text{occ}}(\mathbf{p}) = d_{k'}(\mathbf{p}) \tag{2.8}$$

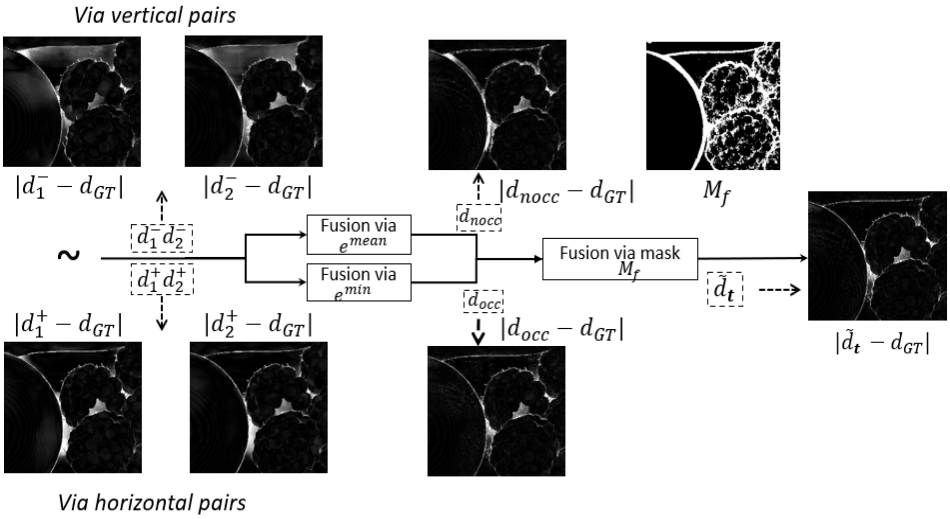


Figure 2.3 – Disparity error (compared to the ground truth) at each step of the fusion process. d_1^-, d_2^- and d_1^+, d_2^+ are coarse disparity estimates using *FN2-ft-stereo*, computed with horizontal or vertical image pairs ('-' indicates horizontal image pairs and '+' indicates vertical ones). d_{nocc} and d_{occ} are respectively fused disparity maps via e^{mean} and e^{min}, and $\widetilde{d}$ is the final resulting map using the binary mask M_f.

and

$$k'' = \arg\min_k e_k^{\text{mean}}(\mathbf{p}) \tag{2.9}$$

$$d_{\text{nocc}}(\mathbf{p}) = d_{k''}(\mathbf{p}) \tag{2.10}$$

Two fused disparity maps are obtained, d_{nocc} for occlusion-free zones and d_{occ} for occluded areas. To reduce local inconsistency, a 3×3 mean filter is applied on the error maps e_k^{mean} and e_k^{min}. A binary mask M_f defined as

$$M_f(\mathbf{p}) = \begin{cases} 1 & \min_k(e_k^{\text{mean}}(\mathbf{p})) > \theta \\ 0 & \text{otherwise} \end{cases} \tag{2.11}$$

is used to merge these two resulting disparity maps. The value $M_f(\mathbf{p})$ equals to 1 if $\min_k(e_k^{\text{mean}}(\mathbf{p}))$ exceeds a certain scene-dependent threshold θ, which is fixed at the value at the 90 percentile (errors at the occluded pixels are generally higher than those at the non-occluded ones). Note that M_f is not the real occlusion mask contrary to that used in [83]. However, it can be computed much more efficiently, and it approximates well the real mask.

Finally, one unique disparity map at the target position $\mathbf{t}$ is obtained as

$$\widetilde{d}_{\mathbf{t}} = d_{\text{occ}} \odot M_f + d_{\text{nocc}} \odot (1 - M_f) \tag{2.12}$$

where $\odot$ denotes pixel-wise Hadamard product. Fig. 2.3 demonstrates the gain in the fusion process. The errors on the estimated disparity compared to the ground truth are illustrated for each step of the process.

2.2.4 Multi-scale disparity refinement

The fusion step enables us to take advantage of multiple estimates and significantly improves the estimation accuracy. Nevertheless, as the fusion is implemented by a per pixel *winner-takes-all* (WTA) selection, discontinuity may exist in the resulting disparity map. Further refinement operation is useful for quality enhancement.

The work in [85] proposed a two-stage disparity learning framework. The learned disparity between a stereo pair is refined with a multi-scale residual learning network. As input, their network takes two stereo color images, the previously estimated disparity map, the warped image, as well as the corresponding warping error image. In a multi-view stereo scenario where we exploit the color information from multiple stereo views $L_{\mathbf{s}}, \mathbf{s} \in S$ and anchor views $L_{\mathbf{a}}, \mathbf{a} \in A$, it is obvious that this structure is no longer applicable. Indeed, the large number of input views, as well as the number of initialized disparity maps, will rapidly enlarge the size of the network and increase the computational cost during training. Moreover, the scheme is not flexible with respect to the varying number of input views.

The fusion of disparity estimates (Ch. 2.2.3) partially resolves this problem. Regardless of the variable number of stereo views, as well as that of the initialized disparity maps, only one single disparity map $\widetilde{d}_{\mathbf{t}}$ has been obtained for the target position $\mathbf{t}$. Besides $\widetilde{d}_{\mathbf{t}}$, two other images are fed to our refinement network: the target view $L_{\mathbf{t}}$ serving as color guidance and a binary mask M indicating contour misalignment. The mask M is computed as

$$M = |\Gamma(\Psi(\widetilde{d}_{\mathbf{t}})) - \Gamma(\Psi(\widetilde{d}_{\mathbf{t}})) \odot \Gamma(\Psi(L_{\mathbf{t}}))| \tag{2.13}$$

with $\Gamma(\cdot)$ being the dilation operator and $\Psi(\cdot)$ being the canny contour detector (Three parameters for canny detector: standard deviation of the Gaussian filter is 2, lower threshold is set to 10% of the max value and higher threshold is set to 20% of the max value).

Compared to the network used in [85], we change the inputs and the first layer of the refinement network, and we construct a 9-layer convolutional encoder to extract features and a 16-layer decoder to retrieve the estimated disparity map. The proposed network contains about 3.6×10^7 trainable parameters. During training, the residual signals of the disparity are learned at different resolution scales $n \in [0, N]$, supervised by the ground truth disparity. At each scale n, the network generates the residual signal $R^{(n)}$, which is added to the downsampled disparity map $\widetilde{d}_{\mathbf{t}}^{(n)}$

$$\widehat{d}_{\mathbf{t}}^{(n)} = \widetilde{d}_{\mathbf{t}}^{(n)} + R^{(n)}, \tag{2.14}$$

where $\widehat{d}_{\mathbf{t}}^{(0)}$ denotes the final obtained full resolution disparity map.

The loss is summed over all the resolution scales $n \in [0, N]$ as

$$\mathcal{L} = \sum_{n=0}^{N} \mu_n \mathcal{L}^{(n)}, \tag{2.15}$$

where μ_n is the contribution weight for the loss at the scale n and $\mathcal{L}^{(n)}$ is the sum of two losses

$$\mathcal{L}^{(n)} = \lambda_1 \mathcal{N}(\widehat{d}_{\mathbf{t}}^{(n)}, d_{\mathrm{GT}}^{(n)}) + \lambda_2 \mathcal{G}(\widehat{d}_{\mathbf{t}}^{(n)}, d_{\mathrm{GT}}^{(n)}), \tag{2.16}$$

where $\mathcal{N}$ denotes the sum of absolute differences (SAD)

$$\mathcal{N}(d_1, d_2) = \sum_{\mathbf{p}} |d_1(\mathbf{p}) - d_2(\mathbf{p})| \tag{2.17}$$

and where $\mathcal{G}$ is a gradient term defined as

$$\mathcal{G}(d_1, d_2) = \sum_{\mathbf{p}} \|G(d_1, d_2, \mathbf{p})\|_2 \tag{2.18}$$

with

$$G(d1, d2, \mathbf{p}) = \Big(\nabla_x d_1(\mathbf{p}) - \nabla_x d_2(\mathbf{p}), \nabla_y d_1(\mathbf{p}) - \nabla_y d_2(\mathbf{p})\Big)^\top. \tag{2.19}$$

2.3 Implementation details

2.3.1 Training data preparation

For fine-tuning *FlowNet 2.0*, stereo views are extracted on the same row or the same column of a light field. Image pairs located on the same column are rotated with a counterclockwise 90° to convert vertical pixel displacement to horizontal displacement. The two images in an extracted pair are separated by an angular *distance* corresponding to a view index difference $l \in [2, 3, ...8]$ for dense light fields, which corresponds to a disparity range within [-32,32] pixels, whereas for sparse light fields, this *distance* is set to be $l \in [1, 2, 3]$, corresponding to a disparity range within [-60,60] pixels. In both cases, the extraction of views is done in such a way that the different distances (or disparities) are well represented (with same probability) in the training data.

2.3.2 Data augmentation

The authors in [86, 57] performed geometrical and chromatic transformations to increase diversity in the training data. In our experiments, however, we have found that geometrical transformations such as rotation, translation or scaling that involve data interpolation bring extra errors in the ground truth disparity values, and thus harm the learning convergence. As a consequence, only chromatic transformation has been applied by changing the hue, saturation, contrast and brightness of training images. Concretely, we convert the images from the RGB space to the HSV space, add an offset to the hue and saturation channels, and then convert the images back to RGB color space. The hue and saturation offsets are uniformly picked from $[-0.3, 0.3]$ and $[0.7, 1.3]$. To perform contrast augmentation, we compute the mean pixel values $\bar{c}$ of each image channel c, then adjust c to $(c - \bar{c}) \times \zeta + \bar{c}$, where ζ is a contrast factor uniformly picked from $[0.7, 1.3]$. The brightness augmentation is implemented by adding a brightness offset to each of the RGB channels of an image, which is randomly picked from $[-0.1, 0.1]$.

2.3.3 Learning details

Different learning schedules are employed for fine-tuning the *FN2-ft-stereo* model and for training the refinement network. In the finetuning step, thanks to the pre-trained model, a shorter learning schedule can be adopted with an initial learning rate set to 0.0001 for the first 500 epochs. The learning rate is then decreased by half every 200

epochs. For the training of the refinement network which is randomly initialized, the schedule is longer with an initial learning rate of 0.0001 for the first 1200 epochs. The learning rate is then divided by 2 every 200 epochs. We use the Adam optimizer, and becaused of the limited GPU memory, a batch size of 4 is used. Tensorflow is used to implement our network. It takes about 2 days to train our network with a NVIDIA Tesla P100 GPU with 16G GRAM.

2.4 Experiments

2.4.1 Setup

To validate the effectiveness of our proposed framework, we conduct experiments on both synthetic and on real light fields.

Synthetic Dataset

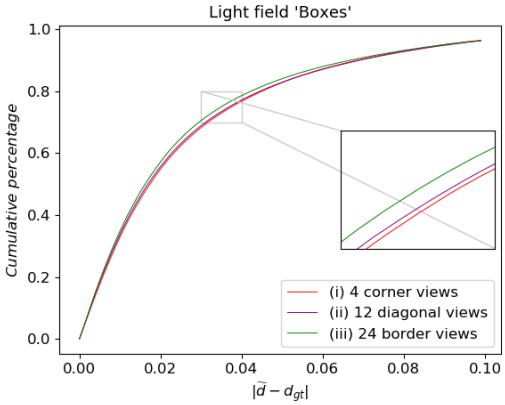
For sake of comparison, we use the synthetic light fields of the old and new HCI datasets [55][54] and keep the same test light fields as in [83]: *Stilllife, Buddha, Butterfly, MonasRoom* from [54] and *Boxes, Cotton, Dino, Sideboard* from [55]. We also evaluated the proposed framework using our own datasets INRIA synthetic dataset introduced in Ch. 1.1.5. Four test light fields *Furniture, Lion, Toy_bricks, Electro_devices* are used for evaluation. The scene *Lion* contains a single object and the other three scenes contain multiple objects.

Real Light Fields

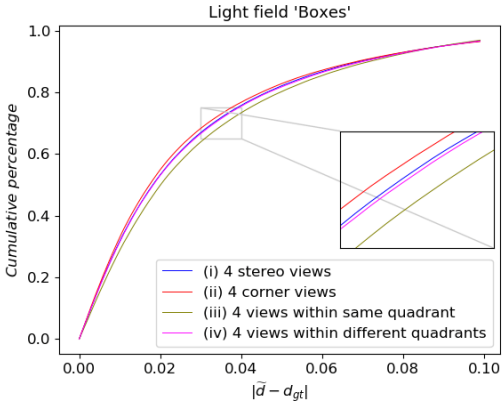
We have also tested our framework with real-world light fields, using INRIA illum dataset and EPFL illum dataset presented in Ch. 1.1.5. Compared with synthetic datasets, light fields captured by plenoptic cameras are more challenging due to the fact that the extracted views contain noise and geometrical distortions. Finally, experiments have been also carried out for real-world sparse dataset captured by Google camera array [60].

2.4.2 Impact of the anchors views

In contrary to other deep learning frameworks [57, 80], our network is flexible with respect to the number and the positions of the input views. Indeed, it is possible to



(a) Varying the number of anchor views:(i) 4 corner views; (ii) 12 views on the diagonals; (iii) 24 border views.



(b) Anchor views at different positions: (i) 4 anchor views = 4 stereo views; (ii) 4 anchor views = 4 corner views; (iii) 4 views located in the same quadrant; (iv) 4 views randomly selected such that each quadrant contains one anchor view.

Figure 2.4 – Cumulative percentage of pixels below a certain error threshold for different strategies to select anchor views: (a) varying the number of anchor views; (b) with a fixed number (4) of anchor views at different positions.

arbitrarily select a subset of light field views as anchor views.

Fig. 2.4 evaluates the percentage of pixels below a certain error threshold for different strategies to select anchor views. The higher is this percentage, more accurate is the estimation. We consider the 7×7 central views of the light field "Boxes" is studied. Fig. 2.4(a) assesses the impact of the number of anchor views on the final estimation accuracy. The 4 corner views, the 12 views on the diagonals and the 24 views on the border of the light field are respectively used as the subset A of anchor views. We observe that using more anchor views is useful for improving estimation accuracy, though the improvement may be limited. This suggests that when the time consumption or GPU memory is the bottleneck, less anchor views can be exploited without too much degrading the estimation accuracy.

In Fig. 2.4(b), the number of anchor views is fixed to 4, and we evaluate the impact of the anchor view positions on the performance. The assessed sets A of the anchor views can be: (i) 4 stereo views ($A = S$), (ii) 4 corner views, (iii) 4 views located in the same quadrant (the light field can be divided into four quadrants, "northeast", "southeast", "southwest" and "northwest", according to the location with respect to the target view), and (iv) 4 views randomly selected such that each quadrant contains one anchor view.

The strategy (iii) achieves the worst performance, since the 4 views located in the

same quadrant do not contain occlusion information of the other quadrants. The use of the stereo views as anchor views (i) obtains worse performance than (ii), since the geometry information of the stereo views is already exploited in the coarse estimation step. And indeed, for a dense light field such as "Boxes", geometry information of the 3D scene can be mostly recovered from the 4 corner viewpoints.

2.4.3 Results with densely sampled synthetic light fields

We compare our approach with both traditional and learning based state-of-the-art methods for densely sampled light fields. First, 4 reference methods [78, 81, 84, 83] using traditional approaches are considered. The disparity range is discretized for the methods [81, 78, 84]. As suggested in the light field depth estimation challenge held in 2017 LF4CV workshop [127], the number of disparity levels is set to 100 for the method [81] and 256 for the method [78]. For the method [84], the disparity step is set to 0.01, which corresponds to the minimal threshold of bad pixel ratios that we use. Both explicit and implicit discretization operations in [81, 78, 84] need disparity ranges as priors. To estimate the disparity map for the central view, the methods [78, 81] exploit the whole light field containing 49 views. The method in [83] takes four corner views to infer the central view disparity while the method in [84] chooses 5 images in the crosshair with target view in the center. For our framework, we employ the same crosshair pattern as [84] with 4 images serving as stereo and anchor views at the same time.

The upper part of Table 2.1 compares the estimation accuracy obtained with 8 HCI test scenes using different metrics: Mean Square Error (MSE) and Bad Pixel Ratios (BP) with thresholds 0.01, 0.03 and 0.07 (BP represents the percentage of pixels having an error superior to a certain threshold). In the experiment, we consider the central 7×7 sub-aperture images of the light field and estimate the disparity of the central view. The experiment shows that our framework achieves superior performances compared with other methods for most of the scenes both in terms of MSE and BP. In some cases, our framework yields the second best results with a slight difference only with the method ranked first. Compared with the methods in [78, 81] which exploits all the light field views, our method gives better results in spite of the fact that we use only a subset of light field views. In comparison with the other two methods [84, 83] using a subset of light field views, our method is competitive and sometimes wins with a large margin.

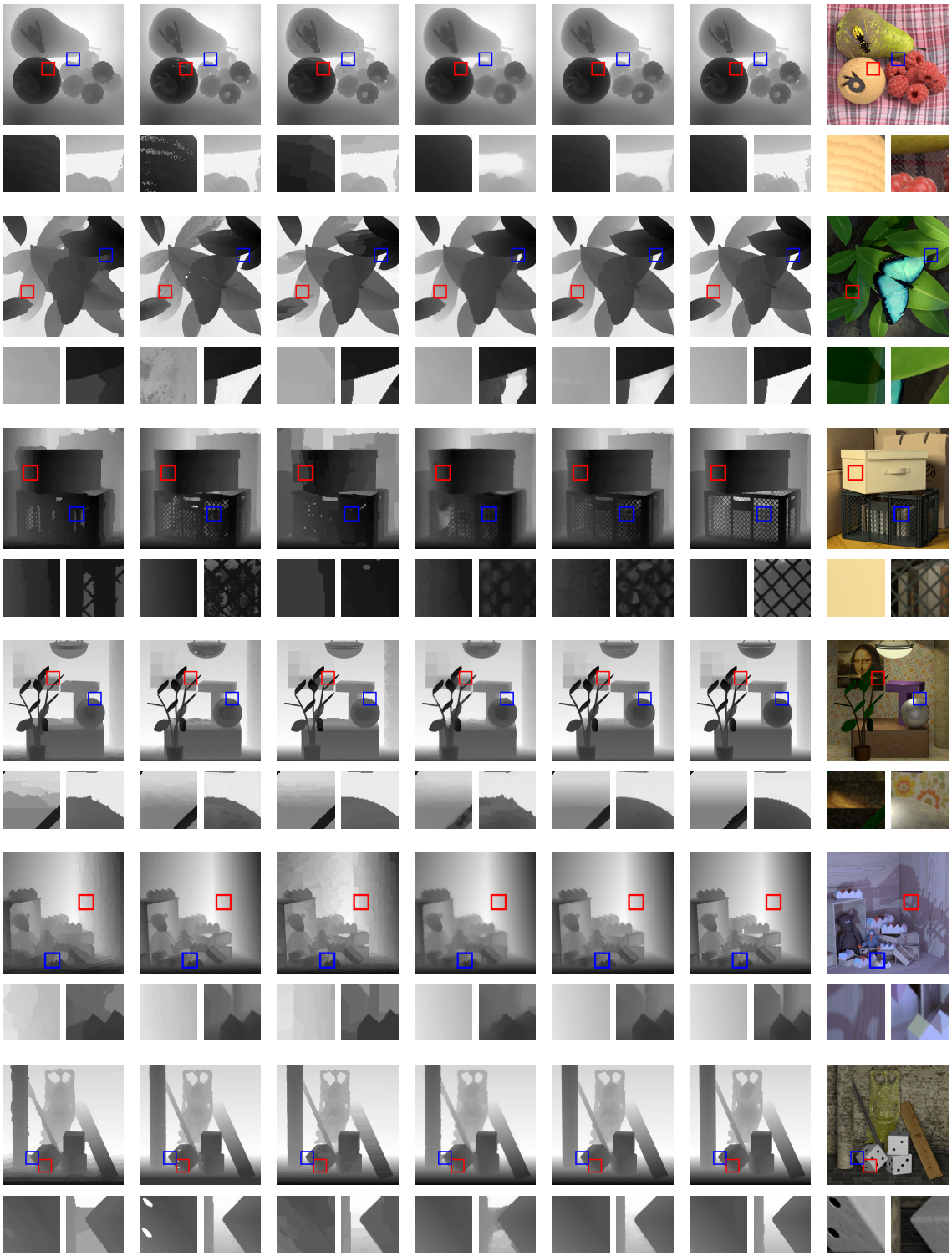
Fig. 2.5 shows the estimated disparity maps for the central view. Readers are recommended to zoom and view these results on the screen for visual comparison. The methods

Light fields	MSE					BP1					BP2					BP3				
	[81]	[78]	[84]	[83]	Ours	[81]	[78]	[84]	[83]	Ours	[81]	[78]	[84]	[83]	Ours	[81]	[78]	[84]	[83]	Ours
Stilllife	2.02	1.72	1.53	2.56	**1.07**	81.2	76.2	76.0	71.3	**70.5**	51.0	32.1	41.0	25.0	**24.8**	20.9	6.8	16.2	9.2	**5.8**
Buddha	1.13	0.97	0.49	0.82	**0.41**	57.7	41.2	52.6	**34.9**	35.3	24.4	14.8	15.0	12.3	**7.7**	10.1	6.7	2.9	5.4	**2.2**
MonasRoom	0.76	0.58	0.66	0.53	**0.39**	46.0	42.5	48.2	**38.6**	39.0	22.1	17.8	20.2	18.6	**13.7**	11.7	7.8	10.4	8.2	**6.1**
Butterfly	4.79	0.74	0.80	1.84	**0.58**	82.5	78.9	83.4	**70.8**	73.4	49.1	48.5	50.9	**36.0**	42.0	15.4	14.1	17.6	6.7	**6.3**
Boxes	14.15	**8.23**	11.30	12.71	9.16	72.7	**62.3**	87.2	65.8	68.4	45.5	**28.1**	65.0	37.7	38.5	26.4	**15.8**	42.0	23.9	22.1
Cotton	9.98	1.44	2.04	1.18	**0.94**	60.5	41.7	75.8	42.6	**38.2**	23.3	11.1	37.5	10.7	**10.6**	8.9	**2.7**	10.4	4.1	3.3
Dino	1.23	**0.29**	0.67	0.88	0.50	76.6	57.5	84.8	49.1	**45.6**	48.4	17.9	57.2	20.0	**14.5**	20.9	**3.4**	24.1	9.5	4.7
Sideboard	4.16	**0.92**	1.34	10.31	1.37	67.8	64.3	78.6	**61.7**	63.6	39.3	31.0	44.1	37.5	**26.3**	23.0	10.4	15.0	19.6	**10.1**
Average	4.78	1.86	2.35	3.85	**1.80**	68.1	58.1	73.3	54.4	**54.3**	37.9	25.2	41.4	24.7	**22.3**	17.2	8.5	17.3	12.1	**7.6**
Furniture	-	-	**0.37**	1.94	0.39	-	-	86.3	41.3	**40.7**	-	-	73.1	41.3	**23.0**	-	-	36.0	20.2	**8.6**
Lion	-	-	0.10	0.87	**0.09**	-	-	**35.9**	73.0	47.6	-	-	23.9	59.5	**9.0**	-	-	5.5	9.5	**2.6**
Toy_bricks	-	-	**0.22**	1.10	0.56	-	-	59.5	66.4	**50.5**	-	-	33.2	44.6	**23.7**	-	-	**4.7**	11.2	12.4
Electro_devices	-	-	0.20	0.63	**0.19**	-	-	76.9	60.7	**52.8**	-	-	57.4	43.4	**30.5**	-	-	22.5	18.6	**8.7**
Average	-	-	**0.22**	1.14	0.31	-	-	64.7	64.7	**47.9**	-	-	46.9	47.2	**21.6**	-	-	17.2	14.9	**8.1**

Table 2.1 – Quantitative comparison with Non-learning-based methods on synthetic light fields datasets. Number of input views: [81]-**49 views**, [78]-**49 views**, [84]-**5 views**, [83]-**4 views**, Ours-**5 views**. For the first 8 scenes (dense LFs), MSE denotes 100*Mean Square Error, BP1, BP2, BP3 denote Bad Pixel Ratios with thresholds 0.01, 0.03, 0.07. For the last 4 scenes (sparse LFs), MSE denotes Mean Square Error, BP1, BP2, BP3 denote Bad Pixel Ratios with thresholds 0.05, 0.1, 0.3.

in [81] and [83] obtain disparity maps with distorted boundaries, while the method in [84] loses precision on slanted surfaces where disparity values gradually vary. In contrast, the methods[78, 84] as well as our framework can estimate disparities with more precise boundaries. Our method, although it may suffer from a subtle smoothness along boundaries, it yields less artifacts within homogeneous and slanted areas, and gives more visually pleasing results.

We also compare our method against two state-of-the-art methods based on deep learning [57, 80]. Both of them exploit epipolar geometry of the light field. The method in [57] retrieves disparity values by exploiting 3D EPI volumes containing texture information on two spatial dimensions and one angular dimension, whereas the method in [80] constructs the input volume by taking views on four different angular directions: horizontal, vertical, left and right diagonals. As these trained models are not publicly available for 7×7 light field views, we re-trained the models following the instructions in the corresponding papers. Since the network in [80] was implemented without zero-padding after each convolutional layer, the resulting disparity map loses 11 pixels at each border. For the sake of comparison, we cropped the same amount of bordering pixels for the method in [57] and our method as well (this explains the slight difference of measurement of our method in Table 2.1 and Table 2.2). Quantitative results are shown in Table 2.2. Our proposed method has several advantages.



(a) *Jeon* (b) *Zhang* (c) *Huang* (d) *Jiang* (e) *Ours* (f) *GT* (g) *Image*

Figure 2.5 – Qualitative comparison with non deep learning-based methods. Each row shows the estimated disparity maps with two zoomed areas (homogeneous area framed in red and contour area framed in blue) for different methods: (a) *Jeon et al.* [81], (b) *Zhang et al.* [78], (c) *Huang* [84], (d) *Jiang et al.* [83], (e) *Our framework*. The (f) *Ground truth* and (g) *Color image* are also shown.

Light fields	MSE			BP1			BP2			BP3		
	[57]	[80]	Ours	[57]	[80]	Ours	[57]	[80]	Ours	[57]	[80]	Ours
Stilllife	3.02	1.96	**1.14**	84.2	77.3	**71.0**	56.2	39.4	**25.8**	22.9	11.5	**6.2**
Buddha	0.52	**0.26**	0.43	75.9	39.9	**35.7**	37.3	**5.2**	8.0	9.1	**1.4**	2.3
MonasRoom	1.06	0.60	**0.41**	76.8	42.5	**39.6**	41.9	14.5	**14.3**	16.2	7.8	**6.4**
Butterfly	1.13	1.41	**0.57**	85.5	84.3	**72.4**	57.2	59.7	**40.2**	22.4	24.1	**6.0**
Boxes	9.06	**5.20**	9.97	82.5	**62.4**	68.4	53.7	**27.4**	39.6	30.5	**15.1**	23.6
Cotton	0.97	**0.25**	0.76	77.8	51.7	**36.6**	42.0	**4.9**	10.0	10.9	**0.9**	2.9
Dino	1.25	**0.19**	0.53	83.6	**41.0**	45.1	54.4	**6.6**	14.6	23.7	**1.9**	5.0
Sideboard	2.33	**0.80**	1.45	82.9	**58.6**	66.0	54.1	**21.0**	27.9	24.6	**6.6**	10.9
Average	2.42	**1.33**	1.91	81.2	57.2	**54.4**	49.6	**22.3**	22.6	20.0	8.7	**7.9**
Furniture	9.18	1.73	**0.42**	96.4	85.1	**40.2**	92.8	71.3	**23.0**	78.8	38.4	**8.9**
Lion	1.59	3.41	**0.09**	95.3	87.4	**48.7**	90.6	76.4	**8.7**	72.9	56.3	**2.6**
Toy_bricks	3.70	**0.36**	0.57	96.0	85.1	**49.5**	92.0	70.6	**23.3**	76.5	29.6	**12.6**
Electro_dev	7.82	0.74	**0.20**	95.0	80.3	**51.5**	89.9	60.6	**29.2**	72.0	22.8	**8.9**
Average	5.57	1.56	**0.32**	95.7	84.5	**47.5**	91.3	69.7	**21.1**	75.0	36.8	**8.3**

Table 2.2 – Quantitative comparison with Learning-based methods on synthetic light fields datasets. Number of input views: [57]-**7 views (dense), 3 views (sparse)**, [80]-**25 views (dense), 9 views (sparse)**, Ours-**5 views**, MSE and BP are kept as same as those in Table 2.1.

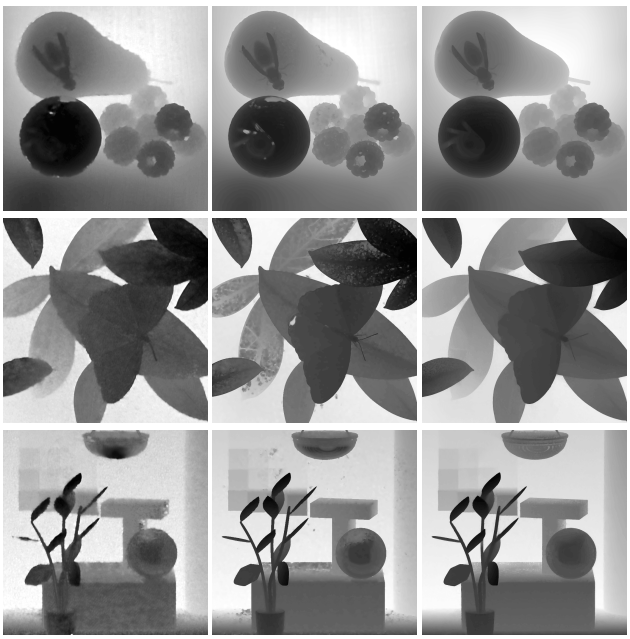
1/- With less input views (the number of input views for the methods [57], [80] and our method are respectively 7, 25 and 5), our method outperforms [57] and achieves competitive results against [80].

2/- Our method benefiting from the WTA fusion and the CNN based refinement is more robust and generates less artifacts for some light field scenes (c.f. Fig. 2.6).

3/- As [57], our method can predict disparity at each viewpoint, whereas the method [80] only predicts for the center view.

4/- Compared to EPI based learning frameworks, training our model is less demanding. Indeed, it is required in [80] to manually mark out and exclude all the reflection, refraction and textureless regions when preparing training data, which can be very time-consuming with a large dataset.

As learning-based methods may fail when characteristics in the input images differ from that in the training dataset, we have tested the robustness of our framework by adding Gaussian noise to the test light fields in comparison with two other learning-based methods [57, 80]. Fig. 2.7(a) shows the averaged MSE and Bad Pixel Ratio (threshold 0.07) over 8 synthetic LFs as a function of the standard deviation of the Gaussian noise. When increasing the standard variance of the noise, the performances of all the reference methods degrade, while the quality of depth maps estimated with the proposed framework remains more stable. To explain this difference in terms of robustness, we show in Fig. 2.7(b) 2.7(c),

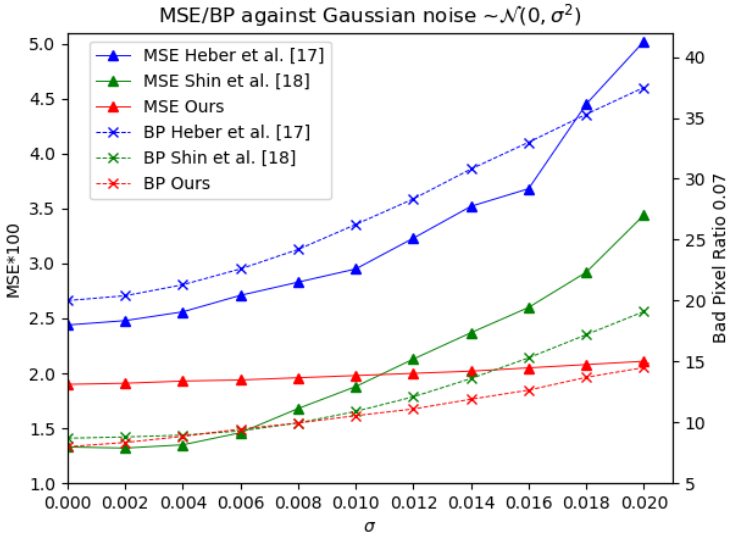


(a) *Heber et al.* [57] (b) *Shin et al.* [80] (c) *Ours*

Figure 2.6 – Qualitative comparison to deep learning-based methods, with methods (a) *Heber et al.* [57], (b) *Shin et al.* [80], (c) *Ours*.

a clean and a polluted EPI. The added Gaussian noise destroys the geometric structures in the EPI that are used by methods like [57, 80] for depth estimation, while our framework exploits spatial information of each sub-aperture image, hence stays more robust against noise. Apart from the above experimental results on synthetic data, Fig. 2.8 also shows disparity maps estimated from light fields captured by plenoptic cameras, hence that are prone to noise and distortions. Fig. 2.8 shows that our framework can still estimate satisfying disparity maps.

A general flexibility and complexity comparison is summarized in Table 2.3 for all the compared methods. The proposed framework can adapt to light fields with wide baselines and, unlike methods using plane sweep volumes, does not require a discretized depth range at the input. Furthermore, some methods may be limited by their specific viewpoints selection pattern and cannot be used for estimated disparity maps for views located at the border of the light field. In contrast, the proposed approach uses a flexible stereo and anchor view selection pattern that allows us to estimate disparity maps for all light field views. In terms of computational cost, our framework takes less than 2 seconds to estimate one disparity map, that is much faster than traditional methods, but a bit slower than the two learning-based methods [57, 80]. Note that the implementations of



(a) Evolution of the MSE and BP (threshold = 0.07) measures obtained with three learning-based estimation methods when increasing the standard deviation of Gaussian noise added to the input light fields $\sim \mathcal{N}(0, \sigma^2)$.

(b) EPI without Gaussian noise

(c) EPI with additive Gaussian noise $\sim \mathcal{N}(0, 0.02^2)$

Figure 2.7 – (a) Impact of noise on the quality (in terms of MSE and BP) of estimated depth maps. (b) EPI from scene *stilllife* without Gaussian noise. (c) Same EPI but with Gaussian noise.

Property	[81]	[78]	[84]	[83]	[57]	[80]	Ours
Adaptability to wide baselines	×	×	√	√	×	×	√
Estimation for any view	×	×	√	√	√	×	√
Without disparity discretization	×	×	×	√	√	√	√
Computational cost (one view)	960s	>1h	127s	16s	0.04s	0.52s	1.93s

Table 2.3 – Flexibility Comparison

the methods [83, 57] estimate disparity maps for all the views in a light field or for views on one row at one time, consequently we divided their costs by the number of estimated views. Since the codes of four traditional methods are not available for GPU, they are tested on an Intel i7 CPU with 16G RAM, whereas the three learning-based methods are tested on a NVIDIA Tesla P100 GPU with 16G memory.

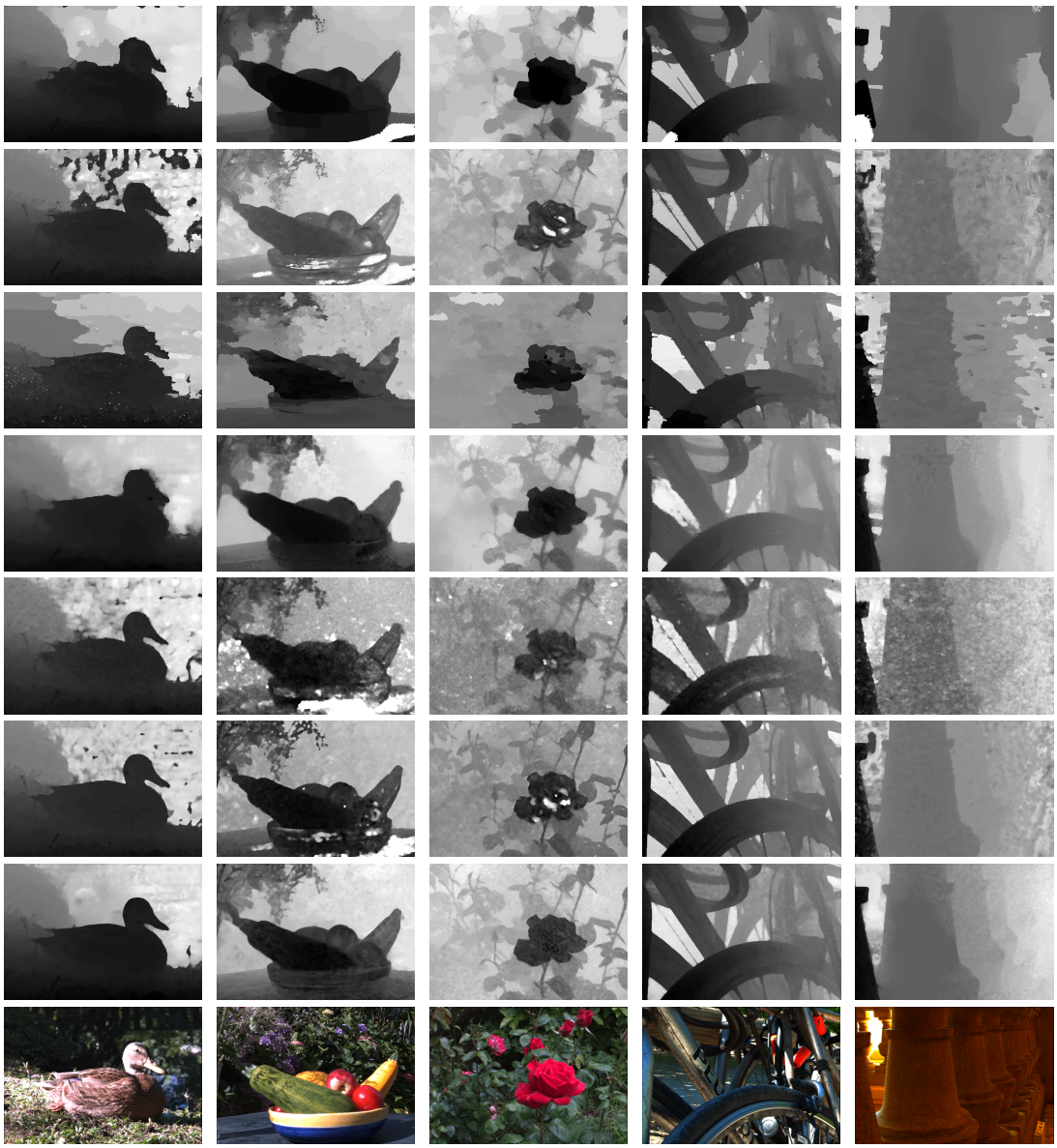


Figure 2.8 – Qualitative comparisons for estimated disparity maps. The first three light fields are from the INRIA Illum Dataset [128], and the last two light fields are from the EPFL Illum dataset [52]. From top to bottom, figures show the disparity maps estimated with methods in *Jeon et al.* [81], *Zhang et al.* [78], *Huang* [84], *Jiang et al.* [83], *Heber et al.* [57], *Shin et al.* [80] and our proposed method. The final row shows the central views of the light fields.

Processing step	*FN2*	*FN2-ft-stereo*	Fusion	Refinement
MSE	10.94	2.27	2.00	1.91
BP	90.5	63.6	56.2	54.4

Table 2.4 – Contribution of each block in framework. MSE denotes 100*Mean Square Error, BP denotes Bad Pixel Ratios with threshold 0.01. The values for *FN2* and *FN2-ft-stereo* are averaged values of the estimations between 4 stereo views and the target view.

Additionally, Table 2.4 gives the contribution of each building block to the performance of the proposed framework. Thanks to our new dataset and our new finetuning strategy, *FN2-ft-stereo* significantly improves the accuracy of the estimated disparity maps compared with the original *FN2* model. By taking into account multiple anchor views at different viewpoints (the occluded regions for these views are unlikely all overlapping), the disparity fusion step is able to cope with errors in occluded regions. And the refinement aims at coping with disparity discontinuities that may be introduced by the fusion step. However, due to the fact that the occluded regions have a much smaller pixel number compared to the whole image, the fusion and refinement steps bring less quantitative improvement on the entire disparity map than the stereo estimation step.

2.4.4 Results with sparsely sampled synthetic light fields

Among the state-of-the-art approaches mentioned above, the methods in [78] and [81] derive the disparity estimate from EPI analysis, thus are hardly applicable for sparse light fields with large baselines. For sparsely sampled light fields, we compare our framework with the methods in [83, 84, 57, 80], using the objective metrics: Mean Square Error (MSE) and Bad Pixel Ratios with larger thresholds 0.05, 0.1, 0.3. We consider the central 3×3 views of the light field, and the evaluation is performed for the estimated disparity of the central view. The step length in the method in [84] is set to 0.05, corresponding to the minimal Bad pixel ratio threshold. Both the input view number in [57] and the stream length in [80] are set to be 3. The rest of the setup is identical to the experiments for dense light fields. The lower parts of Table 2.1 and 2.2 show that our framework yields better Bad Pixel Ratios with large margins when compared with other methods. In terms of MSE, our method ranks the second, slightly lagging behind [84]. The first two rows in Fig. 2.9 show disparity maps estimated with different methods. Compared with the first two non deep learning-based methods, our algorithm functions well in both contours and homogeneous zones. Although our method gives smoother boundaries than those with

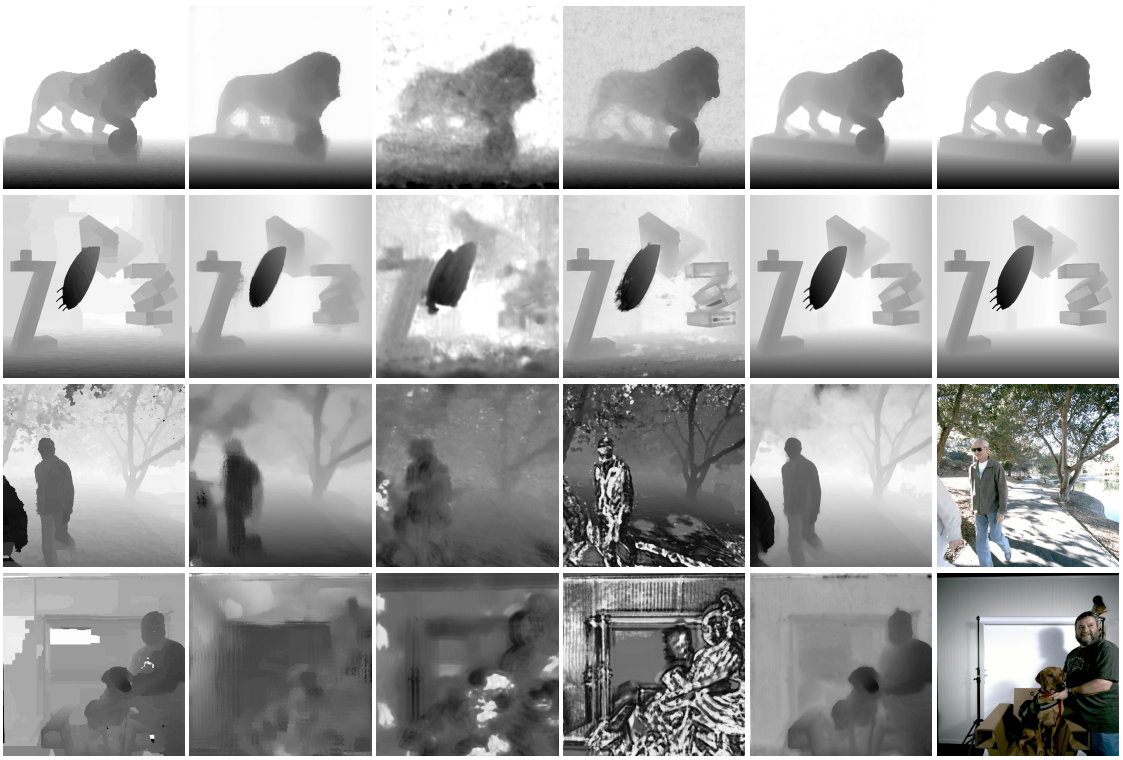


Figure 2.9 – Visual comparison for estimated disparity maps for sparse light fields. The first two rows are obtained with synthetic data, while the last two rows are obtained for real-world data.

[84], it works well in slanted zones where the method in [84] tends to fail. For the two deep learning-based methods [57, 80], even though we have re-trained the corresponding models with our sparse light field data, the results contain severe deformations and artifacts, since epipolar line continuity is no longer guaranteed with sparse light fields.

2.4.5 Results with real light fields

To assess the disparity estimation performance of our network with real light fields, we consider both dense light fields (7×7 central views are considered) captured with the Lytro Illum camera [128, 52] and sparse light fields (3×3 central views are considered) captured with camera arrays [60]. Because of the lack of ground truth disparity values, the prior disparity range required by these methods is set using our estimation results, with a margin of 10%: $[d_{min} - 0.05(d_{max} - d_{min}), d_{max} + 0.05(d_{max} - d_{min})]$. Fig. 2.8 shows the estimated disparity maps for dense light fields using different methods. Two sparse light fields are tested in Fig. 2.9. Compared with other methods, our framework gives

more accurate estimates on object boundaries, especially for the scenes which have more texture details. Among all results, our disparity maps have less artifacts in spite of the fact that relatively less views are used for the estimation. Note that our network has been trained with only noise-free synthetic light fields. An additional fine-tuning with noisy images could still improve the estimation accuracy of our network for real light fields.

Overall, regardless their limitations in terms of view number and view selection, the methods in [81, 83] show inaccuracies at the boundaries, while methods in [78, 84] can give relatively more accurate estimates at the boundaries. The method in [78] often contains artifacts in homogeneous regions, while the method in [84] fails on slanted surfaces. Our experiments with noisy light fields have shown that the performance of the learning-based methods [57] and [80] depends on the quality of EPI. On the contrary, our method can estimate accurate depth maps both at the boundaries of objects, and in homogeneous and slanted regions with an acceptable computational cost. It has also been show to be more robust to noise than the other two learning-based methods.

2.5 Conclusion

In this chapter, we have proposed a learning based approach to estimate disparity maps between all light field views. The algorithm takes a variable subset of input views and estimates accurate disparity maps for both densely and sparsely sampled LF data.

The proposed framework starts with a stereo estimation step which takes a flexible number of light field views to give first disparity maps estimates. A fusion step then aggregates these disparity maps into a single one by conducting pixel-wise selection based on the warping error. A multiscale residual refinement step is then used to eliminate noise and improve spatial coherence. In order to train the model so that it can apply to both sparsely and densely sampled light fields, we have also created two synthetic light fields datasets with different disparity ranges. To our knowledge, this is the first publicly available dataset for sparsely sampled synthetic light fields given together with ground truth disparity maps for all the views.

The effectiveness of our algorithm has been demonstrated with both synthetic and real light fields datasets, in comparison with several state-of-the-art reference methods. The proposed algorithm outperforms state-of-the-art algorithms despite of the use of less input views. It is robust in both homogeneous areas and along the contours, as well as in slanted zones. Experimental results with real light fields show that our algorithm estimates consistent objects boundaries, and preserves details in the scene, although the network

has been only trained using synthetic data.

Chapter 3

View synthesis in light field

3.1 Introduction

In the previous chapter, we saw how to estimate scene geometry from a subset of views. We now move beyond the geometry estimation and focus on view synthesis (or light field angular super-resolution) problem based on estimated disparities.

The problems of enhancing the light field angular resolution can be tackled from different perspectives, i.e. as a problem of light field reconstruction from a subset of views using signal priors (e.g. sparsity in the continuous 4D Fourier domain [105]), of angular super-resolution [77, 107] or of view synthesis. While Image Based Rendering (IBR) techniques have been predominant over the past years in the field of view synthesis (see e.g. [60, 129]), this field has significantly evolved thanks to the emergence of learning-based approaches.

In this chapter, we propose a novel learning-based framework to synthesize light field views from a sparse set of input views. We design an end-to-end learning framework combining two reconstruction strategies, one in the pixel domain and the other in the feature space. A CNN is first used to estimate disparity from the input views. Using the estimated disparity, we project the input color views and their features to the target view position. The features are extracted using the lower layers of the VGG19 classification network[130]. For the pixel-wise reconstruction, occlusions are explicitly handled by a disparity-dependent interpolation filter, and the target view is predicted based on warped views using convolutional layers. For the feature-based reconstruction, multi-scale features at the target viewpoint are successively reconstructed based on warped features from the input views, and the reconstructed view is inferred from the feature maps of the finest scale. Finally, a soft mask is learned to merge the results of the pixel-wise and feature-based reconstructions. The entire Fused Pixel and Feature-based Reconstruction network (FPFR) is trained in an end-to-end fashion.

Experimental results with both synthetic and real-world light fields, with a large range

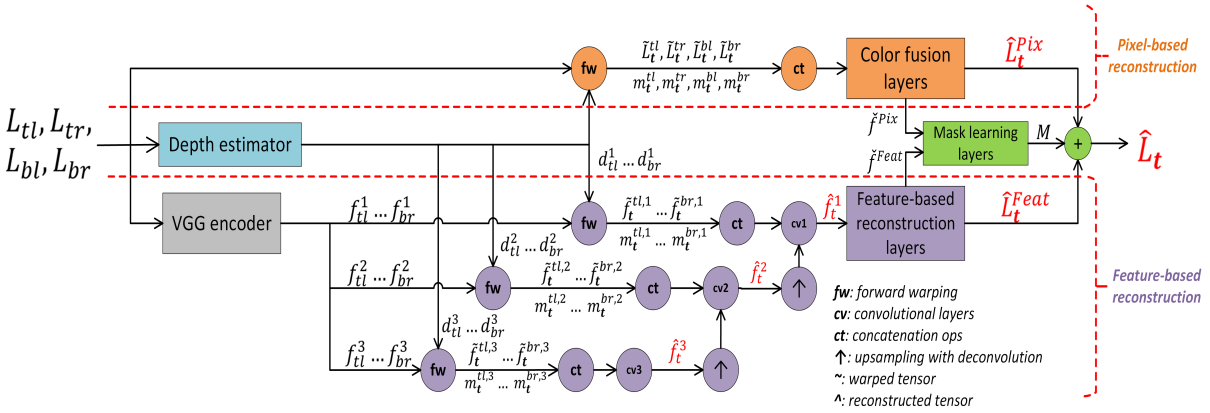


Figure 3.1 – Overview of our end-to-end framework. Given the corner views $\{L_{tl}, L_{tr}, L_{bl}, L_{br}\}$ as input, the depth estimator (blue) predicts depths $\{d_{tl}, d_{tr}, d_{bl}, d_{br}\}$. PixRNet (orange) reconstructs the target view $\hat{L}_{\mathbf{t}}^{\text{Pix}}$ based on the warped views $\{\tilde{L}_{\mathbf{t}}^{tl}, \tilde{L}_{\mathbf{t}}^{tr}, \tilde{L}_{\mathbf{t}}^{bl}, \tilde{L}_{\mathbf{t}}^{br}\}$. Parallelly, FeatRNet (purple) infers the features of the target view based on warped multi-scale input features, extracted from bottom layers of VGG network. The view $\hat{L}_{\mathbf{t}}^{\text{Feat}}$ is then reconstructed. Finally, in the FusNet (green), $\hat{L}_{\mathbf{t}}^{\text{Pix}}$ and $\hat{L}_{\mathbf{t}}^{\text{Feat}}$ are merged using a soft mask M, which is learnt from $\check{f}^{\text{Pix}}$ and $\check{f}^{\text{Feat}}$, denoting input feature maps of the last layer of PixRNet and FeatRNet, respectively.

of disparities between input views, demonstrate that our proposed framework significantly outperforms the state-of-the-art methods. Our approach gives excellent reconstruction quality on fine textures and object boundaries, despite the fact that intermediate disparity outputs may not be accurate and consistent across input views. Furthermore, we show that our network can also achieve competitive performances for light field view extrapolation without additional training.

3.2 Methodology

3.2.1 Architecture overview

We focus on synthesizing a novel view $L_{\mathbf{t}}$ at the target angular position $\mathbf{t} = (u_{\mathbf{t}}, v_{\mathbf{t}})$ in a light field, using a given set of input views $\{L_{\mathbf{i_1}}, ..., L_{\mathbf{i_N}}\}$, where $\mathbf{i} = (u_{\mathbf{i}}, v_{\mathbf{i}})$ is the angular position of source view. More specifically, in our work, we take a sparsely sampled 2×2 views as inputs. For the sake of convenience, these input views are also denoted by L_{tl} (top left), L_{tr} (top right), L_{bl} (bottom left) and L_{br} (bottom right) according to their relative position. We also simplify disparity notation $d_{\mathbf{i}}^{\mathbf{t}}$, used in Chapter 2.2.1, denoting disparity between two views $L_{\mathbf{i}}$ and $L_{\mathbf{t}}$ normalized by the distance between the two views, as $d_{\mathbf{i}}$, accordingly, normalized disparity maps of four input views are becoming $d_{tl}, d_{tr}, d_{bl}, d_{br}$.

Fig. 3.1 shows our learning-based framework. First, a lightweight disparity estimator module (in blue) estimates one disparity map for each input view. Two parallel reconstruction schemes are then applied. The target view is synthesized either by PixRNet (Pixel-wise Reconstruction Network) or FeatRNet (Feature-based Reconstruction Network). For the pixel-wise scheme, given the disparity estimates, the input views are first projected to the target position by applying forward warping. PixRNet takes as input the projected views as well as their occlusion masks to synthesize the novel view. PixRNet provides accurate pixel values in lowly textured areas. However, due to inconsistencies between disparity values estimated for the different input views, a simple fusion in the pixel space of the projected views may lead to blurriness in highly textured areas as well as on object contours. FeatRNet is thus designed to compensate for this drawback. In FeatRNet, the reconstruction is based on low-level features inferred for the novel viewpoint. Extracted from lower layers of VGG19-Net [130], the features of the input views are warped to the target position at different resolution scales to generate the corresponding target view features, from which the decoder reconstructs the color view. Finally, a learned soft combination mask obtained by FusNet (Fusion network) merges the outputs of both PixRNet and FeatRNet.

3.2.2 Disparity estimation

Given the input view set $\{L_{tl}, L_{tr}, L_{bl}, L_{br}\}$ from a rectified and structured light field, two disparity maps can be computed at each input viewpoint. Let us take the view on the top left L_{tl} as an example, disparity can be computed either between the horizontal pair (L_{tl}, L_{tr}) or between the vertical pair (L_{tl}, L_{bl}) as

$$d_1 = \mathrm{FNet}(L_{tl}, L_{tr})_h, \tag{3.1}$$

$$d_2 = \mathcal{R}^{-1} \circ \mathrm{FNet}(\mathcal{R}(L_{tl}), \mathcal{R}(L_{bl}))_h. \tag{3.2}$$

FNet is a convolutional neural network which estimates an optical flow between two stereo views, the subscript h denotes taking horizontal component of the optical flow and normalizing it by their angular distance. In this work, we employ a pre-trained PWC-Net [90] model which is then finetuned by light field image pairs on the same row. The symbols $\mathcal{R}(\cdot)$ and $\mathcal{R}^{-1}(\cdot)$ are counterclockwise and clockwise rotation of 90°, which enables to treat vertical pairs in the same way as horizontal ones.

Both d_1 and d_2 may contain estimation errors due to color inconsistencies or occlusions.

We further improve disparity accuracy by merging two disparity maps d_1 and d_2 into a single one d_{tl}. More specifically, based on each disparity map, other three views L_{tr}, L_{bl} and L_{br} are projected to the position of L_{tl} and the corresponding warping errors e_1 (for d_1) and e_2 (for d_2) are computed as:

$$e_k = \sum_{\mathbf{i}\in\{tr,bl,br\}} \sum_{RGB} |L_{tl} - \tilde{L}^k_{\mathbf{i}\to tl}|, k = 1, 2, \tag{3.3}$$

where $\tilde{L}^k_{\mathbf{i}\to tl}$ represents image warped from current position $\mathbf{i}$ to top left corner using disparity d_k. The fusion of d_1 and d_2 is performed via value selection for each pixel $\mathbf{p}$:

$$k' = \arg\min_k e_k(\mathbf{p}), d_{tl}(\mathbf{p}) = d_{k'}(\mathbf{p}). \tag{3.4}$$

Similarly, we can obtain disparity maps d_{tr}, d_{bl} and d_{br} for the other three views.

This part of the work has been inspired from our previous work in Ch. 2 and its variant [91], however, we made several modification to adapt it into our view synthesis pipeline:

1-) The flow estimator used in our previous work and in [91], which is a time and computation resource-consuming netwokr FlowNet2.0 [86], here, we adopt a lightweight PWC-Net architecture to save more resources for other modules, making end-to-end training the whole pipeline feasible.

2-) In our previous work and in [91], a single disparity for each view is obtained by taking both minimum and average warping errors into account (details found in Ch. 2.2.3), and this operation is computationally heavy and underivable, we hence simplify in this work the disparity fusion step by merely considering warping error minimization.

3.2.3 Pixel-wise reconstruction

The pixel-wise reconstruction module (PixRNet) follows the conventional DIBR approach to generate novel views. In particular, based on the estimated disparity maps, input views are warped to the target position and then fused to generate the final view. Similar design can be found in [1]. Apart from the fact that [1] first infers the disparity map at the target position and then employs backward projection, and our scheme applies forward projection, the main advantage of our scheme is the use of disparity-dependent interpolation which handles occlusion.

Interpolation with occlusion handling

Let us project the pixel $\mathbf{p} = (x_p, y_p)$ from the input viewpoint $\mathbf{i}$ to the target viewpoint $\mathbf{t}$, at a position with non-integer coordinates $\tilde{\mathbf{p}} = (x_{\tilde{p}}, y_{\tilde{p}})$, with the disparity value $d_{\mathbf{i}}(\mathbf{p})$:

$$\tilde{\mathbf{p}} = \mathbf{p} + (\mathbf{t} - \mathbf{i}) d_{\mathbf{i}}(\mathbf{p}). \tag{3.5}$$

The pixel value $\widetilde{L}_{\mathbf{t}}(\mathbf{q})$ at integer coordinates $\mathbf{q} = (x_q, y_q)$ is interpolated from nearby values $L_{\mathbf{i}}(\mathbf{p})$ as

$$\widetilde{L}_{\mathbf{t}}(\mathbf{q}) = \frac{\sum_{\mathbf{p}} L_{\mathbf{i}}(\mathbf{p}) W(\mathbf{p}, \mathbf{q})}{\sum_{\mathbf{p}} W(\mathbf{p}, \mathbf{q}) + \epsilon}, \tag{3.6}$$

where ϵ is a very small value for numeric stability.

The computation of the weights $W(\mathbf{p}, \mathbf{q})$ is crucial for end-to-end learning performance. Three concerns should be addressed:

1/) The weight computation should be differentiable.

2/) As in traditional interpolation, the distance separating two pixels should be reflected in the weight.

3/) Occlusions should be handled.

Therefore, we propose

$$W(\mathbf{p}, \mathbf{q}) = w_D(\mathbf{p}, \mathbf{q}) w_O(\mathbf{p}), \tag{3.7}$$

with w_D being a coordinate distance metrics

$$w_D(\mathbf{p}, \mathbf{q}) = l(x_{\widetilde{p}}, x_q) l(y_{\widetilde{p}}, y_q), \tag{3.8}$$

where

$$l(x_1, x_2) = \begin{cases} (1 - |x_1 - x_2|), & \text{if } |x_1 - x_2| < 1; \\ 0, & \text{otherwise,} \end{cases} \tag{3.9}$$

and w_O a term handling occlusions defined as

$$w_O(\mathbf{p}) = \exp(-\lambda d^*_{\mathbf{i}}(\mathbf{p})). \tag{3.10}$$

The disparity map $d_{\mathbf{i}}$ is normalized between 0 and 1 to become $d^*_{\mathbf{i}}$. By taking the exponential function, w_d gives more importance to the foreground pixels (small normalized disparity values) rather than background ones (large normalized disparity values). Dis-

parity normalization also avoids weight saturation at large disparity values.

Disocclusion handling

We concatenate thereby four warped views $\{\widetilde{L}_{\mathbf{t}}^{tl}, \widetilde{L}_{\mathbf{t}}^{tr}, \widetilde{L}_{\mathbf{t}}^{bl}, \widetilde{L}_{\mathbf{t}}^{br}\}$ and the corresponding disocclusion masks $\{m_{\mathbf{t}}^{tl}, m_{\mathbf{t}}^{tr}, m_{\mathbf{t}}^{bl}, m_{\mathbf{t}}^{br}\}$. The detection of the disocclusion mask is straightforward with forward warping, which identifies spatial positions having no projected pixels in their neighborhood, i.e. Eq.(3.8) equals to zero for all $\mathbf{p}$. The inpainting on the disoccluded areas is then handled by a small network of 4 convolutional layers to obtain the reconstructed view $\hat{L}_t^{\text{Pix}}$. The loss function is computed as the mean of absolute differences (MAD) between the reconstructed view and the ground truth:

$$\mathcal{L}_1 = \text{MAD}(\hat{L}_{\mathbf{t}}^{\text{Pix}}, L_{\mathbf{t}}). \tag{3.11}$$

3.2.4 Feature-based reconstruction

Due to inconsistencies between disparity estimates for the different input views, a simple fusion of the projected views in the pixel space may lead to blurriness in highly textured zones as well as on object contours. Thus, we propose the feature-based reconstruction module (FeatRNet) as a complementary module of PixRNet.

For each input view $L_{\mathbf{i}}$, we extract low-level features:

$$\forall L_{\mathbf{i}} \in \mathcal{I}, \{f_{\mathbf{i}}^1, f_{\mathbf{i}}^2, f_{\mathbf{i}}^3\} = FeatExt(L_{\mathbf{i}}), \tag{3.12}$$

with $FeatExt(\cdot)$ being the operator that extracts features from the layers *relu1_2,relu2_2* and *relu3_4* of a pre-trained VGG19-Net, and $f_{\mathbf{i}}^s$ being feature volumes at scale s (the resolution of the feature maps in $f_{\mathbf{i}}^{s+1}$ is half of that in $f_{\mathbf{i}}^s$). These features are then warped to the target position in a similar manner as described in Ch. 3.2.3 for the pixels:

$$\forall \mathbf{i}, \forall s, \{\widetilde{f}_{\mathbf{t}}^{\mathbf{i},s}, m_{\mathbf{t}}^{\mathbf{i},s}\} = \text{Warp}(f_{\mathbf{i}}^s, \mathbf{t}). \tag{3.13}$$

The warped features are input to convolutional layers to infer a feature volume of the target view at each scale s:

$$\hat{f}_{\mathbf{t}}^s = \begin{cases} \text{Conv}(\{\widetilde{f}_{\mathbf{t}}^{\mathbf{i},s}, m_{\mathbf{t}}^{\mathbf{i},s}, \forall \mathbf{i}\}), & \text{if } s = 3; \\ \text{Conv}(\{\widetilde{f}_{\mathbf{t}}^{\mathbf{i},s}, m_{\mathbf{t}}^{\mathbf{i},s}, \forall \mathbf{i}\}, \uparrow \hat{f}_{\mathbf{t}}^{s+1}), & \text{if } s = 1, 2. \end{cases} \tag{3.14}$$

At scale s=1 and 2, the inferred features at the previous scale $s+1$ are upsampled by 2 and fed into the network as well. The upsampling operator $\uparrow$ is implemented by a deconvolution layer. Finally, the target view $\hat{L}_{\mathbf{t}}^{\text{Feat}}$ is reconstructed based on features at the finest scale $\hat{f}_{\mathbf{t}}^{1}$. The reconstruction is supervised both in color and feature space by computing

$$\mathcal{L}_2 = \text{MAD}(\hat{L}_t^{\text{Feat}}, L_t) + \sum_{s=1}^{3} \gamma_i \text{MAD}(\hat{f}_{\mathbf{t}}^{s}, f_{\mathbf{t}}^{s}), \tag{3.15}$$

where the second term in Eq.(3.15) represents the difference between the inferred features and those of the ground truth target view.

Note that the use of features has been exploited in recent works for light field reconstruction. However, in most of these works [37, 102], the reconstruction is supervised by a perceptual loss minimizing the distance between features computed on the reconstructed view and those of the reference view. Here, we propose instead a bottom-up approach. We first compute the target view features by warping the warped source view VGG features. The target view is then inferred from the generated target view features. This is motivated by the intuition that VGG features optimized for object recognition can be good texture generative models.

3.2.5 End-to-end learning with fusion

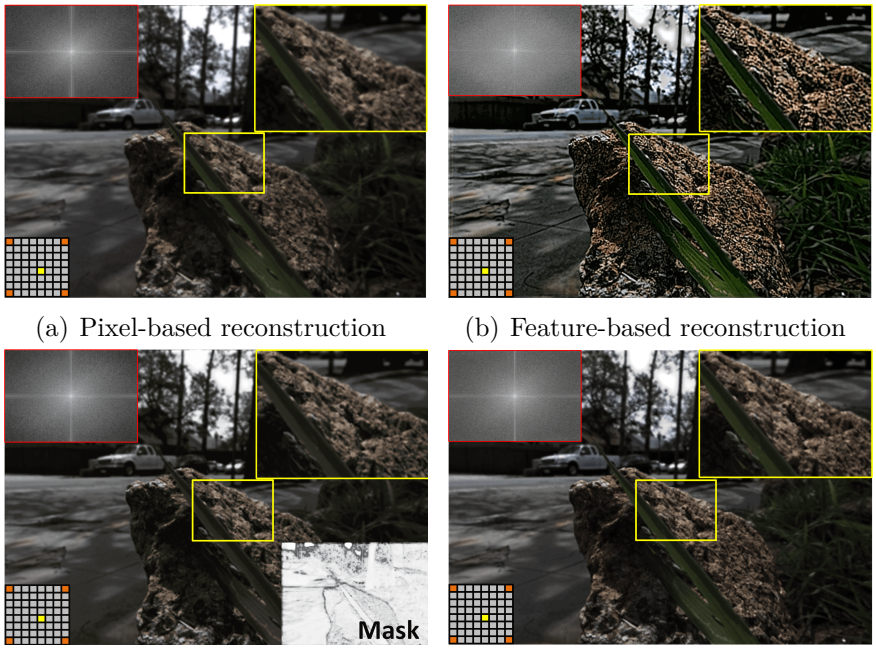
End-to-end learning is finally performed including a fusion module to merge $\hat{L}_{\mathbf{t}}^{\text{Pix}}$ and $\hat{L}_{\mathbf{t}}^{\text{Feat}}$, making the final reconstruction $\hat{L}_{\mathbf{t}}$ well perform in both highly textured and textureless areas. Based on input feature maps of the last layer of PixRNet and FeatRNet, the FusNet learns a soft mask M with values between 0 and 1 (forced by sigmoid activation), which minimizes the pixel-wise reconstruction error:

$$\mathcal{L} = \text{MAD}(\hat{L}_{\mathbf{t}}, L_{\mathbf{t}}), \tag{3.16}$$

with

$$\hat{L}_{\mathbf{t}} = M\hat{L}_{\mathbf{t}}^{\text{Pix}} + (1-M)\hat{L}_{\mathbf{t}}^{\text{Feat}}. \tag{3.17}$$

We demonstrate in Fig. 3.2 the outputs at different stages of our framework and their corresponding Fourier domains, it is obvious that image reconstructed from pixel cues is more blurry that the one reconstructed from feature cues, when observing frequency domain of each image, the energy of pixel-based reconstruction is concentrating in low frequency domain, while the energy of feature-based reconstruction scatter over all fre-



(a) Pixel-based reconstruction (b) Feature-based reconstruction

(c) Fused view (d) Reference view

Figure 3.2 – Visualization of the outputs (both in pixel domain and frequency domain) at different stages of our framework. (a) View reconstructed with pixel cue. (b) View reconstructed with feature cue. (c) View obtained by fusing pixel and feature-based reconstructions. (d) Ground truth view.

quencies. Final fused view takes advantage of both, both its Fourier domain and pixel domain are closer to the ground truth view. Finally, we offer in Table 3.1 the architectures of both PixRNet, FeatRNet and FusNet. The architecture of PWC-Net can be found in [90].

3.3 Training details

3.3.1 Training data preparation

We employ 78 synthetic light fields in INRIA synthetic dataset [131] and 16 light fields in 'Additional' category of new HCI dataset [55] to form our synthetic training set. All light fields are with resolution $512 \times 512 \times 9 \times 9$. We also use 100 scenes proposed in UCSD Illum dataset [1], which are captured by a Lytro Illum camera with resolution $376 \times 541 \times 14 \times 14$, by extracting the 8×8 central views of each scene to form our real-world training set. Since real-world data contains complicated lighting conditions and artifacts caused by hardware limitations, making it distinct from synthetic data,

PixRNet	***k***	***s***	***in/out***	***input***
conv3Px	3	1	16/64	$concat(\{\widetilde{L}^{\mathbf{i}}_{\mathbf{t}}, m^{\mathbf{i}}_{\mathbf{t}}, \forall \mathbf{i}\})$
conv2Px	3	1	64/32	*conv3Px*
conv1Px	3	1	32/16	*conv2Px*
$\hat{L}^{Pix}_{\mathbf{t}}$	3	1	16/3	*conv1Px*
FeatRNet				
conv3_1Ft	3	1	1028/256	$concat(\{\widetilde{f}^{\mathbf{i},3}_{\mathbf{t}}, m^{\mathbf{i},3}_{\mathbf{t}}, \forall \mathbf{i}\})$
conv3_2Ft	3	1	256/256	*conv3_1Ft*
$\hat{f}^{3}_{\mathbf{t}}$	3	1	256/256	*conv3_2Ft*
up3	4	2	256/32	$\hat{f}^{3}_{\mathbf{t}}$
conv2_1Ft	3	1	548/128	$concat(\{\widetilde{f}^{\mathbf{i},2}_{\mathbf{t}}, m^{\mathbf{i},2}_{\mathbf{t}}, \forall \mathbf{i}\}, up3)$
conv2_2Ft	3	1	128/128	*conv2_1Ft*
$\hat{f}^{2}_{\mathbf{t}}$	3	1	128/128	*conv2_2Ft*
up2	4	2	128/16	$\hat{f}^{2}_{\mathbf{t}}$
conv1_1Ft	3	1	276/64	$concat(\{\widetilde{f}^{\mathbf{i},1}_{\mathbf{t}}, m^{\mathbf{i},1}_{\mathbf{t}}, \forall \mathbf{i}\}, up2)$
conv1_2Ft	3	1	64/64	*conv1_1Ft*
$\hat{f}^{1}_{\mathbf{t}}$	3	1	64/64	*conv1_2Ft*
conv0_1Ft	3	1	64/32	$\hat{f}^{1}_{\mathbf{t}}$
conv0_2Ft	3	1	32/32	*conv0_1Ft*
$\hat{L}^{Feat}_{\mathbf{t}}$	3	1	32/3	*conv0_2Ft*
FusNet				
$\breve{f}^{Pix}$	3	1	16/8	*conv1Px*
$\breve{f}^{Feat}$	3	1	32/8	*conv0_2Ft*
conv3_Fn	3	1	16/8	$concat(\breve{f}^{Pix}, \breve{f}^{Feat})$
conv2_Fn	3	1	8/8	*conv3_Fn*
conv1_Fn	3	1	8/4	*conv2_Fn*
M	3	1	4/1	*conv1_Fn*

Table 3.1 – The proposed network architecture. k, s and *in/out* represent the kernel size, the stride and the number of input/output channels, whereas '*up*' and '*concat*' represent upsampling by deconvolution and concatenation.

we therefore prepare two models, one trained on synthetic data and the other further finetuned on real-world data. As light fields from DLFD and SLFD in INRIA synthetic dataset and those in UCSD Illum dataset captured by a Lytro Illum camera, are with different baselines, we therefore adopt different configurations when extracting views: for light fields in DLFD that have narrow baseline, four corner views are selected by an *angular distance* corresponding to view index difference $l \in \{5, 6, 7, 8\}$. For light fields in SLFD having wide baseline, corner views are extracted with $l \in \{2, 3\}$. When using the real-world training set,the corresponding angular distance is set to be $l \in \{6, 7\}$. The ground truth view is randomly selected within the square defined by corner views. We

work on randomly cropped light field patches of size 160×160 and the batch size is 5.

3.3.2 Data augmentation

We also apply geometrical and chromatic transformations to increase the variety of the training data. The chromatic transformation is carried out by changing the contrast, color balance, brightness and gamma value. The contrast is sampled within the interval $[-0.8, 0.4]$, the color balance is adjusted with a multiplicative factor sampled within $[0.5, 2]$, the brightness is modified by adding a value drawn according to a Gaussian distribution with $\sigma = 0.2$, and the gamma value is sampled within the interval $[0.7, 1.5]$. For the geometrical transform, we randomly rotate the image by 0°, 90°, 180° or 270° counterclockwise and flip it on the left and right.

3.3.3 Training schedule

End-to-end learning from scratch for such a network containing multiple modules can be intractable. In order to make sure that each module converges well and the final view inference is correctly learned, we follow a specific training schedule. We first finetune a pre-trained PWC-Net with stereo pairs of light field views to make it adapt to disparity estimation. Then, PixRNet and FeatRNet are trained separately using the loss functions of Eq.(3.11) and Eq.(3.15) respectively. At this stage, the weights of PWC-Net are fixed for two reasons. The first is to accelerate the model convergence. The second reason is that, to reduce the size of the complete model, we constrain the two reconstruction schemes to use the same disparity. Finally, an end-to-end training including PWC-Net, PixRNet and FeatRNet is performed. Note that at this final stage, as our purpose is to minimize the pixel-wise reconstruction error (Eq.(3.16)), the training is no longer supervised by the feature-level reconstruction errors.

The model is trained with a fixed learning rate of 0.00001 and the hyperparameters are set as $\lambda = 10, \gamma_1 = 1/64, \gamma_2 = 1/32, \gamma_3 = 1/4$. The training takes approximately 5 days on a GPU Tesla V100 with 32GB of memory. Our work is implemented with the *tensorflow* package.

LFs	Disparity range	DeepVS	Soft3D	LLFF	EPI	PurePix	PureFeat	FPFR	FPFR*
mona†	[-5,5] (10)	38.90	40.92	41.20	37.54	39.90	40.35	42.47	**42.86**
butterfly†	[-6,8] (14)	40.68	42.75	41.35	39.61	41.64	39.33	42.69	**42.96**
buddha†	[-10,6] (16)	41.08	41.86	40.66	40.05	41.24	40.99	42.78	**43.06**
cotton⋆	[-9,9] (18)	47.24	**48.95**	47.07	47.97	48.18	46.45	48.58	48.76
boxes⋆	[-7,13] (20)	33.64	32.14	**34.97**	31.65	33.44	33.90	33.86	34.46
dino⋆	[-10,10] (20)	38.41	41.69	41.26	38.44	40.63	39.38	42.66	**42.98**
sideboard⋆	[-10,12] (22)	30.91	30.23	**32.33**	27.30	29.43	30.79	31.85	32.18
Toy_bricks◇	[-1,22] (23)	28.90	36.58	37.98	31.46	36.55	36.01	38.84	**39.35**
Elec_devices◇	[-10,17] (27)	34.09	36.24	36.76	31.55	35.53	34.87	37.63	**38.03**
stilllife†	[-16,16] (32)	26.29	34.73	32.73	32.02	33.96	32.77	36.39	**37.05**
Lion◇	[-5,29] (34)	28.05	35.18	35.22	33.91	35.10	34.99	35.47	**35.59**
Two_vases◇	[-5,39] (44)	25.65	32.49	35.82	29.09	33.46	34.78	35.56	**35.99**
Sculpture◇	[-26,34] (60)	22.31	29.15	29.68	26.22	28.56	29.51	30.09	**30.30**
Bear◇	[-38,53] (91)	18.36	28.00	33.22	23.40	29.00	32.64	31.87	**33.84**
Average	-	32.49	36.49	36.43	33.59	36.19	36.20	37.92	**38.39**

Table 3.2 – Quantitative results (PSNR) for the reconstructed central view on the synthetic test light fields. The corresponding datasets are indicated by symbols: ⋆ [55], ◇ [131] and † [54]. With the best performance being bold and the second ones being underlined.

3.4 Experiments

3.4.1 Results with synthetic light fields

We evaluate our framework with test light fields from both HCI datasets [55, 54] and INRIA synthetic dataset [131]. Our approach is compared against four state-of-the-art view synthesis methods for light fields which represent well the recent trend in the domain: conventional DIBR in a deep learning framework (DeepVS [1]), synthesis via multi-layer scene representation with learning (LLFF [37]) or without learning (Soft3D [60]), and view interpolation based on learned EPI structure (EPI [98]).

All reference methods except LLFF take 2×2 corner views to generate intermediate views. The released model of LLFF requires at least 5 input views. Thus, for comparison purposes, LLFF is tested with 2×2 views together with a fifth view which is the horizontal immediate neighbor to the view L_{tl}. As the performance of learning-based methods can be highly impacted by the training data, for a fair comparison, all the pre-trained models are finetuned with the same datasets that our model is trained on. We also replace the estimated camera poses required by LLFF with ground truth ones, in order to make sure that the error is merely due to the reconstruction pipeline.

Table 3.2 compares the reconstruction quality of the center view in terms of PSNR. The vanilla version of our model is named FPFR. FPFR* refers to 'test-time augmentation' of our model: the test scene is rotated and flipped before being processed by the network,

Methods	DeepVS[1]	LLFF[37]	EPI[98]	FPFR	FPFR*
Runtime	778s	212s	1332s(CPU)	172s	684s

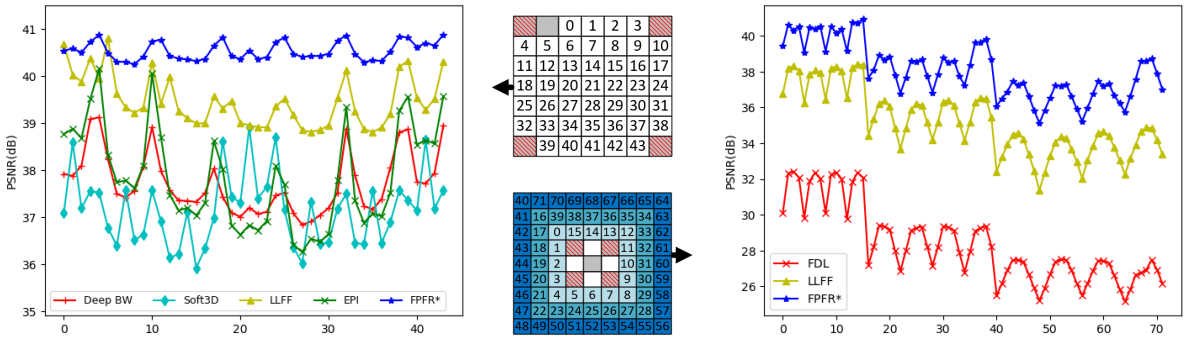
Table 3.3 – Runtime for generating a $768 \times 768 \times 7 \times 7$ light field for learning-based methods.

and the final reconstructed image is computed as the average of eight reconstructed views (after inverse rotation and flipping operation) based on different versions of the same scene. Note that for comparison purposes, PixRNet and FeatRNet can be trained separately with the disparity estimation module to become two fully independent view synthesis models on their own. We denote these two models 'PurePix' and 'PureFeat', which are also trained in an end-to-end fashion. Test light fields are arranged according to their disparity range (from dense to sparse). On average, our method (FPFR and FPFR*) significantly outperforms other methods: a gain of nearly 2dB is observed against the best reference method. Our method performs especially well for highly textured scenes e.g. *stilllife*. Reconstruction error maps are shown in Fig. 3.4. One can observe that our method generates less error on the object contours and thin structures in textured regions (e.g. the tablecloth in *stilllife* and the wallpaper in *sideboard*).

Furthermore, Fig. 3.3(a) shows that FPFR consistently generates high quality views across different viewpoints, whereas the reconstruction quality decreases for other methods when the target view is distant from the input views.

Note that the single mode schemes PurePix and PureFeat also obtain competitive results in Table 3.2. In the same vein as DeepVS [1], PurePix is on average 3.7dB better than DeepVS, especially for sparse scenes. We believe that it is mainly due to the occlusion and disocclusion handling in our PixRNet design.

We compare the runtime of the three learning-based reference methods[1, 37, 98] against ours. In Table 3.3, we record the running time for synthesizing a light field of resolution $768 \times 768 \times 7 \times 7$ on a GPU of type Nvidia Tesla V-100 with 32GB memory. (The method [98] is run on CPU Intel i7-7600U 2.80GHz 16GB RAM. We did not succeed in running it on GPU, and in their paper the running time comparison was also done with CPU.) The running time includes model loading and computation of color views. Our proposed framework, when compared with other learning-based methods, still outperforms others in terms of computational complexity. Although runtime has increased by several times due to a test-time augmentation application, it is an acceptable price when considering the PSNR gains it brings.



(a) PSNR for **interpolated** viewpoints (b) View indices (c) PSNR for **extrapolated** viewpoints

Figure 3.3 – Average PSNR over 8 synthetic scenes ([55, 54]) for each novel viewpoint. (a) Interpolation. (c) Extrapolation. (b) View indices for interpolation and extrapolation. 4 input views (red slash) are used for FPFR, DeepVS[1], EPI[98] or FDL[132], whereas 5 input views (grey) are used for LLFF[37].

LFs	DeepVS[1]	Soft3D[60]	LLFF[37]	EPI[98]	FPFR
Cars	31.53	27.68	29.06	28.17	**32.05**
Flower1	33.13	30.29	30.00	30.44	**34.19**
Flower2	31.95	30.52	28.90	29.26	**33.80**
Rock	34.32	32.67	32.60	32.46	**36.48**
Leaves	27.97	27.34	27.74	26.48	**32.27**
Seahorse	32.03	30.41	28.50	26.62	**34.68**
Average	31.82	29.82	29.47	28.90	**33.91**

Table 3.4 – Quantitative results (PSNR) for the reconstructed view (5,5) on the real-world data (8×8 views) [1]. With the best performance being bold and the second ones being underlined.

3.4.2 Results with real light fields

For real-world experiments, we use the same training and test sets as used in [1] (UCSD Illum dataset, which is introduced in Chaptre 1.1.5). For a fair comparison, all learning-based models are finetuned with the same dataset. Table 3.4 shows that our approach achieves the highest PSNRs. Note that a gain of 4.3dB is obtained for the scene *leaves*. As for synthetic data, similar observations can be made in Fig. 3.4 for real-world scenes: we obtain more accurate contours and well preserved textures.

3.4.3 Ablation study

Pixel vs. feature

To put in evidence the different tasks the pixel-wise and feature-based reconstruction carry out, in Fig. 3.5 we show examples of feature maps taken from the last layers in

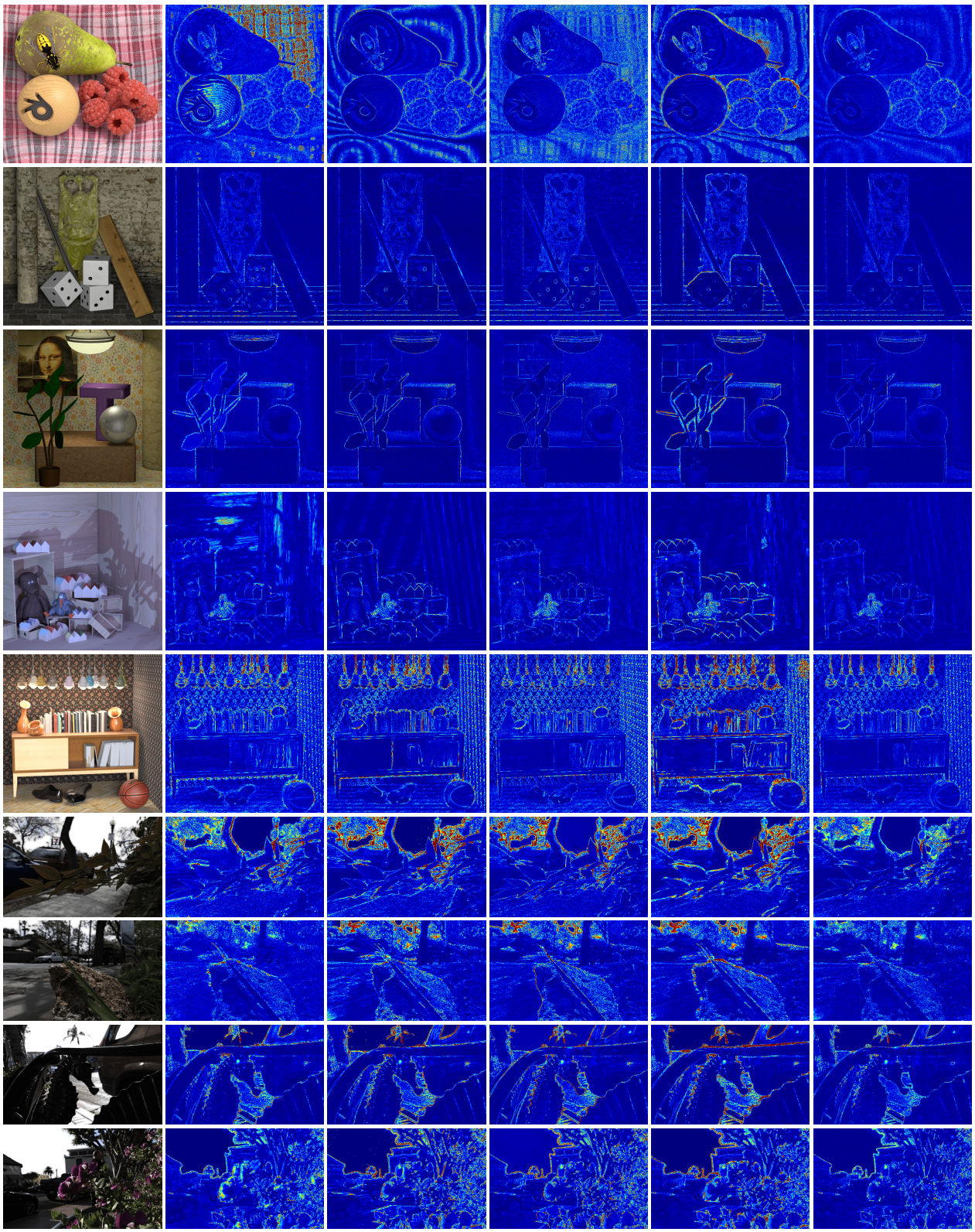


Figure 3.4 – Visual comparison of reconstruction error maps for different methods. With larger error value in red, and smaller one in blue.

Figure 3.5 – Feature maps taken from the last latent layers of PixRNet and FeatRNet.

PixRNet and FeatRNet, from which the color view is reconstructed. We observe highly enhanced textures and clear line structures in FeatRNet feature maps, whereas PixRNet feature maps can provide information such as brightness, color and contrast, etc.

Feature-based reconstruction vs. perceptual loss

A common practice to optimize the view reconstruction quality via the feature space is to apply the so-called perceptual loss [133] when performing end-to-end learning. To valid our concept of merging the pixel-wise reconstruction with the feature-based one, we compare FPFR with the single mode scheme PurePix learned end-to-end with a perceptual loss. In experiments, we observe that FPFR is about 0.7dB better.

Fusion vs. single mode

With Fig. 3.6, we first analyse the convergence behavior at different stages of the network (the output of PixRNet $\hat{L}_{\mathbf{t}}^{\text{Pix}}$ in cyan, that of FeatRNet $\hat{L}_{\mathbf{t}}^{\text{Feat}}$ in red, and the final output $\hat{L}_{\mathbf{t}}$ in blue). The fusion pushes each mode to excel in its domain: the image $\hat{L}_{\mathbf{t}}^{\text{Pix}}$ is more accurate on colors and low frequencies, whereas $\hat{L}_{\mathbf{t}}^{\text{Feat}}$ contains a higher level of texture than the reference image (this explains the clear degradation of the PSNR for $\hat{L}_{\mathbf{t}}^{\text{Feat}}$ during training). Therefore, the final image $\hat{L}_{\mathbf{t}}$ (blue curve) obtains better quality both in low and high frequencies (see Fig. 3.2). In Fig. 3.6, we also compare FPFR with single mode schemes PurePix (in green) and PureFeat (in yellow). A clear advantage is observed for FPFR.

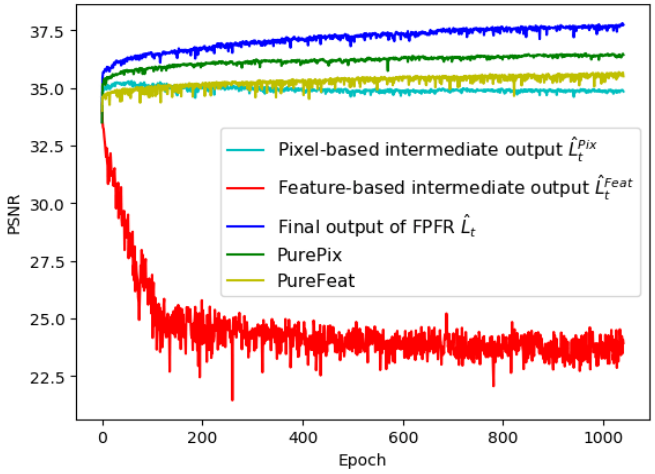


Figure 3.6 – Learning curves of different end-to-end schemes (FPFR, PurePix and PureFeat). For FPFR, the curves for intermediate outputs ($\hat{L}_{\mathbf{t}}^{\text{Pix}}$ and $\hat{L}_{\mathbf{t}}^{\text{Feat}}$) are also shown.

Single-scale vs. multi-scale

A single-scale architecture ($s = 1$) for feature-based reconstruction is compared with its multi-scale counterparts ($s = 1, 2, 3$). In experiments, a gain of about 0.5dB is observed in favor of the multi-scale architecture.

3.4.4 Extrapolation

Extrapolation of light fields with plausible disocclusions can be a more difficult task than interpolation, since less information of the target view is known in the input views. In Fig. 3.3(c), we evaluate the inherent capacity of our framework to extrapolate against two reference methods FDL[132] and LLFF [37]. As input, our method and FDL take 4 corner views (red slashes in Fig. 3.3(b)) of a subset of 3×3 views with narrow baseline. Since LLFF requires at least 5 views, the central view is also included in input views (grey in Fig. 3.3(b)). The output is an extended light field with 9×9 views ($4\times$ baseline). One can observe that our method outperforms reference methods by a large margin. Wider the extended baseline, more important is our gain. Note that both LLFF and our model are trained for interpolation tasks, here we evaluate the inherent capacity to extrapolate without any further training.

3.5 Conclusion

In this chapter, we have presented a novel learning-based model for light field view synthesis. In order to obtain high quality reconstruction both in low and high frequencies, end-to-end learning is performed including a pixel-wise reconstruction module and a feature-based reconstruction module. Experiments have demonstrated that the proposed model achieves state-of-the-art performance both for synthetic and real-world light fields.

Chapter 4

From angular dimension to temporal dimension

4.1 Introduction

In the previous chapter, we have concentrated on view synthesis (or angular SR) problem in light field and proposed a novel learning-based architecture capable of reconstructing novels views from a given subset of input views to finally increase angular sampling rate. If considering this undersampling problem in a broader perspective, we can find light field isn't the only visual content that suffers from perception degradation caused by undersampling. One typical example is classical video sequence, where a small video frame rate can lead to jerky motion. Despite the fact that light field and video are essentially two different types of media content, they both sample a scene in similar manners: a light field records the scene by capturing several angular views at a certain instant, i.e. sampling a scene in spatial and angular dimensions. While a video records scene through frames captured at a series of consecutive instants but at each instant from a single viewpoint, which means sampling a scene in spatial and temporal dimensions. The similarity of sampling a scene motivates us to generalize our previous workflow dedicated for light field angular upsampling to video temporal upsampling problem (from angular dimension to temporal dimension). Moreover, in this generalization process, we found that some network designs used in Ch. 3 can be further improved to gain a better view synthesis performance. Therefore, in this chapter, we mainly focus on two aspects: the improvement of light field view synthesis (LFVS) pipeline, and adaptation from angular dimension to temporal dimension for video frame interpolation (VFI) task.

In summary, our novel pipeline extends previous FPFR architecture presented in Ch. 3 in several ways:

1.) While the weights in the FPFR depth estimator are pre-trained using ground truth optical flows, here we instead pre-train the weights of the optical flow estimator to

be optimized for the view synthesis problem or the temporal frame interpolation problem, without using ground truth optical flows or disparity maps. The pre-training of the flow estimator for the target tasks make the disparity maps or the optical flows better adapted for the targeted reconstruction problem and in addition reduces the time needed for end-to-end training.

2.) We replace the fixed VGG19 [130] encoder by a lighter and trainable one. Instead of using a feature extractor trained for image classification, the proposed simpler trainable encoder allows us to extract deep features better suited for the following reconstruction steps.

3.) The introduction of residual connections in both pixel-based and feature-based synthesis convolutional modules is shown to improve the quality for light field view synthesis.

This novel architecture denoted as FPFR+ not only gives synthesized light fields views with a higher quality compared with FPFR, and outperforms state-of-the art methods for both dense and sparse light fields, but it can also be used for video frame rate conversion with high quality results. Comprehensive and intensive experiments show that the proposed architecture produces high quality results in terms of quantitative and qualitative measures, for both light field view synthesis and video frame interpolation compared with the state-of-the-art methods. We indeed show that the quality of interpolated frames obtained with the proposed method is quite competitive compared with the one obtained with reference methods specifically dedicated to the temporal frame interpolation problem.

4.2 Methodology

4.2.1 Architecture overview

In the following we consider both video sequences and light fields. Although in previous chapters, we adopt $L_{\mathbf{i}}$, with $\mathbf{i} = (u_0, v_0)$ to denote a view in light field. In this chapter, for the sake of simplicity, we unify the notation for both video frame and light field views, by using I_t to represent a frame captured at instant t, and using $I_{\mathbf{i}}$ to represent angular view at position $\mathbf{i} = (u_0, v_0)$.

Our proposed architecture is an extension of work in Ch. 3, hence it can naturally be used to reconstruct a densely sampled light field from a subset of input views. Moreover,

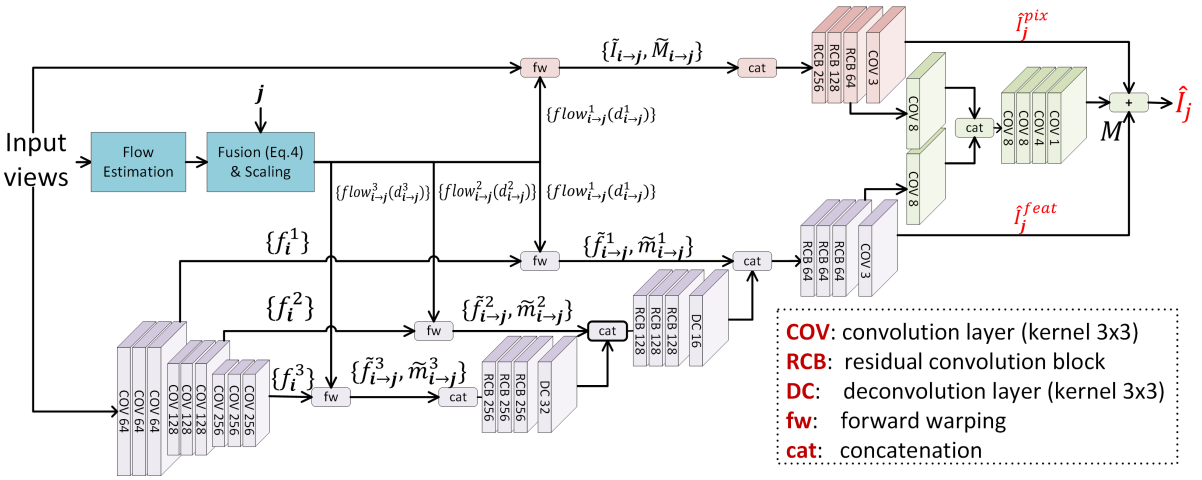


Figure 4.1 – Overview of the proposed deep architecture for both light field view synthesis and video frame temporal interpolation. In the case of the light field view synthesis problem, the input views are 4 sparse light field views (e.g. corner views) with $\mathbf{i} = \{tl, tr, bl, br\}$ being the index of the source view positions, and $\mathbf{j}$ the index of the target view position. In the case of video frame temporal interpolation, the input views are temporally adjacent views, with $\mathbf{i} = \{0, 1\}$ the index of source frame time instants, and $\mathbf{j} = t$ the target frame instant.

we show here that it can also be used to reconstruct a video sequence from a subset of input frames. Like explained in the introduction, we made several modifications based on previous architecture. The novel architecture is depicted in Fig 4.1. For both problems, it is composed of the same three modules like in Ch. 3.2.1. The first module Flow estimation (blue) which will compute disparity maps for the angular view synthesis problem and optical flows for the frame temporal interpolation problem. The flow estimation module is followed by a pixel-based view synthesis module, PixRNet (orange), and by a feature-based view synthesis module, called FeatRNet (violet). The outputs of these two synthesis modules are then merged using a learned soft fusion mask generated by a network called FusNet (green).

Depending on the targeted problem, the application of the above architecture varies only in terms of input images and in the semantic of the estimated flows. In the case of light field angular view synthesis, the inputs of the architecture are 2×2 light field views $\{I_{tl}, I_{tr}, I_{bl}, I_{br}\}$. In the case of video frame temporal interpolation, the inputs of the architecture are two temporally adjacent frames I_0 and I_1 and the architecture produces temporally intermediate frames. Depending on the targeted problem, the flows estimated by the Flow estimation module are disparity maps in the case of light field angular view synthesis, whereas are they are bi-directional optical flows in the case of temporal frame interpolation.

4.2.2 Flow estimation

The Flow estimation module predicts disparity maps for angular view synthesis or bidirectional motions for temporal interpolation.

Disparity maps

In the case of light field view synthesis, given four corner views $\{I_{tl}, I_{tr}, I_{bl}, I_{br}\}$, we adopt the same workflow as in Ch. 3.2.2 to estimate a single disparity map for each input view. Please refer to Ch. 3.2.2 for details. However, despite the fact that the same state-of-the-art flow estimation network PWC-Net [90] are employed as flow estimator FNet, we utilize a novel finetuning strategy that optimizes its weights for the targeted view synthesis or temporal interpolation tasks without using any ground truth optical flows. The training strategy hence differs from the one in Ch. 3, where we finetune PWC-Net with ground truth disparity maps between light field image pair, More details will be presented in Ch. 4.2.3.

Bidirectional optical flows

In the case of temporal interpolation task, the flow estimation module takes two input frames I_0 and I_1 and output a forward flow (from instant 0 to 1) and a backward flow (from instant 1 to 0):

$$flow_{0\to1} = \text{FNet}(I_0, I_1), \tag{4.1}$$

$$flow_{1\to0} = \text{FNet}(I_1, I_0). \tag{4.2}$$

The estimated optical flows have horizontal and vertical components. Moreover, as there is only one estimated motion for each view, further refinement by fusion is not required. Both flows are used for forward and backwarp warping when performing the temporal interpolation, as explained later in Ch. 4.2.3.

4.2.3 View warping and fusion

Based on the estimated optical flows or disparity maps, the input views and their feature maps are warped onto the target view position $\mathbf{j}$ via a differentiable forward warping operation while computing at the same time binary occlusion masks. The warped images and feature maps are respectively fed into the PixRNet and FeatRNet synthesis modules

along with the occlusion masks to synthesize two views $\hat{I}_{\mathbf{j}}^{pix}$ and $\hat{I}_{\mathbf{j}}^{feat}$. Viewing $\hat{I}_{\mathbf{j}}^{pix}$ and $\hat{I}_{\mathbf{j}}^{feat}$ have complementary properties in homogeneous and textured regions, finally, the FusNet module learns a soft mask M to merge $\hat{I}_{\mathbf{j}}^{pix}$ and $\hat{I}_{\mathbf{j}}^{feat}$ into an image $\hat{I}_{\mathbf{j}}$ which well reconstructs both homogeneous and textured regions.

Image forward warping

Backward warping (BW) has been a widely used operation for various image processing tasks, e.g. image classification [134], depth image based rendering (DIBR) [1, 37] etc. It consists in projecting pixels from one source view to a target one, using the optical flow associated to the target view, hence requires the optical flow at the target position. In Ch. 3.2.3, we adopt a forward projection (or warping) operation, which does not require flow estimation at the target position, and can moreover better handle occlusions via disparity-dependent interpolation. As corresponding operation details for light field view synthesis have already been presented in Ch. 3.2.3, we thus explain in this part how image forward warping and interpolation are conducted in the case of video frame interpolation.

Let us assume that we would like to project a pixel $\mathbf{p} = (x_p, y_p)$ from a source frame to a target frame. Given optical flow $flow_{0\to1}$ between source frames I_0, I_1 and the target intermediate instant t in the case of video frame interpolation, the non-integer coordinates of the projected pixel $\tilde{\mathbf{p}} = (x_{\tilde{p}}, y_{\tilde{p}})$ can be computed as:

$$\tilde{\mathbf{p}} = \mathbf{p} + t \times flow_{0\to1}(\mathbf{p}) \tag{4.3}$$

where the term $t \times flow_{0\to1}$ approximates the flow $flow_{0\to t}$ between the frame I_0 and the frame at time instant t, which is then used to warp pixels of frame I_0. The frame I_1 is similarly warped using $flow_{1\to t} = (1-t) \times flow_{1\to0}$.

The pixel value $\tilde{I}_{0\to t}(\mathbf{q})$ at integer coordinates $\mathbf{q} = (x_q, y_q)$ of the warped image will be interpolated as a weighted sum of nearby source pixels or $I_0(\mathbf{p})$:

$$\widetilde{I}_{0\to t}(\mathbf{q}) = \frac{\sum_{\mathbf{p}} I_0(\mathbf{p}) W(\mathbf{p},\mathbf{q})}{\sum_{\mathbf{p}} W(\mathbf{p},\mathbf{q}) + \epsilon}, \tag{4.4}$$

with

$$W(\mathbf{p},\mathbf{q}) = w_D(\mathbf{p},\mathbf{q}) w_O(\mathbf{p}), \tag{4.5}$$

an interpolation weight term as the product of a distance term w_D and an overlap handling term w_O. Readers can refer to Eq. 3.8 and Eq. 3.9 for the computation of w_D.

The term w_O is used for handling overlapping warped pixels, since pixels from foreground and background are often overlaid in occluded regions. In the light field view synthesis problem, we use the disparity maps $d_{\mathbf{i}}$ of each input view to compute this weight (Eq. 3.10). However, disparity maps are not available for the video frame interpolation problem, and since scene depth is difficult to estimate from a monocular video, we use the warping error for handling overlapping pixels.

More precisely, in the case of video frame interpolation, we use brightness error e_0^* to calculate w_O as

$$w_O(\mathbf{p}) = exp(-\lambda e_0^*(\mathbf{p})), \tag{4.6}$$

The exponential function gives more importance to pixels having smaller brightness error and less importance to those with larger brightness error. The brightness error e_0 of I_0 is calculated as

$$e_0 = \sum_{RBG} |I_0 - \mathcal{BW}(I_1, flow_{0\to1})|, \tag{4.7}$$

where I_0 and I_1 are the two input frames and $\mathcal{BW}$ denotes the backward warping operation.

Non occupied pixel positions on the target view after pixel warping are indicated as disoccluded positions via a binary mask $\tilde{M}$. However, to detect disoccluded pixels in a differentiable manner, and to handle interpolation in disoccluded positions, we also warp the weight map w_O. For each warped position, we thus have RGB and weight values. If there is no pixels in the neighbourhood, then w_D is 0 in all neighbourhood positions, indicating, in a differentiable manner, that the corresponding pixel position is a disoccluded pixel. The warped weights are also used to compute the interpolation weights.

Here, by taking forward warping for both light field view synthesis and video frame interpolation cases into account, we can summarize FW operation as follows:

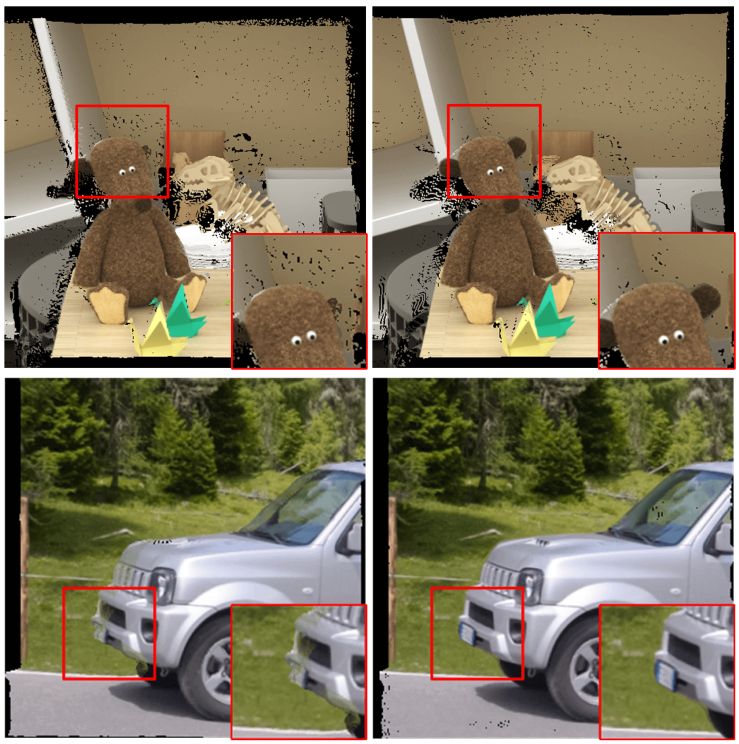
$$\tilde{I}_{0\to t}, \tilde{M}_{0\to t} = \mathcal{FW}(I_0, flow_{0\to t}, e_0) \tag{4.8}$$

for VFI, or

$$\tilde{I}_{\mathbf{i}\to\mathbf{j}}, \tilde{M}_{\mathbf{i}\to\mathbf{j}} = \mathcal{FW}(I_{\mathbf{i}}, d_{\mathbf{i}}, \mathbf{i}, \mathbf{j}) \tag{4.9}$$

for LFVS.

Both disparity and brightness error are effective cues in handling pixel overlaps, as the larger is the brightness error of a pixel, or the larger is its disparity value, the more likely it will be overlaid after warping. Similar brightness error-based image warping operation,



(a).Warped view before FT (b).Warped view after FT

Figure 4.2 – Visual comparison of views before & after finetuning (FT). The views in the 1st row are from the INRIA synthetic dataset [131], while those in the 2nd row are from the video frame interpolation dataset in [135]

called 'softmax splatting', can be found in [119].

Finetuning the flow estimation module

Depth-dependent LFVS approaches such as in [60] and our previous work, or flow-based VFI schemes such as in [118, 117], using an off-the-shelf disparity or optical flow estimation module, both rely on a good initialization of that module whether or not end-to-end finetuning is conducted later. However, most of these concerned networks are trained on synthetic datasets with ground truth disparities [56, 89] or optical flows [136, 86], and are not fully adapted to the view synthesis problem. Here, based on our aforementioned FW operation, we propose instead a simple but effective finetuning procedure that enables us to finetune the initial flow estimation module under view synthesis or frame-interpolation supervision, i.e. without any ground truth disparity maps or optical flows.

More specifically, in the case of LFVS, given four input views I_{tl}, I_{tr}, I_{bl}, I_{br} and

corresponding disparity maps d_{tl}, d_{tr}, d_{bl}, d_{br} estimated by Flow estimation module, we aim at synthesizing the view at $\mathbf{j}$. We finetune the estimated flows, without using ground truth flows, by optimizing

$$argmin_\theta \sum_{\mathbf{i}\in\{tl,tr,bl,br\}} |(\tilde{I}_{\mathbf{i}\to\mathbf{j}} - I_{\mathbf{j}})\tilde{M}_{\mathbf{i}\to\mathbf{j}}|_1 \tag{4.10}$$

where θ is the set of parameters of the FNet module, and $\tilde{I}_{\mathbf{i}\to\mathbf{j}}$, $\tilde{M}_{\mathbf{i}\to\mathbf{j}}$ are obtained by the FW operation in Eq. 4.9.

In the case of VFI, given the input frames I_0, I_1 and the associated estimated flows $flow_{0\to1}$, $flow_{1\to0}$, we interpolate a frame at an intermediate instant t, by optimizing

$$argmin_\theta \sum_{t'\in\{0,1\}} |(\tilde{I}_{t'\to t} - I_t)\tilde{M}_{t'\to t}|_1, \tag{4.11}$$

where $\tilde{I}_{t'\to t}$, $\tilde{M}_{t'\to t}$ are obtained via Eq. 4.8, with $flow_{0\to t} = t \times flow_{0\to1}$ and $flow_{1\to t} = (1-t) \times flow_{1\to0}$.

Fig. 4.2 illustrates the warped images obtained before and after finetuning. Before finetuning, we can observe severe deformations in regions like bear's ear and licence plate that are challenging for the synthesis process. These artifacts are corrected thanks to the finetuning procedure.

Pixel and feature-based reconstruction

Thanks to the finetuned Flow estimation module, we can obtain disparity maps or optical flows optimized for synthesis. The synthesis proceeds in parallel in both pixel and deep feature domains. Since it has been explained once in Ch. 3 for LFVS, in this part, we will present synthesis procedure with more of focus on VFI case and new designs of our network.

In the case of VFI, PixRNet takes concatenated warped frames $\{\tilde{I}_{0\to t}, \tilde{I}_{1\to t}\}$ and masks $\{\tilde{M}_{0\to t}, \tilde{M}_{1\to t}\}$ to learn the reconstructed $\hat{I}_t^{pix}$. Different from our previous PixRNet in Ch. 3 that cascades four simple convolutional layers, we replace the first three layers with novel residual convolutional block (RCB) architecture illustrated in Fig 4.3. A RCB is composed of three convolutional layers, where the first layers with kernel size 1×1 convert $C1$ input channels to $C2$ channels. Then two cascaded convolutional layers with kernel size 3×3 learn a residual map acting on the input. Moreover, in comparison with

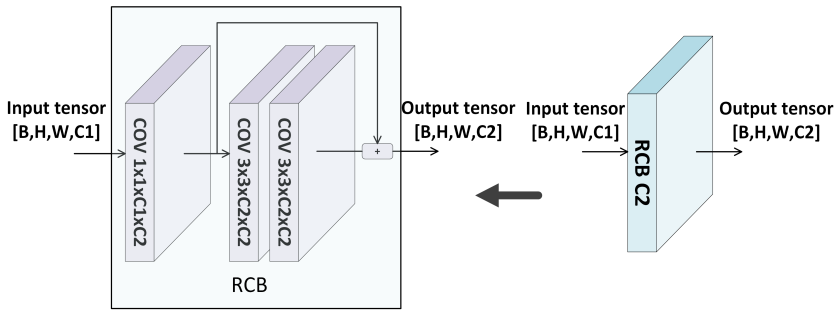


Figure 4.3 – Architecture of Residual Convolutional Block (RCB) with input tensor of dimension $[B, H, W, C1]$ and output tensor of dimension $[B, H, W, C2]$.

the original design in Ch. 3, we quadruple the number of channels in RCB to further enhance its learning ability. The usage of RCB improves the reconstruction efficiency. More details will be investigated in Ch. 4.4.3.

Same as in Ch.3.2.4, due to inconsistency between disparities or optical flows of input views, those warped views or frames are not fully aligned and the fusion may yield blurriness especially in highly textured regions and object contours. To overcome this limitation, we use FeatRNet as an complementary feature-based reconstruction module, in which an encoder is employed to extract deep features at three different resolutions later used in a pyramidal reconstruction step. However, some modifications are also made in FeatRNet. Besides using RCBs to replace convolutional layers, a trainable encoder is employed to substitute previous VGG19 encoder with fixed weights, and it is 1-) more lightweight than the VGG19 encoder (1.9M parameters vs 2.3M parameters). 2-) more flexible to extract features suitable for synthesis.

The feature maps of input frames are extracted at three different resolutions as:

$$\{f_{t'}^1, f_{t'}^2, f_{t'}^3\} = FeatExt(I_{t'}), \forall t' \in \{0, 1\} \tag{4.12}$$

where $FeatExt$ denotes the feature extraction operation and the extracted feature maps $f_{t'}^{s+1}$ are at half resolution compared with $f_{t'}^s$. Like in the PixRNet module, these feature maps are then warped to the desired instant t using forward warping as

$$\tilde{f}_{t'\to t}^s, \tilde{m}_{t'\to t}^s = \mathcal{FW}(I_{t'}, flow_{t'\to t}, e_{t'}), \forall t' \in \{0, 1\} \tag{4.13}$$

Warped features and masks at lower levels are first concatenated and fused by RCBs, and then 2× upsampled by a deconvolutional layer to be fed to the next level. Finally, at the

highest level, the warped feature maps and the occlusion masks along with upsampled features are merged to get the desired view $\hat{I}_t^{feat}$. In case of LFVS, the FeatRNet module will follow the same reconstruction process to obtain $\hat{I}_{\mathbf{j}}^{feat}$.

The adoption of novel encoder have also helped us to simplify the training schedule of network. In Ch.3, feature maps of the ground truth image are extracted using a fixed VGG19 encoder to supervise the feature reconstruction at each level of the FeatRNet module. While with a trainable encoder, the extracted feature maps of the ground truth image vary with the learned encoder weights, and supervising the feature reconstruction with varying ground truth feature maps results in model performance oscillations during training. We therefore no longer supervise feature reconstruction and directly focus on the final reconstruction of $\hat{I}_t^{feat}$ (or $\hat{I}_{\mathbf{j}}^{feat}$) in the FeatRNet module. The simplified training schedule will be detailed later in Ch. 4.3.2.

To make final view well performs in both homogeneous and textured regions, the same FusNet as in Ch. 3 is employed to merge two reconstructed views into the final target view $\hat{I}_t$ (or $\hat{I}_{\mathbf{j}}$) with a soft mask M having value between 0 and 1 (forced by sigmoid activation) as follows

$$\hat{I}_t = M\hat{I}_t^{pix} + (1 - M)\hat{I}_t^{feat}. \tag{4.14}$$

To train the entire architecture, we propose a loss function composed of three reconstructed views as follows

$$\mathcal{L} = \gamma_1\mathcal{L}_l(\hat{I}_{\mathbf{j}}^{pix}, I_{\mathbf{j}}) + \gamma_2\mathcal{L}_l(\hat{I}_{\mathbf{j}}^{feat}, I_{\mathbf{j}}) + \gamma_3\mathcal{L}_l(\hat{I}_{\mathbf{j}}, I_{\mathbf{j}}), \tag{4.15}$$

where $\mathcal{L}_l$ represents a Laplacian loss [137] with 3 levels. $\{\gamma_1, \gamma_2, \gamma_3\}$ are weight hyperparameters.

4.3 Implementation details

4.3.1 Training data preparation and augmentation

In case of LFVS, we adopt the same data preparation and augmentation strategies to process training data, readers can refer to Ch. 3.3.1 and Ch. 3.3.2 for setting details.

For VFI, we adopt the widely used Vimeo90K dataset in [114] for training. This training set contains 51,313 video triplets with resolution 256×448. We use the intermediate frame as ground truth and other two frames as inputs, and randomly crop patches in size

160×160 during training, the batch size is set to 18.

4.3.2 Training schedule

Like previously mentioned, by adopting novel encoder, we have simplified the training schedule. The training schedule of the model can be divided into an initialization step followed by an end-to-end optimization step. In the initialization step, we fix the weights in the FNet module and set $\gamma_1 = \gamma_2 = \gamma_3 = 1$ in the loss function. The training will enable both PixRNet, FeatRNet and FusNet to generate views that approximate the ground truth view. We set the learning rate to 0.0001 for this step, and the whole pipeline is trained for 3000 epoches in the case of VSLF and for 30 epoches in the case of VFI.

Then, by setting $\gamma_1 = \gamma_2 = 0$ and $\gamma_3 = 1$ in the loss function, we end-to-end train the framework to optimize the final synthesized view, which makes PixRNet and FeatRNet to retrieve views with complementary properties. The learning rate in this stage is set to 0.00001 and we stop the training when there is no performance amelioration within 2000 epoches for LFVS or 20 epoches for VFI.

The training procedure takes about 4 days for LFVS and 5 days for VFI on a Nvidia Tesla V100 GPU with 32GB GRAM, the pipeline is implemented using the *tensorflow* framework.

LFs	Disparity range	DeepVS[1]	Soft3D[60]	LLFF[37]	EPI[98]	FPFR	FPFR+	FPFR+*
mona†	[-5,5] (10)	38.90	40.92	41.20	37.54	42.47	<u>43.26</u>	**43.57**
butterfly†	[-6,8] (14)	40.68	42.75	41.35	39.61	42.69	<u>43.52</u>	**43.79**
buddha†	[-10,6] (16)	41.08	41.86	40.66	40.05	42.78	<u>43.50</u>	**43.72**
cotton⋆	[-9,9] (18)	47.24	48.95	47.07	47.97	48.58	<u>50.48</u>	**50.59**
boxes⋆	[-7,13] (20)	33.64	32.14	**34.97**	31.65	33.86	33.41	<u>34.29</u>
dino⋆	[-10,10] (20)	38.41	41.69	41.26	38.44	42.66	<u>43.69</u>	**44.09**
sideboard⋆	[-10,12] (22)	30.91	30.23	32.33	27.30	31.85	<u>32.37</u>	**32.75**
Toy_bricks◇	[-1,22] (23)	28.90	36.58	37.98	31.46	38.84	<u>40.44</u>	**40.89**
Elec_devices◇	[-10,17] (27)	34.09	36.24	36.76	31.55	37.63	<u>39.25</u>	**39.50**
stilllife†	[-16,16] (32)	26.29	34.73	32.73	32.02	36.39	<u>37.20</u>	**37.74**
Lion◇	[-5,29] (34)	28.05	35.18	35.22	33.91	35.47	<u>35.66</u>	**35.75**
Two_vases◇	[-5,39] (44)	25.65	32.49	<u>35.82</u>	29.09	35.56	35.54	**36.26**
Sculpture◇	[-26,34] (60)	22.31	29.15	29.68	26.22	<u>30.09</u>	30.02	**30.12**
Bear◇	[-38,53] (91)	18.36	28.00	33.22	23.40	31.87	<u>33.41</u>	**34.22**
Average	-	32.49	36.49	36.43	33.59	37.92	<u>38.70</u>	**39.09**

Table 4.1 – Quantitative results (PSNR) for the reconstructed central view of the synthetic test light fields. The corresponding datasets are indicated by symbols: ⋆ [55], ◇ [131] and † [54]. The best performances are shown in bold and the second best ones are underlined.

4.4 Experiments

In this section, we evaluate the effectiveness of our novel architecture for both LFVS and VFI tasks using various datasets.

4.4.1 Light field view synthesis

Results with synthetic light fields

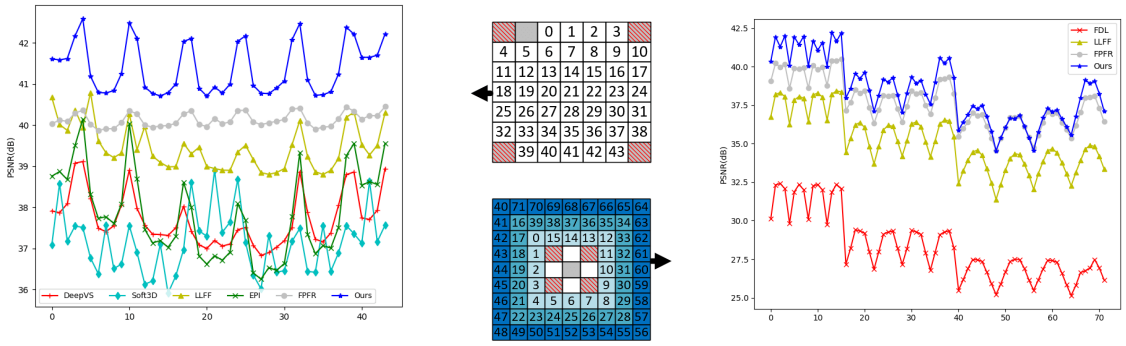
For experiments on synthetic light field data, we keep the same settings as in Ch. 3.4.1 and compare with SOTA view synthesis methods, including our previous architecture FPFR.

Table 4.1 gives a quantitative evaluation of the synthesized central view in terms of PSNR, with all tested light fields arranged from dense to sparse according to the disparity range. In the last two columns, we show the performance of the vanilla version (FPFR+) and test-time augmented version (FPFR+*) of our method. We can observe from the table that our novel pipeline outperforms the state-of-the-art methods with most scenes, and in average gives about 0.8dB gain against the best reference method. Besides testing on the central view that is the most distant from corner views, we further trace the PSNR curves when varying the target position for the different methods, Fig 4.4(a) shows the synthesis quality evolution in terms of viewpoints. We can observe that our method outperforms reference ones with a very large margin. Furthermore, in comparison with the FPFR network, our novel architecture gains in average 1dB for views distant from the corner views, and even more for views that are near the source input views.

Results with real light fields

We also carry out experiments on real light fields with the same setting as in Ch. 3.4.2. Table 4.2 shows the quantitative measures obtained for the central reconstructed views. A gain of 0.5dB is observed against FPFR, the best reference method.

Fig. 4.5 shows reconstruction error maps for both synthetic and real-world light fields, we can observe that our method reconstructs views with less errors, especially on the object contours and subtle structures in highly textured regions.



(a) PSNR for **interpolated** viewpoints (b) View indices (c) PSNR for **extrapolated** viewpoints

Figure 4.4 – Averaged PSNR curves for the different viewpoints. The 8 synthetic scenes are from the datasets [54, 55]. (a) Interpolation. (c) Extrapolation. (b) View indices for interpolation (top) and extrapolation (bottom). 4 input views (red slash) are used for DeepVS[1], EPI[98], Soft3D[60], FPFR, FDL[132] and FPFR+ (Ours), whereas 5 input views (grey) are used for LLFF[37].

LFs	DeepVS	Soft3D	LLFF	EPI	FPFR	FPFR+
Cars	31.53	27.68	29.06	28.17	<u>32.05</u>	**32.86**
Flower1	33.13	30.29	30.00	30.44	<u>34.19</u>	**34.63**
Flower2	31.95	30.52	28.90	29.26	<u>33.80</u>	**34.12**
Rock	34.32	32.67	32.60	32.46	<u>36.48</u>	**37.13**
Leaves	27.97	27.34	27.74	26.48	<u>32.27</u>	**32.65**
Seahorse	32.03	30.41	28.50	26.62	<u>34.68</u>	**34.95**
Average	31.82	29.82	29.47	28.90	<u>33.91</u>	**34.39**

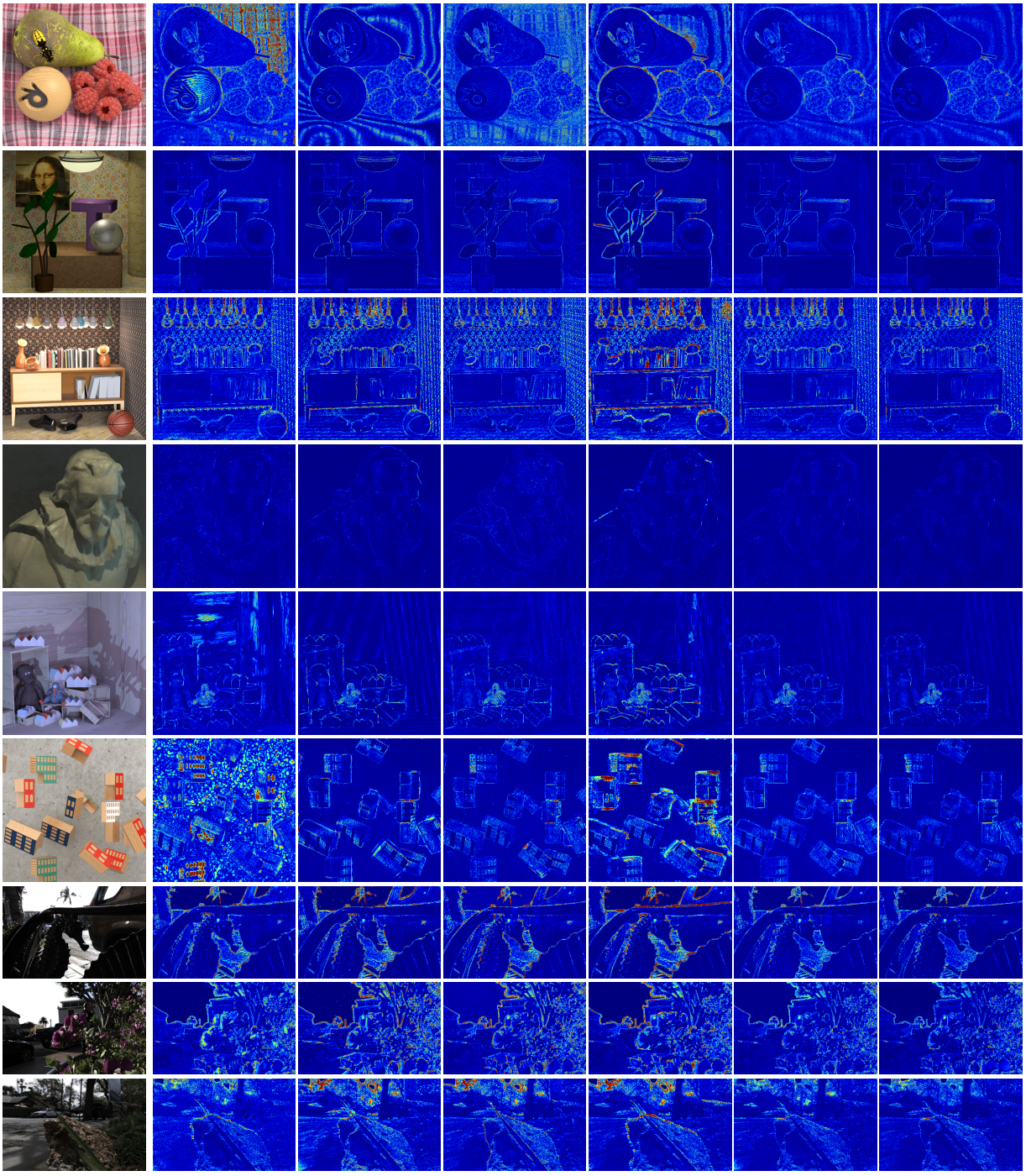
Table 4.2 – Quantitative results (PSNR) for the reconstructed view (5,5) with real-world data (8×8 views) [1]. The best performances are shown in bold and the second best ones are underlined.

Extrapolation

We have also conduct experiment on view extrapolation task like in Ch. 3.4.4. Fig. 4.4(c) shows the performances for each method. We can observe that our method yields better results than the reference methods. Wider is the baseline, more important are the gains with our approach. Let us note that all learning-based methods used in this test have been trained for the interpolation task, hence here we assess the inherent extrapolation capability of each method without any further finetuning.

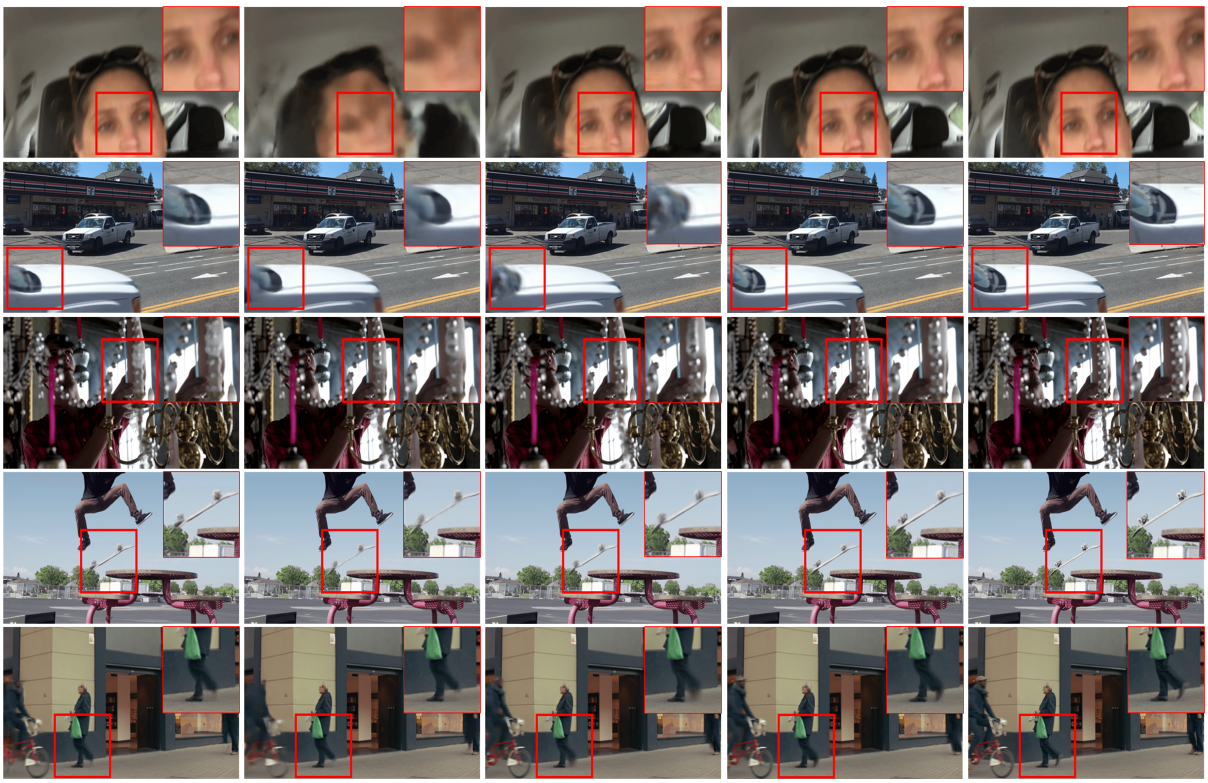
4.4.2 Video frame interpolation

Besides angular view synthesis in light field, we also assess our pipeline for video frame temporal interpolation, using two datasets:



Light field view DeepVS [1] Soft3D [60] LLFF [37] EPI [98] FPFR FPFR+

Figure 4.5 – Reconstruction error maps of view $\hat{I}_{5,5}$ for different synthesis methods, with higher error value in red and lower error value in blue.



(a) MEMC-Net [124] (b) FeFlow [121] (c) SMSP [119] Ours GT

Figure 4.6 – Video frame interpolation results in comparison with (a) MEMC-Net [124], (b) FeFlow [121], (c) SMSP [119]. The 1st two rows use frames from [138], the last three rows use frames from [114]. We use official outputs of SMSP for the last three rows.

— The Vimeo90K datatset [114] that contains 3,782 video triplets with resolution 256×448, covering a large variety of scenes and actions. We interpolate the intermediate frames using the two adjacent frames as inputs.

— The Adobe240fps dataset [138] that is a real-world video set captured with a hand-held camera, it is more challenging than Vimeo90K as it contains more motion blur. We extracted 777 frame triplets (about 10% of the total number of frames) with resolution 360×640 and synthesized the intermediate frames.

We compare the quality of the interpolated frames with the one obtained with state-of-the-art VFI methods, i.e. with MEMC-Net [124], which merges kernel and flow-based synthesis in its pipeline; FeFlow [121], which exploits a flow of features instead of pixels for motion estimation; and Softmax Splatting (SMSP) [119], a flow-based framework that employs pixels and features in the synthesis. For the MEMC-Net and FeFlow methods, We use the official authors source code and their pre-trained model. However, as the code and model are not available for SMSP, we re-implemented this method and trained it using the

Methods	Vimeo90K[114]		Adobe240fps[138]	
	PSNR	SSIM	PSNR	SSIM
MEMC-Net[124]	34.43	0.963	31.54	<u>0.927</u>
FeFlow[121]	35.09	0.963	31.50	0.925
SMSP[119]	35.51	<u>0.967</u>	<u>31.60</u>	<u>0.927</u>
*SMSP**[119]	**36.10**	**0.970**	-	-
FPFR+	<u>35.96</u>	**0.970**	**31.72**	**0.928**

Table 4.3 – Quantitative evaluations (PSNR&SSIM) for the synthesized intermediate frames. The best performances are shown in bold and the second best ones are underlined.

Vimeo90K training set following the instructions in the paper. We could not reproduce the results of the paper, therefore, we give in Table 4.3 both results, the ones taken from the paper [119] (noted as SMSP*), and those obtained with our implementation.

Table 4.3 gives the PSNR and SSIM performances for all tested methods, we can notice that our proposed architecture, although originally designed for angular view synthesis in light field, is well suited for temporal video frame interpolation. It can generate intermediate frames of a quality that is comparable to the one of the top-rank official SMSP method. The table shows that the proposed method outperforms state-of-the-art methods by a large margin.

Fig 4.6 shows interpolated frames obtained with different methods. Compared with other methods, our architecture is able to retrieve high quality frames with subtle details. Let us note that when the input frames contain some blur, like in the 1st row, the reference methods may suffer from deformation or may fail to synthesize intermediate views, while our method is more robust and can still generate plausible results.

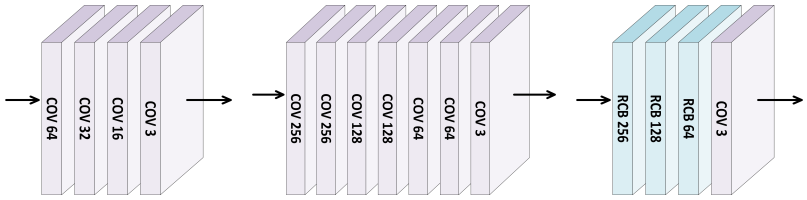
4.4.3 Ablation study

In this section, we carry out an ablation study to analyze the performance gain brought by the new architecture when compared with FPFR proposed in Ch. 3.

Residual convolutional block (RCB)

We assess the performance gain obtained by using the RCB instead of simple convolutional layers. We consider the PixRNet module and fix the weights in the FNet module so that the quality of the synthesized views only depends on the synthesis block. We compare the three variants shown in Fig 4.7, where Fig 4.7(a) shows the fundamental architecture adopted in FPFR, Fig 4.7(b) an enhanced architecture with more kernels and layers,

which can also be considered as the non-residual connection version of Fig 4.7(c), and where Fig 4.7(c) is our current PixRNet architecture based on RCB. Their corresponding learning curves are compared in Fig 4.8, with all curves have been obtained using four light field scenes in old HCI dataset [55]. One can notice that adding kernels and layers like in Fig. 4.7(c) allows improving the network performance. Residual connections further ameliorate the synthesis quality.



(a) PixRNet used in FPFR (b) PixRNet using CBs (c) PixRNet using RCBs

Figure 4.7 – Three variants for PixRNet, with (a) fundamental architecture used in FPFR, (b) PixRNet based on CB (without residual connection), (c) PixRNet based on RCB.

Trainable encoder

Another improvement in our pipeline is the use of a trainable encoder, instead of a fixed VGG19 encoder. A lightweight trainable encoder that flexibly extracts feature maps in cooperation with the synthesis step effectively accelerate the network convergence. By fixing weights in the FNet module, we trace the learning curves of the FeatRNet module using two different encoders (see Fig. 4.9). We can note that with a trainable encoder, the FeatRNet module quickly converges to a high PSNR value, which means that the extracted features are suitable for synthesis. While the features extracted with the VGG19 encoder originally designed for the classification task are less suitable for the reconstruction task, and yields slower network convergence.

Global architecture

Finally, Fig. 4.10 shows that the training of the proposed architecture, when compared with FPFR, converges faster. Tables 4.1 and 4.2 show that this novel architecture is quite efficient for both light field view synthesis and video frame temporal interpolation.

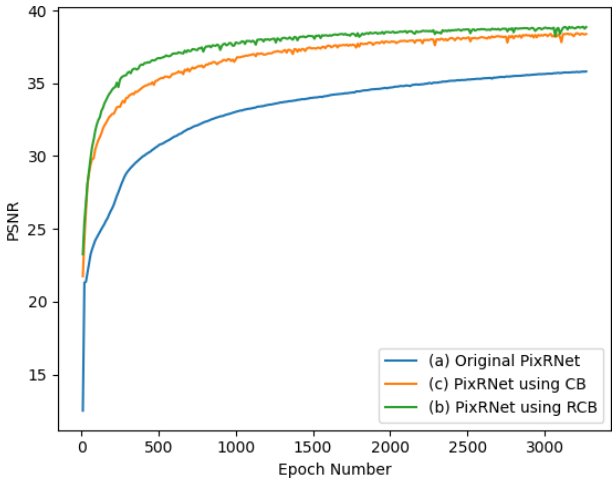


Figure 4.8 – Learning curves of different architectures, with (a) PixRNet in FPFR, (b) PixRNet using RCB, (c) PixRNet using CB.

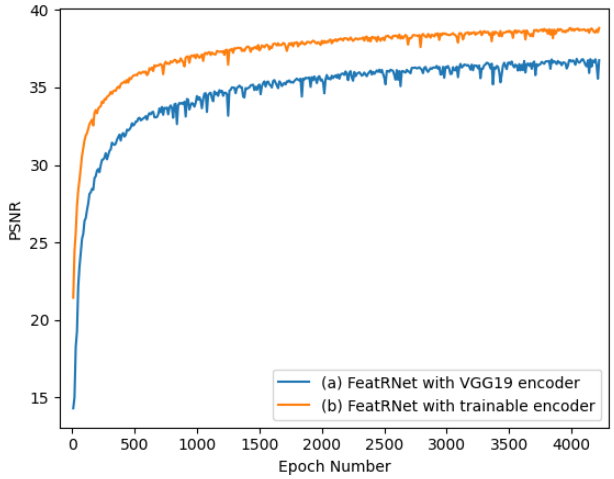


Figure 4.9 – Training curves for different encoders: (a) FeatRNet using VGG19 encoder; (b) FeatRNet using a trainable encoder.

4.4.4 Light field compression using view synthesis

Let us note that pipelines proposed in Ch. 3 and Ch. 4 are not only effective for view synthesis task, but also very useful in some other tasks when serving as a sub-module. In this part, we take light field compression task as an application example. Although light fields give a very rich description of a scene and in particular information about the parallax and depth of the scene which allows constructing panoramic images, 3D viewing with full parallax. However, this comes with very large volumes of extremely redundant data, hence very high processing costs and very large storage needs.

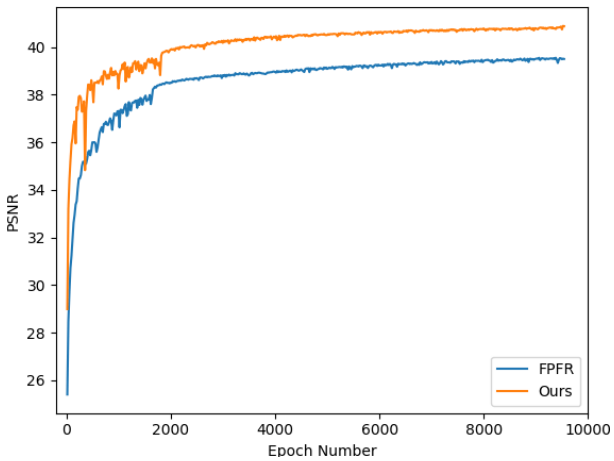


Figure 4.10 – Training curves of FPFR and FPFR+(Ours)

Facing this challenge, light field compression has been a very active field, a variety of methods have been proposed for light field compression. One typical method to compress a light field is applying HEVC inter coding of the video with a lozenge sequencing of all sub-aperture views (denoted as HEVC in the following experiments), another widely used compression algorithm is JPEG-Pleno [2]. Both HEVC and JPEG-Pleno require to encode all sub-aperture views in encoder, then decoder decodes transmitted data into a light field. However, encoding each view of light field isn't very efficient, we hence show that by adopting view synthesis techniques proposed in this thesis (FPFR and FPFR+), a more efficient compression pipeline can be achieved.

Fig. 4.11 illustrates the light field compression pipeline using view synthesis network FPFR and FPFR+. In this scheme, in order to compress a light field L, we first extract four corner views $\{L_{tl}, L_{tr}, L_{bl}, L_{br}\}$ and employ HEVC inter coding to encoder these views on the encoder side. Then on the decoder side, transmitted data are first decoded to obtain corner views $\{L'_{tl}, L'_{tr}, L'_{bl}, L'_{br}\}$, and FPFR or FPFR+ is utilized to reconstruct the whole light field L'. In comparison with HEVC inter coding or JPEG-Pleno, compression based on view synthesis doesn't require to encoder all sub-aperture images, therefore, being easily to reach a better compression performance.

Fig. 4.12 shows rate-distortion curves for scenes *stilllife* and *buddha* in HCI dataset. We compress four corner views with HEVC inter coding and use FPFR and FPFR+ to reconstruct the whole 9×9 views on the decoder side. While HEVC and JPEG-Pleno involve all 81 views. It is obvious that a better PSNR-rate performance is achieved with the help of our view synthesis schemes. As FPFR+ proposed in Ch. 4 has better synthesis

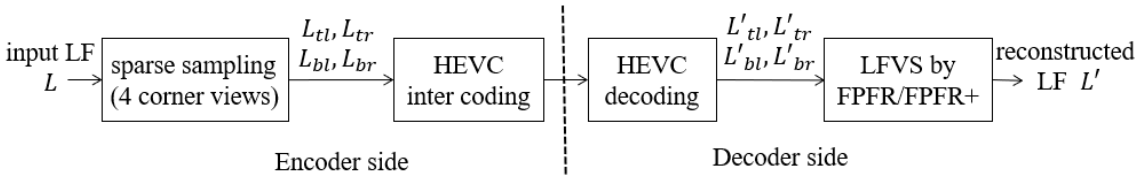
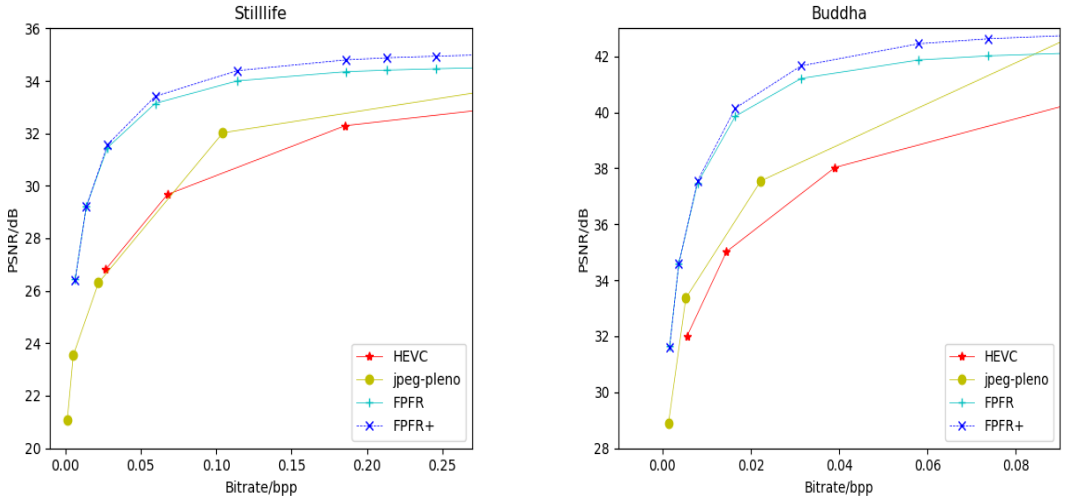


Figure 4.11 – A light field compression pipeline using view synthesis technique.

performance than FPFR proposed in Ch. 3, corresponding PSNR-rate curve of FPFR+ shows consequently better performance than that of FPFR.



(a) PSNR-rate performance for *stilllife*

(b) PSNR-rate performance for *buddha*

Figure 4.12 – PSNR-rate performance for view synthesis-based compression scheme. The proposed methods FPFR and FPFR+ are compared with HEVC and JPEG-Pleno compression solutions.

4.5 Conclusion

In this chapter, we have presented a flexible learning-based architecture for both angular view synthesis for light field and video frame temporal interpolation. The proposed scheme employs a state-of-the-art flow estimation network to predict flows between input images. The synthesis is then conducted in parallel in pixel-based and feature-based branches followed by a soft mask fusion to obtain the final synthesized view. Compared with the FPFR solution originally designed for light field view synthesis, the novel architecture adopts residual convolutional blocks (RCBs) and a lighter weight trainable encoder

that allows improving both training efficiency and synthesis performance for light field view synthesis. The architecture is further generalized to handle video frame temporal interpolation.

The effectiveness of the proposed solution is demonstrated by comparing with state-of-the-art methods using both light field synthesis and video interpolation datasets. Experimental results show that our method tackles well both tasks with high quality results. Besides view synthesis task, we have also integrated our pipeline into a light field compression workflow, which aims at reconstructing the whole light field from a subset of views with the help of view synthesis technique. Finally, by using FPFR and FPFR+ in the scheme, we achieve a better compression performance than two typical compression algorithms HEVC inter coding and JPEG-Pleno.

Chapter 5

Conclusion

5.1 Summary

Light field imaging has been an active research field in recent years. Although it shows great potential in a variety of computer vision applications, technical challenges remain to be investigated due to hardware limitations or huge data redundancy before the technology can actually be used in practice. In this thesis, we address some of the most common problems encountered in light field applications, as we believe solving such problems is meaningful and helpful for more practical applications of the light field. In each of the following three chapters (Ch.2,Ch.3,Ch.4), we pose a key question that reveals our motivations and objectives before setting about constructing corresponding workflows:

1.How to precisely and efficiently estimate a depth map using a small subset of views in a light field?

2.Can we retrieve the entire light field from a small subset of views and the corresponding estimated depth values?

3.Can a view synthesis architecture for light field be generalized to solve temporal view synthesis problem?

And the work in Ch.2,Ch.3,Ch.4 can be considered as the answer to each of these questions.

We first target the problem of light field depth (or disparity) estimation. Existing methods relying on EPI structure is merely suitable for densely sampled data, while others based on pixel-matching technique generate imprecise estimates. Viewing this, we propose a framework that takes a flexible subset of input views to estimate disparity maps for both densely and sparsely sampled light fields. It first divides the input set of views into several stereo image pairs for predicting multiple disparity maps, then applies a warping error-based fusion and a refinement procedure to output an accurate disparity map. The proposed pipeline shows superior estimation accuracy compared with other reference methods with a more flexible subset of input views.

Another important aspect addressed in this thesis is light field view synthesis, which consists of rendering novel views from a sparse set of input views. The synthesis can increase the angular sampling of a light field, hence be an effective solution of light field's spatio-angular super-resolution. In this part, we propose a learning-based view synthesis pipeline that takes sparsely sampled views and outputs a densely sampled light field based on pixel and feature cues. The synthesized views are of high quality in both homogeneous and textured regions thanks to the fusion of pixel-based and feature-based reconstructions. Experimental results show that our proposed method outperforms other reference methods on both synthetic and real-world light field scenes.

In Ch. 4, based on previous work, we extend our light field view synthesis framework to solve a video frame interpolation problem. Firstly, we focus on some architecture design issues, and successfully improve the synthesis quality by adopting novel residual convolution blocks and a trainable encoder. Then, viewing light field and video sequences as scene sampling in the angular or temporal dimensions, we adapt the view synthesis architecture to perform video frame interpolation. Experiments show that our novel architecture is quite effective for both tasks. In the case of light field view synthesis, it yields better performance than the previous architecture and outperforms other reference methods. And in the case of video frame interpolation, it generates frames with plausible details and is shown to be competitive with those produced by the best reference method.

Finally, we conduct an additional experiment by integrating the view synthesis pipelines proposed in Ch. 3 and Ch. 4 into a light field compression workflow, showing that light fields can be compressed with a PSNR-rate performance better than using HEVC and JPEG-Pleno.

5.2 Future work and perspectives

We think there are several possible extensions of this thesis.

In Ch. 2, we propose a framework composed of three modules in charge of initial disparity estimation, error-based fusion and refinement. Although being proven effective for disparity estimation, it still has two obvious shortcomings: first, the number of parameters in *FlowNet2.0* is very high, which is an issue for training, and a sub-network by sub-network training procedure will tremendously prolong the global training schedule. Then, the *winner takes all* strategy adopted in the fusion step is non-differentiable, consequently preventing end-to-end finetuning the whole pipeline. Considering these two drawbacks, future extensions are envisaged to make the pipeline slimmer and differentiable:

we can replace *FlowNet2.0* with other lightweight flow estimation networks, for example, *LiteFlownet* proposed in [139], which has much less parameters than *FlowNet2.0* while remaining high estimation quality. Then the fusion step based on *winner takes all* strategy can be replaced with a soft fusion strategy (e.g. using softmax function to estimate confidences) to guarantee the derivability of such an operation. The filter numbers in the refinement network can be reduced as well according to practical GRAM consumption. In this way, we can obtain a more lightweight depth estimation pipeline, and end-to-end training such a pipeline may produce better depth estimation results.

Although pipelines in Ch. 3 and Ch. 4 are end-to-end trainable, model size can similarly be reduced by using *LiteFlownet* to replace *PWC-Net*. Moreover, the attention mechanism that modulates intermediate data with attention vectors along one or multiple dimensions (e.g. spatial attention, channel-wise attention) [140], has succeeded in a large variety of computer vision tasks. In our synthesis block, warped images are fused in a CNN without attention modulation, which means all views (inter-view) and all regions inside a single image (intra-view) are tackled equally with the same attention. Therefore, a possible extension of our pipeline is to integrate attention mechanism (both inter-view and intra-view attentions) into the synthesis block. We expect that by applying attention mechanism, during the synthesis procedure, our pipeline will pay more attention to the views warped from nearby inputs than those from distant inputs (inter-view attention), and to the textured regions than those homogeneous regions within each warped view (intra-view attention).

Another interesting extension of this work is combining the intra-view attention mechanism with deformable convolutions to handle inter-view misalignment. After warping several angular views to the same position, misaligned contours or textures may attract 'too much' attention than those well aligned ones and lead to 'blurriness' in the attention map. While deformable convolutions generating local offsets have potentials to correct misaligned pixels. We can hence add an attention term in the loss function as a regularizer, and optimize deformable convolutions weights in order to minimize attention paid to misalignment.

Besides extensions on existing works, one of our future work is targeting dynamic light field data, the work in Ch. 4 handles either light field view synthesis or video frame interpolation problem. We plan to construct a novel framework for light field video synthesis by taking both the angular and temporal renderings into account. This means we should combine angular view consistency at each instant and inter-frame correlations at each

viewpoint in our future pipeline to ensure a better synthesis quality. But this is a long story as lots of challenges remained to be conquered, such as the selection of scene flow estimator, loss function design and light field video data acquisition etc.

To end this thesis, light field imaging has been a very active domain in recent years. Although there is still a long journey for light field to become a widely adopted technology, various algorithms proposed to address its spatio-angular trade-off and other application problems have gradually made light field an optional imaging modality for cinematography, medical care and industrial production. We hope our solutions proposed in this thesis will be helpful to people who work on or use light field imaging, and pave the way for a large-scale adoption of light field imaging techniques.

MY PUBLICATIONS

[1] J. Shi, X. Jiang, and C. Guillemot. « A Framework for Learning Depth From a Flexible Subset of Dense and Sparse Light Field Views ». In: *IEEE Trans. Image Process. (TIP)* 28.12 (Dec. 2019), pp. 5867–5880.

[2] J. Shi, X. Jiang, and C. Guillemot. « Learning Fused Pixel and Feature-based View Reconstructions for Light Fields ». In: *IEEE. Int. Conf. on Computer Vision and Pattern Recognition (CVPR)*. 2020, pp. 2555–2564.

[3] X. Jiang, J. Shi, and C. Guillemot. « A Learning Based Depth Estimation Framework for 4D Densely and Sparsely Sampled Light Fields ». In: *IEEE Int. Conf. on Acoustics, Speech and Signal Processing (ICASSP)*. 2019, pp. 2257–2261.

BIBLIOGRAPHY

[1] N. Kalantari, T. Wang, and R. Ramamoorthi. « Learning-based view synthesis for light field cameras ». In: *ACM Trans. on Graphics (TOG)* 35.6 (2016), 193:1–193:10.

[2] ISO/IEC JTC 1/SC29/WG1 JPEG. « Verification model software version 2.0 on jpeg pleno light field coding ». In: *Doc. N82046* (2018).

[3] A. Gershun. « The Light Field ». In: *J. Math. Phys* 18.1–4 (Apr. 1939), pp. 51–151.

[4] E. H. Adelson and J. R. Bergen. « The plenoptic function and the elements of early vision ». In: *Computational Models of Visual Processing* (1991).

[5] S. Gortler, R. Grzeszczuk, R. Szeliski, and M. Cohen. « The lumigraph ». In: *ACM Trans. on Graphics (TOG)*. 1996, pp. 43–54.

[6] M. Levoy and P. Hanrahan. « Light field rendering ». In: *Proc. 23rd Annu. Conf. Comput. Graph. Interact. Techn.(CCGIT)*. 1996, pp. 31–42.

[7] J. Yang, M. Everett, C. Buehler, and L. McMillan. « A real-time distributed light field camera. » In: *Eurographics Workshop (EGW)* 2002 (2002), pp. 77–86.

[8] B. Wilburn, N. Joshi, V. Vaish, E. Talvala, E. Antunez, A. Barth, A. Adams, M. Horowitz, and M. Levoy. « High performance imaging using large camera arrays ». In: *ACM Trans. on Graphics (TOG)* 24.3 (July 2005), pp. 765–776.

[9] B. Wilburn, N. Joshi, V. Vaish, M. Levoy, and M. Horowitz. « High-speed videography using a dense camera array ». In: *IEEE. Int. Conf. on Computer Vision and Pattern Recognition (CVPR)*. 2004.

[10] N. Sabater, G. Boisson, B. Vandame, P. Kerbiriou, F. Babon, M. Hog, R. Gendrot, T. Langlois, O. Bureller, A. Schubert, and V. Allié. « Dataset and Pipeline for Multi-view Light-Field Video ». In: *IEEE Conf. on Computer Vision and Pattern Recognition Workshops (CVPRW)*. 2017, pp. 1743–1753.

[11] K. Venkataraman, D. Lelescu, J. Duparré, A. McMahon, G. Molina, P. Chatterjee, R. Mullis, and S. Nayar. « Picam: An ultra-thin high performance monolithic camera array ». In: *ACM Trans. on Graphics (TOG)* 32.6 (2013), pp. 1–13.

[12] J. Tanida, T. Kumagai, K. Yamada, S. Miyatake, K. Ishida, T. Morimoto, N. Kondou, D. Miyazaki, and Y. Ichioka. « Thin observation module by bound optics (TOMBO): concept and experimental verification ». In: *Applied optics* 40.11 (2001), pp. 1806–1813.

[13] S. Baker and T. Kanade. « Limits on super-resolution and how to break them ». In: *IEEE Trans. Pattern Anal. Mach. Intell. (TPAMI)* 24.9 (2002), pp. 1167–1183.

[14] T. Bishop, S. Zanetti, and P. Favaro. « Light field superresolution ». In: *IEEE Int. Conf. on Computational Photography (ICCP)*. 2009, pp. 1–9.

[15] *Pelican Imaging Array Camera*. `http://lightfield-forum.com/light-field-camera-prototypes/pelican-imaging-array-camera-light-field-module-for-smartphones/`.

[16] *Blender website*. `https://www.blender.org/`.

[17] *Maya website*. `https://www.autodesk.com/products/maya/overview?term=1-YEAR&support=null`.

[18] *3ds Max website*. `https://www.autodesk.com/products/3ds-max/overview?term=1-YEAR&support=null`.

[19] *Project CLIM*. `http://clim.inria.fr/DataSoftware.html`.

[20] *Raytrix*. `https://raytrix.de/`.

[21] E. H. Adelson and J. Y. A. Wang. « Single lens stereo with a plenoptic camera ». In: *IEEE Trans. Pattern Anal. Mach. Intell. (TPAMI)* 14.2 (Feb. 1992), pp. 99–106.

[22] R. Ng, M. Levoy, M. Brédif, G. Duval, M. Horowitz, and P. Hanrahan. « Light field photography with a hand-held plenoptic camera ». In: *Computer Science Technical Report (CSTR)* 2.11 (2005), pp. 1–11.

[23] *Compressive Light Field Photography*. `https://web.media.mit.edu/~gordonw/CompressiveLightFieldPhotography/`.

[24] SD. Babacan, R. Ansorge, M. Luessi, R. Molina, and AK. Katsaggelos. « Compressive sensing of light fields ». In: *IEEE Int. Conf. on Image Processing (ICIP)*. 2009, pp. 2337–2340.

[25] K. Marwah, G. Wetzstein, Y. Bando, and R. Raskar. « Compressive light field photography using overcomplete dictionaries and optimized projections ». In: *ACM Trans. on Graphics (TOG)* 32.4 (2013), pp. 1–12.

[26] E. Miandji, J. Unger, and C. Guillemot. « Multi-shot single sensor light field camera using a color coded mask ». In: *Eu. Signal Process. Conf. (EUSIPCO)*. 2018, pp. 226–230.

[27] SD. Babacan, R. Ansorge, M. Luessi, PR. Matarán, R. Molina, and AK. Katsaggelos. « Compressive light field sensing ». In: *IEEE Trans. Image Process. (TIP)* 21.12 (2012), pp. 4746–4757.

[28] E. Miandji, J. Kronander, and J. Unger. « Compressive image reconstruction in reduced union of subspaces ». In: *Computer Graphics Forum*. Vol. 34. 2. 2015, pp. 33–44.

[29] H. Nguyen, E. Miandji, and C. Guillemot. « Multi-Mask Camera Model for Compressed Acquisition of Light Fields ». In: *IEEE Trans. Comput. Imaging (TCI)* 7 (2021), pp. 191–208.

[30] B. Choudhury, R. Swanson, F. Heide, G. Wetzstein, and W. Heidrich. « Consensus convolutional sparse coding ». In: *IEEE Int. Conf. on Computer Vision (ICCV)*. 2017, pp. 4280–4288.

[31] AK. Vadathya, S. Cholleti, G. Ramajayam, V. Kanchana, and K. Mitra. « Learning light field reconstruction from a single coded image ». In: *Asian Conf. on Pattern Recognition (ACPR)*. 2017, pp. 328–333.

[32] M. Gupta, A. Jauhari, K. Kulkarni, S. Jayasuriya, A. Molnar, and P. Turaga. « Compressive light field reconstructions using deep learning ». In: *IEEE Conf. on Computer Vision and Pattern Recognition workshop (CVPRW)*. 2017, pp. 11–20.

[33] O. Nabati, D. Mendlovic, and R. Giryes. « Fast and accurate reconstruction of compressed color light field ». In: *IEEE Int. Conf. Comput. Photography (ICCP)*. 2018, pp. 1–11.

[34] *Light Field Gantry*. `http://lightfield.stanford.edu/acq.html`.

[35] *Github of LLFF*. `https://github.com/Fyusion/LLFF`.

[36] B. Michael, F. John, O. Ryan, E. Daniel, H. Peter, D. Matthew, D. Jason, B. Jay, W. Matt, and D. Paul. « Immersive Light Field Video with a Layered Mesh Representation ». In: *ACM Trans. on Graphics (TOG)* 39.4 (2020), 86:1–86:15.

[37] B. Mildenhall, P. Srinivasan, R. Ortiz-Cayon, N. Kalantari, R. Ramamoorthi, R. Ng, and A. Kar. « Local Light Field Fusion: Practical View Synthesis with Prescriptive Sampling Guidelines ». In: *ACM Trans. on Graphics (TOG)* 38.4 (2019), pp. 1–14.

[38] J. Flynn, M. Broxton, P. Debevec, M. DuVall, G. Fyffe, R. Overbeck, N. Snavely, and R. Tucker. « Deepview: View synthesis with learned gradient descent ». In: *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*. 2019, pp. 2367–2376.

[39] R. Farrugia and C. Guillemot. « Face hallucination using linear models of coupled sparse support ». In: *IEEE Trans. Image Proc. (TIP)* 26.9 (2017), pp. 4562–4577.

[40] N. Wang, X. Gao, D. Tao, H. Yang, and X. Li. « Facial feature point detection: A comprehensive survey ». In: *Neurocomputing* 275 (2018), pp. 50–65.

[41] F. Farooq, J. Ahmed, and L. Zheng. « Facial expression recognition using hybrid features and self-organizing maps ». In: *IEEE Int. Conf. on Multimedia and Expo (ICME)*. 2017, pp. 409–414.

[42] W. Zhao and R. Chellappa. « Illumination-insensitive face recognition using symmetric shape-from-shading ». In: *IEEE. Int. Conf. on Computer Vision and Pattern Recognition (CVPR)*. Vol. 1. 2000, pp. 286–293.

[43] J. Booth, E. Antonakos, S. Ploumpis, G. Trigeorgis, Y. Panagakis, and S. Zafeiriou. « 3D face morphable models " In-The-Wild" ». In: *IEEE. Int. Conf. on Computer Vision and Pattern Recognition (CVPR)*. 2017, pp. 48–57.

[44] M. Feng, S. Gilani, Y. Wang, and A. Mian. « 3d face reconstruction from light field images: A model-free approach ». In: *Eu. Conf. on Computer Vision (ECCV)*. 2018, pp. 501–518.

[45] R. Raghavendra, K. Raja, and C. Busch. « Presentation attack detection for face recognition using light field camera ». In: *IEEE Trans. Image Proc. (TIP)* 24.3 (2015), pp. 1060–1075.

[46] Z. Li, Y. Ji, J. Yu, and J. Ye. « 3D Fluid Flow Reconstruction Using Compact Light Field PIV ». In: *Eu. Conf. on Computer Vision (ECCV)*. 2020, pp. 120–136.

[47] T. Wang, J. Zhu, E. Hiroaki, M. Chandraker, A. Efros A, and R. Ramamoorthi. « A 4d light-field dataset and cnn architectures for material recognition ». In: *Eur. Conf. on Computer Vision (ECCV)*. 2016, pp. 121–138.

[48] D. Dansereau, B. Girod, and G. Wetzstein. « LiFF: Light field features in scale and depth ». In: *IEEE. Int. Conf. on Computer Vision and Pattern Recognition (CVPR)*. 2019, pp. 8042–8051.

[49] J. Zhang, Y. Liu, S. Zhang, R. Poppe, and M. Wang. « Light field saliency detection with deep convolutional networks ». In: *IEEE Trans. Image Process. (TIP)* 29 (2020), pp. 4421–4434.

[50] H. Zhu, Q. Zhang, and Q. Wang. « 4D light field superpixel and segmentation ». In: *IEEE. Int. Conf. on Computer Vision and Pattern Recognition (CVPR)*. 2017, pp. 6384–6392.

[51] *Stanford Light Field Archives*. `http://lightfields.stanford.edu/`.

[52] M. Rerabek and T. Ebrahimi. « New light field image dataset ». In: *Int. Conf. on Quality of Multimedia Experience (QoMEX)*. EPFL-CONF-218363. 2016.

[53] L. Guillo, X. Jiang, G. Lafruit, and C. Guillemot. « Light field video dataset captured by a R8 Raytrix camera (with disparity maps) ». PhD thesis. INTERNATIONAL ORGANISATION FOR STANDARDISATION ISO/IEC JTC1/SC29/WG1 & WG11, 2018.

[54] S. Wanner, S. Meister, and B. Goldluecke. « Datasets and benchmarks for densely sampled 4D light fields ». In: *Conf. on Vision, Modeling & Visualization (VMV)*. 2013, pp. 225–226.

[55] K. Honauer, O. Johannsen, D. Kondermann, and B. Goldluecke. « A dataset and evaluation methodology for depth estimation on 4D light fields ». In: *Asian Conf. on Computer Vision (ACCV)*. 2016, pp. 19–34.

[56] N. Mayer, E. Ilg, P. Hausser, P. Fischer, D. Cremers, A. Dosovitskiy, and T. Brox. « A Large Dataset to Train Convolutional Networks for Disparity, Optical Flow, and Scene Flow Estimation ». In: *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*. 2016, pp. 4040–4048.

[57] S. Heber, W. Yu, and T. Pock. « Neural EPI-Volume Networks for Shape from Light Field ». In: *IEEE Int. Conf. on Computer Vision (ICCV)*. 2017, pp. 2271–2279.

[58] *Chocofur website*. `http://www.chocofur.com/`.

[59] *Sketchfab website*. `https://sketchfab.com/`.

[60] E. Penner and L. Zhang. « Soft 3D reconstruction for view synthesis ». In: *ACM Trans. on Graphics (TOG)* 36.6 (2017), 235:1–235:11.

[61] A. Saxena, M. Sun, and A. Ng. « Make3d: Learning 3d scene structure from a single still image ». In: *IEEE Trans. Pattern Anal. Mach. Intell. (TPAMI)* 31.5 (2008), pp. 824–840.

[62] D. Eigen, C. Puhrsch, and R. Fergus. « Depth map prediction from a single image using a multi-scale deep network ». In: *Advances in Neural Information Processing Systems (NIPS)* (2014).

[63] Y. Cao, Z. Wu, and C. Shen. « Estimating depth from monocular images as classification using deep fully convolutional residual networks ». In: *IEEE Trans. Circuits and Syst. Video Technol. (TCSVT)* 28.11 (2017), pp. 3174–3182.

[64] Y. Kuznietsov, J. Stuckler, and B. Leibe. « Semi-supervised deep learning for monocular depth map prediction ». In: *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*. 2017, pp. 6647–6655.

[65] S. Evain and C. Guillemot. « A Lightweight Neural Network for Monocular View Generation with Occlusion Handling ». In: *IEEE Trans. Pattern Anal. Mach. Intell. (TPAMI)* (2019).

[66] D. Scharstein and R. Szeliski. « A taxonomy and evaluation of dense two-frame stereo correspondence algorithms ». In: *Int. J. of Computer Vision (IJCV)* 47.1-3 (Dec. 2002), pp. 7–42.

[67] O. Veksler. « Stereo matching by compact windows via minimum ratio cycle ». In: *IEEE Int. Conf. on Computer Vision (ICCV)*. 2001, pp. 540–547.

[68] C. L. Zitnick and T. Kanade. « A cooperative algorithm for stereo matching and occlusion detection ». In: *IEEE Trans. Pattern Anal. Mach. Intell. (TPAMI)* 22.7 (July 2000), pp. 675–684.

[69] T. Taniai, Y. Matsushita, and T. Naemura. « Graph cut based continuous stereo matching using locally shared labels ». In: *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*. 2014, pp. 1613–1620.

[70] D. Maurer, Y. Ju, M. Breuß, and A. Bruhn. « Combining Shape from Shading and Stereo: A Joint Variational Method for Estimating Depth, Illumination and Albedo ». In: *Int. J. of Computer Vision (IJCV)* 126.12 (Dec. 2018), pp. 1342–1366.

[71] L. Zhang and S. M. Seitz. « Estimating optimal parameters for MRF stereo from a single image pair ». In: *IEEE Trans. Pattern Anal. Mach. Intell. (TPAMI)* 29.2 (Feb. 2007), pp. 331–342.

[72] D. Scharstein and C. Pal. « Learning conditional random fields for stereo ». In: *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*. 2007, pp. 1–8.

[73] J. Zbontar and Y. LeCun. « Computing the stereo matching cost with a convolutional neural network ». In: *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*. 2015, pp. 1592–1599.

[74] Z. Chen, X. Sun, L. Wang, Y. Yu, and C. Huang. « A deep visual correspondence embedding model for stereo matching costs ». In: *IEEE Int. Conf. on Computer Vision (ICCV)*. 2015, pp. 972–980.

[75] A. Kendall, H. Martirosyan, S. Dasgupta, P. Henry, R. Kennedy, A. Bachrach, and A. Bry. « End-to-End Learning of Geometry and Context for Deep Stereo Regression ». In: *IEEE Int. Conf. on Computer Vision (ICCV)*. 2017, pp. 66–75.

[76] R. Bolles, H. Baker, and D. Marimont. « Epipolar-plane image analysis: An approach to determining structure from motion ». In: *Int. J. of computer vision (IJCV)* 1.1 (Mar. 1987), pp. 7–55.

[77] S. Wanner and B. Goldluecke. « Variational light field analysis for disparity estimation and super-resolution ». In: *IEEE Trans. Pattern Anal. Mach. Intell. (TPAMI)* 36.3 (Aug. 2013), pp. 606–619.

[78] S. Zhang, H. Sheng, C. Li, J. Zhang, and Z. Xiong. « Robust depth estimation for light field via spinning parallelogram operator ». In: *J. of Computer Vision and Image Understanding (JCVIU)* 145 (Apr. 2016), pp. 148–159.

[79] O. Ronneberger, P. Fischer, and T. Brox. « U-net: Convolutional networks for biomedical image segmentation ». In: *Int. Conf. on Medical image computing and computer-assisted intervention (MICCA)*. 2015, pp. 234–241.

[80] C. Shin, H. Jeon, Y. Yoon, I. Kweon, and S. Kim. « EPINET: A Fully-Convolutional Neural Network Using Epipolar Geometry for Depth from Light Field Images ». In: *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*. 2018, pp. 4748–4757.

[81] H. Jeon, J. Park, G. Choe, J. Park, Y. Bok, and Y. Taiand I. So Kweon. « Accurate depth map estimation from a lenslet light field camera ». In: *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*. 2015, pp. 1547–1555.

[82] J. Navarro and A. Buades. « Robust and Dense Depth Estimation for Light Field Images ». In: *IEEE Trans. Image Process. (TIP)* 26.4 (Apr. 2017), pp. 1873–1886.

[83] X. Jiang, M. Le Pendu, and C. Guillemot. « Depth estimation with occlusion handling from a sparse set of light field views ». In: *IEEE Int. Conf. on Image Processing (ICIP)*. 2018, pp. 634–638.

[84] C. Huang. « Empirical Bayesian Light-Field Stereo Matching by Robust Pseudo Random Field Modeling ». In: *IEEE Trans. Pattern Anal. Mach. Intell. (TPAMI)* (Feb. 2018), pp. 1–1.

[85] J. Pang, W. Sun, J. Ren, C. Yang, and Q. Yan. « Cascade Residual Learning: A Two-Stage Convolutional Neural Network for Stereo Matching ». In: *IEEE Int. Conf. on Computer Vision workshops (ICCVW)*. 2017, pp. 878–886.

[86] A. Dosovitskiy, P. Fischer, E. Ilg, P. Hausser, C. Hazirbas, V. Golkov, P. Van Der Smagt, D. Cremers, and T. Brox. « FlowNet: Learning Optical Flow with Convolutional Networks ». In: *IEEE Int. Conf. on Computer Vision (ICCV)*. 2015.

[87] E. Ilg, N. Mayer, T. Saikia, M. Keuper, A. Dosovitskiy, and T. Brox. « Flownet 2.0: Evolution of optical flow estimation with deep networks ». In: *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*. 2017, pp. 1647–1655.

[88] A. Ranjan and M. J. Black. « Optical flow estimation using a spatial pyramid network ». In: *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*. 2017, pp. 2720–2729.

[89] E. Ilg, T. Saikia, M. Keuper, and T. Brox. « Occlusions, Motion and Depth Boundaries with a Generic Network for Disparity, Optical Flow or Scene Flow Estimation ». In: *Eu. Conf. on Computer Vision (ECCV)*. 2018.

[90] D. Sun, X. Yang, M. Liu, and J. Kautz. « PWC-Net: CNNs for optical flow using pyramid, warping, and cost volume ». In: *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*. 2018, pp. 8934–8943.

[91] X. Jiang, J. Shi, and C. Guillemot. « A Learning Based Depth Estimation Framework for 4D Densely and Sparsely Sampled Light Fields ». In: *IEEE Int. Conf. on Acoustics, Speech and Signal Processing (ICASSP)*. 2019, pp. 2257–2261.

[92] M. Tao, S. Hadap, J. Malik, and R. Ramamoorthi. « Depth from combining defocus and correspondence using light-field cameras ». In: *IEEE Int. Conf. on Computer Vision (ICCV)*. 2013, pp. 673–680.

[93] H. Lin, C. Chen, S. Kang, and J. Yu. « Depth recovery from light field using focal stack symmetry ». In: *IEEE Int. Conf. on Computer Vision (ICCV)*. 2015, pp. 3451–3459.

[94] R. Farrugia, C. Galea, and C. Guillemot. « Super resolution of light field images using linear subspace projection of patch-volumes ». In: *IEEE J. Sel. Topics Signal Process. (JSTSP)* 11.7 (2017), pp. 1058–1071.

[95] M. Rossi and P. Frossard. « Geometry-consistent light field super-resolution via graph-based regularization ». In: *IEEE Trans. Image Process. (TIP)* 27.9 (2018), pp. 4207–4218.

[96] Y. Wang, F. Liu, K. Zhang, G. Hou, Z. Sun, and T. Tan. « LFNet: A Novel Bidirectional Recurrent Convolutional Neural Network for Light-Field Image Super-Resolution ». In: *IEEE Trans. Image Process. (TIP)* 27.9 (May 2018), pp. 4274–4286.

[97] R. Farrugia and C. Guillemot. « Light field super-resolution using a low-rank prior and deep convolutional neural networks ». In: *IEEE Trans. Pattern Anal. Mach. Intell. (TPAMI)* 42.5 (2019), pp. 1162–1175.

[98] G. Wu, Y. Liu, Q. Dai, and T. Chai. « Learning Sheared EPI Structure for Light Field Reconstruction ». In: *IEEE Trans. Image Process. (TIP)* 28.7 (2019), pp. 3261–3273.

[99] N. Meng, X. Wu, J. Liu, and E. Lam. « High-order residual network for light field super-resolution ». In: *AAAI Conference on Artificial Intelligence (AAAI)*. Vol. 34. 07. 2020, pp. 11757–11764.

[100] G. Chaurasia, S. Duchene, O. Sorkine-Hornung, and G. Drettakis. « Depth synthesis and local warps for plausible image-based navigation ». In: *ACM Trans. on Graphics (TOG)* 32.3 (2013), pp. 1–12.

[101] P. Srinivasan, T. Wang, A. Sreelal, R. Ramamoorthi, and R. Ng. « Learning to Synthesize a 4D RGBD Light Field from a Single Image ». In: *IEEE Int. Conf. on Computer Vision (ICCV)*. 2017, pp. 2262–2270.

[102] T. Zhou, R. Tucker, J. Flynn, G. Fyffe, and N. Snavely. « Stereo Magnification: Learning View Synthesis using Multiplane Images ». In: *ACM Trans. on Graphics (TOG)* 37.4 (2018), 65:1–65:12.

[103] I. Choi, O. Gallo, A. Troccoli, M. Kim, and J. Kautz. « Extreme View Synthesis ». In: *IEEE Int. Conf. on Computer Vision (ICCV)*. 2019, pp. 7781–7790.

[104] A. Levin and F. Durand. « Linear view synthesis using a dimensionality gap light field prior ». In: *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*. 2010, pp. 1831–1838.

[105] L. Shi, H. Hassanieh, A. Davis, D. Katabi, and F. Durand. « Light field reconstruction using sparsity in the continuous fourier domain ». In: *ACM Trans. on Graphics (TOG)* 34.1 (2014), p. 12.

[106] G. Wu, M. Zhao, L. Wang, Q. Dai, T. Chai, and Y. Liu. « Light Field Reconstruction Using Deep Convolutional Network on EPI ». In: *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*. 2017, pp. 1638–1646.

[107] S. Vagharshakyan, R. Bregovic, and A. Gotchev. « Light field reconstruction using shearlet transform ». In: *IEEE Trans. Pattern Anal. Mach. Intell. (TPAMI)* 40.1 (2018), pp. 133–147.

[108] Y. Wang, F. Liu, Z. Wang, G. Hou, Z. Sun, and T. Tan. « End-to-end View Synthesis for Light Field Imaging with Pseudo 4DCNN ». In: *Eur. Conf. on Computer Vision (ECCV)*. 2018, pp. 333–348.

[109] N. Meng, H. So, X. Sun, and E. Lam. « High-dimensional dense residual convolutional neural network for light field reconstruction ». In: *IEEE Trans. Pattern Anal. Mach. Intell. (TPAMI)* (2019).

[110] H. Yeung, J. Hou, J. Chen, Y. Chung, and X. Chen. « Fast light field reconstruction with deep coarse-to-fine modeling of spatial-angular clues ». In: *Eur. Conf. on Computer Vision (ECCV)*. 2018, pp. 137–152.

[111] W. Lee, K. Choi, and J. Ra. « Frame rate up conversion based on variational image fusion ». In: *IEEE Trans. Image Proc. (TIP)* 23.1 (2013), pp. 399–412.

[112] G. Long, L. Kneip, J. Alvarez, H. Li, X. Zhang, and Q. Yu. « Learning image matching by simply watching video ». In: *Eu. Conf. on Computer Vision (ECCV)*. 2016, pp. 434–450.

[113] Z. Liu, R. Yeh, X. Tang, Y. Liu, and A. Agarwala. « Video frame synthesis using deep voxel flow ». In: *IEEE Int. Conf. on Computer Vision (ICCV)*. 2017, pp. 4463–4471.

[114] T. Xue, B. Chen, J. Wu, D. Wei, and W. Freeman. « Video enhancement with task-oriented flow ». In: *Int. J. Computer Vision (IJCV)* 127.8 (2019), pp. 1106–1125.

[115] H. Jiang, D. Sun, V. Jampani, M. Yang, E. Learned-Miller, and J. Kautz. « Super slomo: High quality estimation of multiple intermediate frames for video interpolation ». In: *IEEE. Int. Conf. on Computer Vision and Pattern Recognition (CVPR)*. 2018, pp. 9000–9008.

[116] W. Bao, X. Zhang, L. Chen, L. Ding, and Z. Gao. « High-order model and dynamic filtering for frame rate up-conversion ». In: *IEEE Trans. Image Proc. (TIP)* 27.8 (2018), pp. 3813–3826.

[117] S. Niklaus and F. Liu. « Context-aware synthesis for video frame interpolation ». In: *IEEE. Int. Conf. on Computer Vision and Pattern Recognition (CVPR)*. 2018, pp. 1701–1710.

[118] W. Bao, W. Lai, C. Ma, X. Zhang, Z. Gao, and M. Yang. « Depth-aware video frame interpolation ». In: *IEEE. Int. Conf. on Computer Vision and Pattern Recognition (CVPR)*. 2019, pp. 3703–3712.

[119] S. Niklaus and F. Liu. « Softmax Splatting for Video Frame Interpolation ». In: *IEEE. Int. Conf. on Computer Vision and Pattern Recognition (CVPR)*. 2020, pp. 5437–5446.

[120] W. Chen, Z. Fu, D. Yang, and J. Deng. « Single-image depth perception in the wild ». In: *Advances in Neural Information Processing Systems (NIPS)* (2016).

[121] S. Gui, C. Wang, Q. Chen, and D. Tao. « FeatureFlow: Robust Video Interpolation via Structure-to-Texture Generation ». In: *IEEE. Int. Conf. on Computer Vision and Pattern Recognition (CVPR)*. 2020, pp. 14004–14013.

[122] S. Niklaus, L. Mai, and F. Liu. « Video frame interpolation via adaptive convolution ». In: *IEEE. Int. Conf. on Computer Vision and Pattern Recognition (CVPR)*. 2017, pp. 670–679.

[123] S. Niklaus, L. Mai, and F. Liu. « Video frame interpolation via adaptive separable convolution ». In: *IEEE Int. Conf. on Computer Vision (ICCV)*. 2017, pp. 261–270.

[124] W. Bao, W. Lai, X. Zhang, Z. Gao, and M. Yang. « Memc-net: Motion estimation and motion compensation driven neural network for video interpolation and enhancement ». In: *IEEE Trans. Pattern Anal. Mach. Intell. (TPAMI)* (2019).

[125] S. Meyer, O. Wang, H. Zimmer, M. Grosse, and A. Sorkine-Hornung. « Phase-based frame interpolation for video ». In: *IEEE. Int. Conf. on Computer Vision and Pattern Recognition (CVPR)*. 2015, pp. 1410–1418.

[126] S. Meyer, A. Djelouah, B. McWilliams, A. Sorkine-Hornung, M. Gross, and C. Schroers. « Phasenet for video frame interpolation ». In: *IEEE. Int. Conf. on Computer Vision and Pattern Recognition (CVPR)*. 2018, pp. 498–507.

[127] O. Johannsen, K. Honauer, B. Goldluecke, A. Alperovich, F. Battisti, Y. Bok, M. Brizzi, M. Carli, G. Choe, M. Diebold, et al. « A Taxonomy and Evaluation of Dense Light Field Depth Estimation Algorithms ». In: *IEEE Conf. on Computer Vision and Pattern Recognition workshop (CVPRW)*. 2017, pp. 1795–1812.

[128] X. Jiang, M. Le Pendu, R. Farrugia, and C. Guillemot. « Light field compression with homography-based low-rank approximation ». In: *IEEE J. Sel. Topics Signal Process. (JSTSP)* 11.7 (Oct. 2017), pp. 1132–1145.

[129] Z. Zhang, Y. Liu, and Q. Dai. « Light field from micro-baseline image pair ». In: *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*. 2015, pp. 3800–3809.

[130] K. Simonyan and A. Zisserman. « Very deep convolutional networks for large-scale image recognition ». In: *arXiv preprint arXiv:1409.1556* (2014).

[131] J. Shi, X. Jiang, and C. Guillemot. « A Framework for Learning Depth From a Flexible Subset of Dense and Sparse Light Field Views ». In: *IEEE Trans. Image Process. (TIP)* 28.12 (Dec. 2019), pp. 5867–5880.

[132] M. L. Pendu, C. Guillemot, and A. Smolic. « A Fourier Disparity Layer Representation for Light Fields ». In: *IEEE Trans. Image Process. (TIP)* 28.11 (Nov. 2019), pp. 5740–5753.

[133] J. Justin, A. Alexandre, and F. Li. « Perceptual losses for real-time style transfer and super-resolution ». In: *Eur. Conf. on Computer Vision (ECCV)*. 2016, pp. 694–711.

[134] M. Jaderberg, K. Simonyan, A. Zisserman, and K. Kavukcuoglu. « Spatial transformer networks ». In: *Advances in Neural Information Processing Systems (NIPS)*. 2015, pp. 2017–2025.

[135] F. Perazzi, J. Pont-Tuset, B. McWilliams, L. Van Gool, M. Gross, and A. Sorkine-Hornung. « A Benchmark Dataset and Evaluation Methodology for Video Object Segmentation ». In: *IEEE. Int. Conf. on Computer Vision and Pattern Recognition (CVPR)*. 2016.

[136] D. Butler, J. Wulff, and M. Black. « A naturalistic open source movie for optical flow evaluation ». In: *Eu. Conf. on Computer Vision (ECCV)*. 2012, pp. 611–625.

[137] P. Bojanowski, A. Joulin, D. Lopez-Paz, and A. Szlam. « Optimizing the latent space of generative networks ». In: *arXiv preprint arXiv:1707.05776* (2017).

[138] S. Su, M. Delbracio, J. Wang, G. Sapiro, W. Heidrich, and O. Wang. « Deep video deblurring for hand-held cameras ». In: *IEEE. Int. Conf. on Computer Vision and Pattern Recognition (CVPR)*. 2017, pp. 1279–1288.

[139] T. Hui, X. Tang, and C. Change Loy. « Liteflownet: A lightweight convolutional neural network for optical flow estimation ». In: *IEEE. Int. Conf. on Computer Vision and Pattern Recognition (CVPR)*. 2018, pp. 8981–8989.

[140] L. Chen, H. Zhang, J. Xiao, L. Nie, J. Shao, W. Liu, and T. Chua. « Sca-cnn: Spatial and channel-wise attention in convolutional networks for image captioning ». In: *IEEE. Int. Conf. on Computer Vision and Pattern Recognition (CVPR)*. 2017, pp. 5659–5667.

UNIVERSITE BRETAGNE LOIRE / MATHSTIC

UNIVERSITÉ DE RENNES 1

Titre : L'estimation de la profondeur de champ de lumière et le synthèse de vue basés sur l'apprentissage

Mot clés : champs de lumière, estimation de profondeur, synthèse de vue, interpolation de trame

Résumé : Un champ de lumière échantillonne la scène à partir de différentes perspectives, et les informations directionnelles contenues dans chaque vue de champ de lumière permettent d'obtenir de meilleures performances dans ces tâches de vision par ordinateur que l'utilisation d'images 2D classiques. Les contributions principales de cette thèse se présentent sous trois aspects.

Notre première contribution se concentre sur l'estimation de la profondeur (ou de la disparité) à partir du champ de lumière avec un sous-ensemble de vues. Les méthodes existantes d'estimation de la profondeur conviennent soit à des champs de lumière simplement échantillonnés de manière dense, soit à la prédiction de profondeurs imprécises avec des artefacts visuels apparents. Nous proposons un nouveau cadre basé sur l'apprentissage qui génère des profondeurs précises avec un plus petit sous-ensemble de vues avec une grande variété de plages de disparités.

Ensuite, dans la deuxième partie, nous nous concentrons sur le problème de synthèse de vue en champ de lumière. En adoptant les techniques d'estimation de la profondeur utilisées dans les travaux précédents et en combinant des pixels et des caractéristiques, nous proposons un pipeline de synthèse dépendant de la profondeur, qui peut suréchantillonner un champ de lumière à toutes les dimensions souhaitées et produire des vues synthétisées de haute qualité.

Dans le dernier aspect, nous améliorons notre pipeline de synthèse avec de nouvelles architectures et l'adaptons à la tâche d'interpolation de trame vidéo. Le schéma amélioré montre une performance supérieure pour la tâche de synthèse de vue de champ de lumière et donne des résultats compétitifs avec les méthodes d'état de l'art d'interpolation de trame vidéo.

Title: Learning-based depth estimation from light field and view synthesis

Keywords: light fields, depth estimation, view synthesis, frame interpolation

Abstract: A light field samples the scene from different perspectives, and the directional information contained in each light field view makes it yield better performance in those computer vision task than using classical 2D images. The main contributions of this thesis are in three aspects.

Our first contribution focuses on the depth (or disparity) estimation from light field with a subset of views. Existing depth estimation methods either is suitable for merely densely sampled light fields, or predict imprecise depths with apparent visual artifacts. We propose a novel learning-based framework that generates precise depths with a smaller subset of views with a large variety of disparity range.

Then in the second part, we focus on light field view synthesis problem. By adopting depth estimation techniques used in the previous work and by combining pixel and feature cues, we propose a depth-dependent synthesis pipeline, which can upsample a light field to any desired dimensions and produce high quality synthesized views.

In the last aspect, we improve our synthesis pipeline with some new architectures, and adapt it to the video frame interpolation task. The improved scheme shows a superior performance for light field view synthesis task and yields results competitive with the state-of-the-art video frame interpolation methods.