Robust Methods for Accurate and Efficient Reconstruction from Motion Imagery

Qiao Chen

A Thesis

in

The Department

of

Computer Science and Software Engineering

Presented in Partial Fulfillment of the Requirements

for the Degree of

Doctor of Philosophy (Computer Science) at

Concordia University

Montréal, Québec, Canada

February 2023

CONCORDIA UNIVERSITY

School of Graduate Studies

This is to certify that the thesis prepared

By: **Qiao Chen**

Entitled: **Robust Methods for Accurate and Efficient Reconstruction from Motion Imagery**

and submitted in partial fulfillment of the requirements for the degree of

Doctor of Philosophy (Computer Science)

complies with the regulations of this University and meets the accepted standards with respect to originality and quality.

Signed by the Final Examining Committee:

_____ Chair
Dr. Yong Zeng

_____ External Examiner
Dr. Hua Wang

_____ Examiner
Dr. Marta Kersten-Oertel

_____ Examiner
Dr. Anjali Agarwal

_____ Examiner
Dr. Adam Krzyzak

_____ Supervisor
Dr. Charalambos Poullis

Approved by _____
Dr. Lata Narayanan
Department of Computer Science and Software Engineering

_____ 2023

_____
Dr. Mourad Debbabi, Dean
Gina Cody School of Engineering and Computer Science

Abstract

Robust Methods for Accurate and Efficient Reconstruction from Motion Imagery

Qiao Chen, Ph.D.

Concordia University, 2023

Creating virtual representations of real-world scenes has been a long-standing goal in photogrammetry and computer vision, and has high practical relevance in industries involved in creating intelligent urban solutions. This includes a wide range of applications such as urban and community planning, reconnaissance missions by the military and government, autonomous robotics, virtual reality, cultural heritage preservation, and many others.

Over the last decades, image-based modeling emerged as one of the most popular solutions. The objective is to extract metric information directly from images. Many procedural techniques achieve good results in terms of robustness, accuracy, completeness, and efficiency. More recently, deep-learning-based techniques were proposed to tackle this problem by training on vast amounts of data to learn to associate features between images through deep convolutional neural networks and were shown to outperform traditional procedural techniques. However, many of the key challenges such as large displacement and scalability still remain, especially when dealing with large-scale aerial imagery.

This thesis investigates image-based modeling and proposes robust and scalable methods for large-scale aerial imagery. First, we present a method for reconstructing large-scale areas from aerial imagery that formulates the solution as a single-step process, reducing the processing time considerably. Next, we address feature matching and propose a variational optical flow technique (HybridFlow) for dense feature matching that leverages the robustness of graph matching to large displacements. The proposed solution efficiently handles arbitrary-sized aerial images. Finally, for general-purpose image-based modeling, we propose a deep-learning-based approach, an end-to-end

multi-view structure from motion employing hypercorrelation volumes for learning dense feature matches. We demonstrate the application of the proposed techniques on several applications and report on task-related measures.

Acknowledgments

This research is based upon work supported by the Natural Sciences and Engineering Research Council of Canada Grants No. N01670 (Discovery Grant) and DNDPJ515556-17 (Collaborative Research and Development with the Department of National Defence Grant).

I would like to express my sincere gratitude and appreciation to my supervisor Dr. Charalambos Poullis, who provided me with exceptional supervision, support and patience throughout my doctoral studies, his remarkable insight and knowledge of the subject steered me through this research. I would like to express heartfelt thanks to my parents and all my family, especially my husband Guan Wang for all his support and love during these years. Special thanks to my dearest friend Tianyi Zhang for her warm caring and emotional support from our adorable furry pets – Nico and the cats. I would like to thank all the people who helped and encourage me during my doctoral studies, without all the people in my life and work, this thesis would not have been possible. To all of them, I dedicate this thesis.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Creating virtual representations of real-world scenes has been a long-standing goal in photogrammetry and computer vision. Modeling the real world in 3D has high practical relevance in industries involved in creating intelligent urban solutions. This includes a wide range of applications such as urban and community planning, reconnaissance missions by the military and government, autonomous robotics, virtual and mixed reality, cultural heritage digital preservation, training of emergency response personnel, inspection and monitoring in manufacturing, virtual tourism, entertainment, and disaster management, to name a few. The reconstruction of urban areas involves the recovery of the camera poses – the position, orientation, field-of-view, and the estimation of metric information about the rigid structures in the scene. The reconstructed models can then be used as the backbone of any digital twin application that allows users to virtually explore and study the urban area and its processes.

Numerous technologies have been created to facilitate 3D reconstruction, which can be classified into active and passive acquisition techniques. Active acquisition techniques, such as LiDAR and structure-light 3D scanning, produce very accurate pointclouds depicting the scene, but they are not scalable and are costly to install. Moreover, data processing is a tedious and time-consuming endeavour. Passive acquisition methods, on the other hand, gather reflectance information without emitting signals, such as photos captured by RGB cameras.

The formulation of the reconstruction problem can vary depending on the application. In autonomous driving applications, for instance, the problem is formulated as estimating the camera's

trajectory relative to its surroundings while simultaneously mapping the environment, whereas in general purpose applications, the problem is formulated as the two-step process of recovering the camera poses and then the geometry.

1.1 Motivation

A broad audience today is interested in applications of image-based 3D reconstruction. The problem of image-based 3D modeling is one of the important tasks in the field of computer vision. The goal of image-based 3D modeling is to derive useful geometric information directly from 2D images to create new models and user interfaces. The objective of the proposed thesis is the automatic 3D reconstruction of large-scale urban areas from 2D images.

Compared to data obtained from active technologies (such as LiDAR and structured-light 3D devices), imagery is abundant and inexpensive. Particularly, aerial and satellite imaging are gaining popularity since they can capture expansive regions in a single image. In recent years, the hardware costs of cameras and unmanned aerial vehicles as commodity sensors have been significantly lower than those of other active devices.

In general, multi-view stereo techniques allow the recovery of 3D information from multiple (at least two) distinct images capturing the same scene. An image-based 3D reconstruction system aims at replicating part of the function of the human visual system, which is to extract 3D metric information from 2D images. Many of the underlying problems in image-based 3D reconstructions are nowadays well-conditioned, however, there are still problems that need to be overcome, especially in the context of large-scale urban areas. Consider an example, reconstructing a large-scale urban area, such as an entire city, where thousands of objects and buildings are present. While tremendous progress has been made in reconstructing 3D urban areas over the last decades, we still do not have a robust reconstruction system for large-scale urban areas, which produces complete and useful 3D models.

Motivated by the above, this thesis presents robust approaches for the precise and efficient 3D reconstruction of urban areas from large-scale 2D aerial images.

1.2 Typical Image-based Modeling Pipeline

Accurately modeling the appearance of 3D geometric reconstructions of urban areas is recognized as one of the most important 3D techniques. As shown in Figure 1.1, a procedural image-based 3D modeling pipeline typically consists of three stages, which are (a) correspondence search, (b) sparse reconstruction, and (c) dense reconstruction. Starting from the 2D image input, the initial step is to search for overlapping and correspondences between image pairs. The resulting correspondence tracks connect multiple images and serve as input for the subsequent 3D reconstruction stage. The 3D reconstruction process is typically solved by recovering a sparse and then a dense model. In the sparse reconstruction, camera information, including intrinsic (focal length, field of view, etc.) and extrinsic (location and orientation), is recovered, along with sparse 3D models of the scene. Then the dense reconstruction process is used to reconstruct a richer representation of the scene, typically in the form of denser point clouds or textured surface meshes.

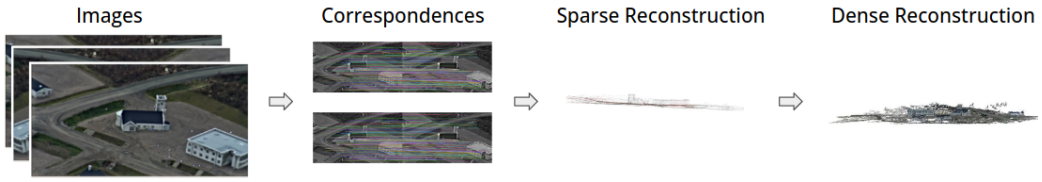


Figure 1.1: Overview of an image-based 3D modeling pipeline.

In deep learning approaches, 3D modeling is commonly processed with techniques where the model learns all the steps between the initial input phase and the final output result. The various applications have different aims, and thus the input and output formats differ. The most common inputs to a multi-view 3D modeling system are the images and the viewpoint, and the output is generally the depth estimation for each view.

1.3 Challenges

The goal of 3D urban area reconstruction is to provide solutions involving no or minimal manual operations and interactions. For human beings, it is a natural ability to conceptually build a 3D model of a shown scene, but it is extremely challenging for computer systems. Organizing and

utilising the extremely rich and diverse image data means increased computation time for processing. There are significant challenges in image-based 3D reconstruction systems, which are mainly caused by the fact that the image formation process is not generally reversible. From the projected position in a camera image plane, a scene point can only be recovered up to a projective transformation. Hence, additional information is needed to solve the reconstruction problem. In contrast to experiments conducted under controlled laboratory conditions, urban areas are difficult to reconstruct because of their characteristics, such as complex illumination, and no or repetitive textures, which cause most of the existing feature-matching techniques to fail. Full automation of urban area matching is hard to achieve because feature matching usually depends on global estimation, but the global estimation is based on the quality of matching, resolving the problem is a dilemma. The related vision problems turn out to be complex computation and optimization tasks, for which many of the traditional methods fail due to the scale of input data. More recently, deep learning techniques have been proposed to tackle this problem. However, because they rely on training on vast amounts of data to learn to associate features between images through deep convolutional neural networks, many of the key challenges, such as large displacement and scalability, still remain, especially when dealing with large-scale aerial imagery.

1.4 Contributions

One of the most important aspects of our approach is finding the matches between pairs of images in our dataset and establishing all possible precise point-wise dense correspondences in a single step. As one of the dense correspondence estimation methods, optical flow is used to find the matches between pairs of images and is an integral part of many computer vision tasks. Techniques of variational optical flow estimation are based on a coarse-to-fine scheme, which interpolates sparse matches and locally optimizes an energy model conditioned on colour, gradient, and smoothness. They are sensitive to noise in the sparse matches, deformations, and arbitrarily large displacements. First, we address this problem and present HybridFlow, a variational motion estimation framework for large displacements and deformations. It is a multi-scale hybrid matching approach performed on the image pairs. This method is ideal for large-scale imagery, such as aerial imagery, because

it does not require training and is robust to large displacements and rigid and non-rigid transformations caused by motion in the scene. More notably, HybridFlow improves motion estimation in the presence of significant deformations by introducing directed graphs of arbitrary topology that represent perceptual groups. Next, the single-step incremental structure from motion (SfM) system is to recover 3D structures using dense flow fields for image pairs from dense correspondences.

We further address the following important problems associated with 3D reconstruction systems: (a) the problems in image matching in the wild due to complex illumination and repetitive or lack of textures; (b) the upper bound on the number of matches imposed by memory limitations due to the large size of the dataset; (c) having different 3D reconstructed models due to multiple disjoint groups of models representing the scene; and (d) the requirement for a two-step approach for generating a dense reconstruction, which is time-consuming.

We present a framework for handling large datasets and densely matched features for generating dense reconstructions. Keypoints are extracted and matched using dense optical flow, which results in vast amounts of data that cannot be handled by any state-of-the-art SfM techniques that are currently available. Moreover, following the recent trend, we propose a deep learning-based approach: an end-to-end multi-view structure from motion employing hypercorrelation volumes for learning dense feature matches.

In summary, our contributions are:

- **On dense feature matching.** We address feature matching and propose a variational optical flow technique (HybridFlow) Q. Chen and Poullis (2022b) for dense feature matching that leverages the robustness of graph matching to large displacements. The proposed solution efficiently handles arbitrary-sized aerial images. This work resulted in a journal publication in Nature Scientific Reports, 2022.

- **On single-step dense reconstruction.** A method Q. Chen and Poullis (2018) for reconstructing large-scale areas from aerial imagery that formulates the solution as a single-step process, reducing the processing time considerably. In this work, we contribute to the 3D reconstruction from arbitrary-sized WAMI datasets of large-scale urban areas in a single shot. This work resulted in a conference paper in IEEE 3DTV, 2018

- **On general-purpose image-based dense reconstruction.** We propose a deep-learning-based approach Q. Chen and Poullis (2022a), an end-to-end multi-view structure from motion employing hypercorrelation volumes for learning dense feature matches. We discuss the extension and further improvements of this work as our future work currently underway. This work resulted in a conference paper in IEEE SPSIS, 2022

- **On applications of motion flow detection for the identification, and tracking.** Using the advantages of the optical flow technique (HybridFlow) Q. Chen and Poullis (2022b), we present an application for motion detection, player identification, and tracking in ice hockey games. This work resulted in a conference paper currently under review by ACM Multimedia Systems, 2023.

1.5 Organization

This thesis is organized as follows:

Chapter 2 introduces the underlying principles of image-based modeling, including feature extraction and matching. Chapter 3 provides an overview of the related work in the areas of image-based modeling and feature matching techniques. Chapter 4 presents an algorithmic variational optical flow technique (HybridFlow) for dense feature matching. Chapter 5 proposes a method for reconstructing large-scale urban areas from aerial imagery that formulates the solution in a single-step process. Chapter 6 describes an end-to-end deep learning-based approach to improve dense multi-view stereo reconstruction. Chapter 7 presents an application of motion detection and player identification and tracking in ice hockey games. Finally, Chapter 8 concludes the thesis and provides an outlook to open problems and future work.

Chapter 2

Principles

2.1 Typical 3D Reconstruction System

One of the most significant issues with traditional 3D modeling systems is the requirement for a two-step approach to apply multi-view stereo (MVS) for dense reconstruction after structure-from-motion (SfM) for sparse reconstruction. As shown in Figure 2.1, Structure from Motion systems involve three main stages: the extraction of features from images, camera motion estimation, and recovery of the 3D structure using the estimated motion and features. Typically, the structure-from-motion (SfM) of the reconstruction problem in computer vision is the problem of recovering the three-dimensional (3D) structure of a stationary scene from a set of projective measurements, represented as a collection of two-dimensional (2D) images, via estimation of the motion of the cameras corresponding to these images. In essence, SfM involves the three main stages of:

(1) extraction of features in images (e.g., points of interest, lines, etc.) and matching these features between images,

(2) camera motion estimation (e.g., using relative pairwise camera positions estimated from the extracted features), and

(3) recovery of the 3D structure using the estimated motion and features (e.g., by minimizing the reprojection error). More specifically, after discussing early factorization-based techniques for motion and structure estimation, this section will provide a detailed account of some of the most recent camera location estimation methods.

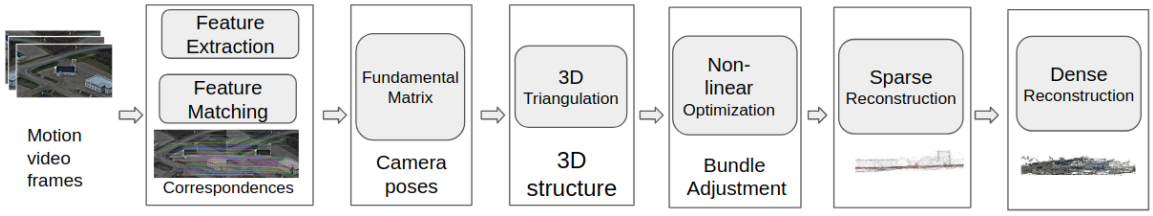


Figure 2.1: Typical 3D reconstruction from images pipeline

Recovering the 3-dimensional structure of a scene from images is a fundamental goal of computer vision. A particularly effective approach to 3D reconstruction involves the use of many images of a stationary scene. This problem, commonly referred to as multiview structure-from-motion (SfM), is the subject of a large body of research in computer vision. Modern methods usually solve the multiview SfM problem using bundle adjustment techniques, which aim to optimize a cost function known as the total reprojection error. With this cost function, given n images of a stationary scene, the objective is to simultaneously determine the structure (3D coordinates of scene points) and the calibration parameters of each of the n cameras that minimize the discrepancy between image measurements and their predictive model.

In 2006 Snavely et al. Snavely, Seitz, and Szeliski (2008) presented a sequential pipeline for SfM, demonstrating that it can produce accurate reconstructions in practical scenarios where hundreds or even thousands of independently captured photographs are provided, sparking a huge interest in the development of efficient SfM techniques for large, unordered image sets. The suggested pipeline begins by detecting keypoints in each image. It then uses the SIFT descriptor D. G. Lowe et al. (1999) to compare those keypoints across images and to produce a set of potential matches. Random sampling and consensus (RANSAC) is applied next to robustly estimate essential matrices between pairs of images (for computation of the relative motion of camera pairs) and to discard outlier matches. Then, starting with a pair of images for which the largest number of inlier matches were found and then greedily adding one image at a time, bundle adjustment is solved repeatedly. Although this sequential pipeline is computationally challenging, it successfully deals with large collections of images, producing in many cases highly accurate reconstructions. However, it is based on greedy steps that may not result in an optimal solution. Clearly, global approaches, which

consider all images simultaneously (at least for the initial camera motion estimation), may potentially yield improved solutions.

2.2 Feature Extraction

Finding correspondences between the input images is a standard initial step in 3D reconstruction. In image-based 3D reconstruction, we are interested in gaining the geometric relationship of multiple images in order to reconstruct the 3D structure of the scene. One of the major problems is robustly and efficiently getting geometric relationships among related input images. This process is typically solved in a two-stage approach: firstly, the contents of all the input images are described independently, then the described features are being matched if they are from the same scene, using the image descriptions.

2.2.1 Hand-crafted Features Descriptors

To describe the contents of an image, local image description decomposes it into locally distinct features. Figure 2.2 visualizes the concept of the local image description. Point-level correspondence information is a mandatory input in order to determine the relative geometric relationship between images. Feature points are typically detected at distinctive locations in the image, such as corners and edges.

The feature points are typically described using the image content in their immediate neighborhood. The features should ideally be distinct and repeatably discernible in different images of the same object. To identify the feature points of the same structure in different images, information about their geometry and appearance is extracted. The geometry information includes position, orientation, size, etc., and the appearance is described by a descriptor vector as a list of colors of a patch around the target point. To make the descriptor invariant under different illumination or viewing conditions, the vector is normalized with respect to its geometry and a standard approach is to encode color gradient information. Scale Invariant Feature Transforms (SIFT) D. G. Lowe et al. (1999), and its derivative work, in Arandjelovic and Zisserman (2012), and more standard local features Bay, Tuytelaars, and Van Gool (2006); Rublee, Rabaud, Konolige, and Bradski (2011), are

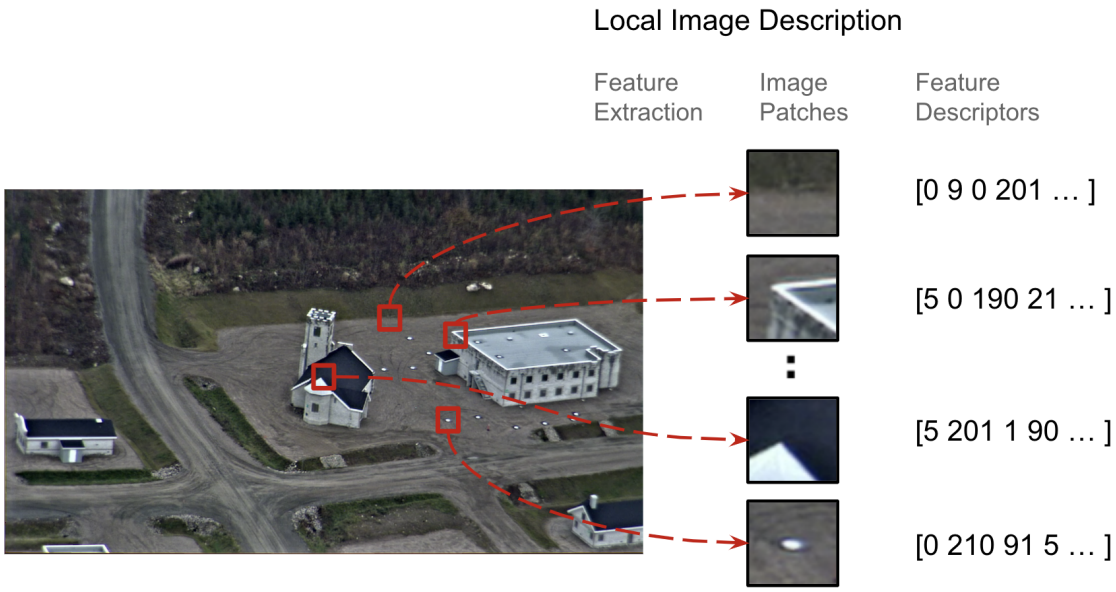


Figure 2.2: An overview of the concept of local image description

the optimal choices for image matching in terms of robustness. Finding correspondences between images is a fundamental process in 3D reconstruction tasks. The traditional way of performing 2D to 3D reconstruction is to find sparse correspondences and refine the camera poses and 3D correspondences, followed by multi-view stereo techniques such as patch matching to generate dense correspondences. The main constraint in the patch matching method is that images from all views have to be loaded in memory; the accuracy of matching depends on the patch size and image content; and this will not generate enough consistent support regions for matching. According to our review, the most popular descriptor for images is the SIFT D. G. Lowe et al. (1999) feature extractor, which was tested on WAMI data, SIFT has been proven to be one of the most robust local invariant feature descriptors. State-of-the-art SfM framework COLMAP Schonberger and Frahm (2016) integrated RootSIFT features and gets correspondences. The computation of SIFT D. G. Lowe et al. (1999) contains four stages: scale space extreme detection, keypoint localization, orientation assignment, and keypoint descriptor.

- Scale-space Extreme Detection;

  The image is convolved with Gaussian filters at different scales, and then the difference of successive Gaussian-blurred images is taken. The keypoints are then identified as maxima

and minima of the Difference of Gaussian (DoG) at multiple scales. Specifically, a DoG $D(x, y, \sigma)$ is given by:

$$D(x, y, \sigma) = L(x, y, k_i\sigma) - L(x, y, k_j\sigma) \tag{1}$$

where $L(x, y, k\sigma)$ is the convolution of the original image $I(x, y)$ with the Guassian blur. DoG image between scales $k_i\sigma$ and $k_j\sigma$ is just the difference of the Gaussian-blurred images at scales $k_i\sigma$ and $k_j\sigma$.

- Keypoint Localization;

  For each candidate keypoint, interpolation of a nearby region is used to accurately determine its position.

- Orientation Assignment;

  Each of the keypoints is assigned one or more orientations based on local image gradient directions.

  For an image sample $L(x, y)$, the gradient magnitude, $m(x, y)$ and orientation, $\theta(x, y)$, are precomputed using pixel differences:

$$m(x, y) = \sqrt{(L(x + 1, y) - L(x - 1, y))^2 + (L(x, y + 1) - L(x, y - 1))^2} \tag{2}$$

$$\theta(x, y) = atan2(L(x, y + 1) - L(x, y - 1), L(x + 1, y) - L(x - 1, y)) \tag{3}$$

  The magnitude and direction calculations for the gradient are done for every pixel in a neighboring region around the keypoint in the Gaussian-blurred image L.

2.2.2 Learned Features Descriptors

Mostly, descriptor learning is formulated as a supervised learning problem. The goal is to learn a way to represent things so that descriptors for the same physical object are close together in descriptor space but descriptors for unrelated things are far apart. For example, given a set $\mathcal{P}$ of positive pairs and a set $\mathcal{N}$ of negative pairs, the margin constraint Simonyan, Vedaldi, and Zisserman (2014) is defined as:

$$d(\boldsymbol{p_1}, \boldsymbol{p_2}) + \tau < d(\boldsymbol{n_1}, \boldsymbol{n_2}) \forall (\boldsymbol{p_1}, \boldsymbol{p_2}) \in \mathcal{P}, (\boldsymbol{n_1}, \boldsymbol{n_2}) \in \mathcal{N} \tag{4}$$

where d is a distance metric and τ is a margin. Working with triplets of descriptors is an alternative to operating with pairs of descriptors. The triplets $l(\boldsymbol{p_1}, \boldsymbol{p_2}, \boldsymbol{n})$ are with $(\boldsymbol{p_1}, \boldsymbol{p_2}) \in \mathcal{P}$ and $((\boldsymbol{p_1}, \boldsymbol{n}), (\boldsymbol{p_2}, \boldsymbol{n})) \in \mathcal{N}$, and potential cost functions are the margin ranking loss J. Wang et al. (2014) is:

$$l(\boldsymbol{p_1}, \boldsymbol{p_2}, \boldsymbol{n}) = max(0, \tau + d((\boldsymbol{p_1}, \boldsymbol{p_2}) - d(\boldsymbol{p_1}, \boldsymbol{n})) \tag{5}$$

which tries to enforce a minimum distance $\tau > 0$ between unrelated descriptors, and the ratio loss Hoffer and Ailon (2015) :

$$l(\boldsymbol{p_1}, \boldsymbol{p_2}, \boldsymbol{n}) = (\frac{e^{d_p}}{e^{d_p} + e^{d_n}})^2 + (\frac{e^{d_n}}{e^{d_p} + e^{d_n}})^2 \tag{6}$$

where $d_p = d(\boldsymbol{p_1}, \boldsymbol{p_2})$ and $d_p = d(\boldsymbol{p_1}, \boldsymbol{n})$. The latter tries to enforce that the distance between related descriptors is significantly smaller than the distance to an unrelated descriptor. The input to the descriptor learning algorithm differs across different techniques. In practice, due to noise in the training data and limited model complexity, it is not possible to fully separate these pair-wise distances, so a margin-based approach will be used, which encourages the distance between the classes of point pairs to separate without enforcing the distance ordering as a hard constraint. Methods based on metric learning often use a fixed descriptor representation as input and learn a discriminative metric, while approaches that learn a new descriptor representation usually operate on raw image patches.

2.3 Feature Matching

Finding correspondences between images is a fundamental process in tasks of 3D reconstruction. The pixel-level correspondence can be established between two or more image regions by matching the local features of two different images depicting the same scene.

2.3.1 Distance Correlation

A metric or distance function is a function $d(x, y)$, that defines the distance between elements of vectors of the non-negative real numbers. If the distance is zero, both elements are equivalent under that specific metric. The L_2 distance (i.e. Euclidean distance) serves as an efficient distance metric to calculate the similarity between two descriptors. The main agenda of distance metrics is to show that if two points p_1 , p_2 in n dimensional space lie near to each other according to the distance metric used, then the two points are supposed to be similar. For n points, the general formula is as:

$$d(x, y) = \sqrt{\sum_{i=1}^{n}(x_i - y_i)^2} \tag{7}$$

DeepMatching Revaud et al. (2016), as an alternative to performing dense wide-baseline matching by first matching a few feature points and triangulating them, then locally rectifying the images, however, in this case, if some matches are mistakenly computed and are even not detected gross reconstruction errors will occur and cause wrong reconstruction results. When it comes to the images with the unique features of multiple targets, weak textures, image scale, and environment occlusions, lead to the failure of image matching. DeepMatching Revaud et al. (2016) computes dense correspondences between images, which relies on a hierarchical, multi-layer, correlational architecture designed for matching images and is inspired by deep convolutional approaches. The proposed matching algorithm can handle non-rigid deformations and repetitive textures and efficiently determines dense correspondences in the presence of significant changes between images. The images are in different sales and each of them is divided into a regular grid, although the descriptor design is SIFT-like D. G. Lowe et al. (1999), the matching accuracy and density are much higher compared to the original SIFT D. G. Lowe et al. (1999).

2.3.2 Graph Matching

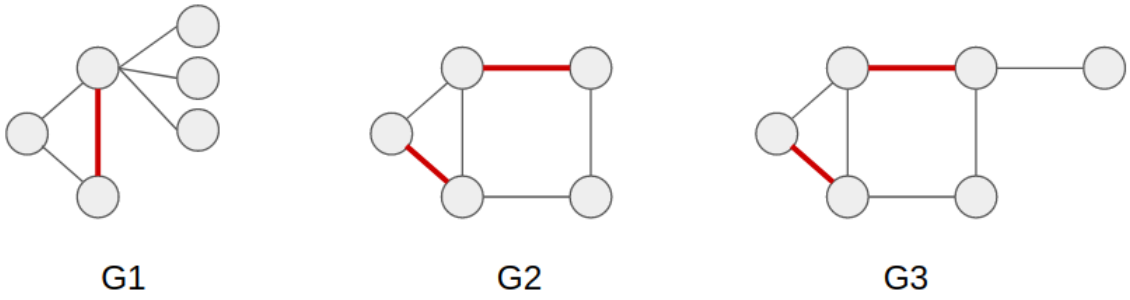


Figure 2.3: An overview of the concept of maximal matching, the red edges denote the matching.

The matching can be a graph matching problem if we consider the feature points to be vertices of a graph. Given an undirected graph, a matching is a set of edges. The goal of graph matching is to determine a mapping between the nodes of the two graphs that preserves the relationships between the nodes in the graph as much as possible. Matching two graphs is defined as node selection on an association graph, with nodes representing candidate correspondences between the two graphs. A vertex is said to be matched if an edge is incident to it, otherwise the vertex is unmatched. Despite the matching of independent feature points, graph matching takes feature location, geometry, and structure into account, which makes it more robust to texture-less surfaces, deformations, and displacement.

Maximal Matching – As illustrated in Figure 2.3, a matching $\boldsymbol{m}$ of graph $\boldsymbol{G}$ is defined as maximal if adding an edge which is in $\boldsymbol{G}$ but not in $\boldsymbol{m}$, makes $\boldsymbol{m}$ not a matching. In other words, a maximal matching $\boldsymbol{m}$ is not a proper subset of any other matching of $\boldsymbol{G}$.

Maximum Matching – A matching that contains the largest possible number of edges. There may be many maximum matchings. The matching number $\nu(G)$ a graph G is the size of a maximum matching. Every maximum matching is maximal, but not every maximal matching is a maximum matching. Figure 2.4 shows examples of maximum matchings in the same three graphs.

Perfect Matching – A matching $\boldsymbol{m}$ of graph $\boldsymbol{G}$ is defined as perfect if every vertex is connected to exactly one edge. Every perfect matching is a maximum matching but not every maximum matching is a perfect matching. Since every vertex has to be included in a perfect matching, the number of edges in the matching must be $\frac{\nu(V)}{2}$ where $\nu(V)$ is the number of vertices. Therefore, a

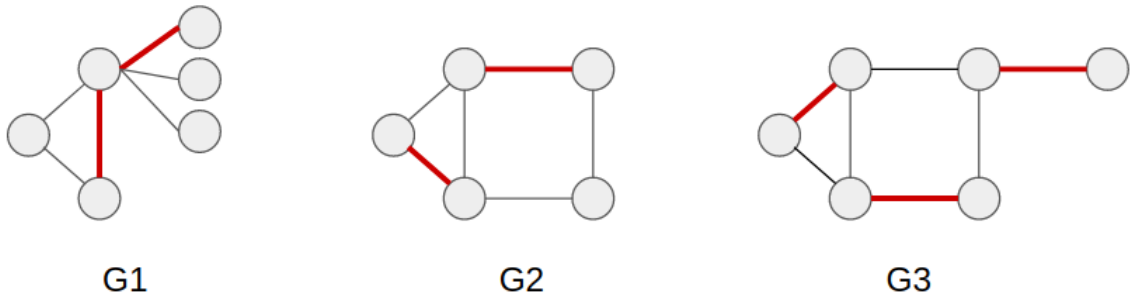


Figure 2.4: An overview of the concept of maximum matching, the red edges denote the matching.

perfect matching only exists if the number of vertices is even. A matching is said to be near perfect if the number of vertices in the original graph is odd, it is a maximum matching and it leaves out only one vertex.

The details of the graph matching algorithm that we implemented is described in Chapter 4.

2.4 Optical Flow

2.4.1 Variational Optical Flow Estimation

We explore the most popular state-of-the-art optical flow techniques to compute our dense matches as input to the later reconstruction. As the state-of-the-art of most recent works, EpicFlow Revaud et al. (2015) is an edge-preserving interpolation of correspondences for optical flow. Ric-flow Hu, Li, and Song (2017) takes the edge-preserving cells and over-segments the scene into superpixels to revitalize an early idea of the piecewise flow model. EpicFlow Revaud et al. (2015), RicFlow Hu et al. (2017) or other traditional optical flow techniques estimate a relative motion for every pixel to the target frame. Because of the nature of how components of the velocity in flow estimation techniques are calculated and the smoothness terms, a large amount of noise is introduced. Contrary to regular grid decomposition of the images into patches, the alternative process is to find the correspondences between structured entities decomposing images by grouping pixels, it enables a semi-dense coverage of the images and adds spacial constraints to pixel-wise correspondences.

Super-pixels are primitives generated by aggregating adjacent pixels, sharing similar characteristics, these entities will provide more reliable areas preserving object shapes and image geometry. The different motion of different objects can be preserved from motion discontinuities, and it is a subset of the super-pixels. In EpicFlow Revaud et al. (2015) noise is inevitable, and matches from super-pixel level offer more consistent correspondences than pixel or patch matches.

Basic Assumptions and Constraints

Let $f(x, y)$ denote the gray value recorded at location (x, y) in the image plane. A basic assumption underlying most approaches to motion estimation is that f is conserved, that is the change of $f(x, y)$ at location (x, y) is due to a movement of $f(x, y)$ to the location $(x + u, y + v)$:

$f(x + u, y + v) = f(x, y)$

A common approach to estimating the optical flow vector (u, v) at some fixed location $(\hat{x}, \hat{y})$ on the image grid $(x, y) = (k \triangle x, l \triangle y), k, l \in \mathbb{Z}$, is to assume u and v to be constant within a local spatial area $N(\hat{x}, \hat{y})$ around $(\hat{x}, \hat{y})$ and to minimize a function of u and v:

$\sum_{k,l \in N(\hat{x},\hat{y})} (f(k + u, l + v) - f(k, l))^2$

Optimization Problem and Discretization

Under mild conditions with respect to the image sequence data, the unique globally minimizing vector field $u(x, y), v(x, y)$, where (x, y) is the location in the image, is determined by a variational equation. The simplest discretization is obtained by choosing a regular triangulation of the image domain ω and attaching to each pixel position a piecewise linear basis function $\phi(x, y)$. Indexing each pixel position (k, l) by 1, 2, . . . , N, we thus have:

$u(x, y) = \sum_{i=1}^{N} u_i \phi_i(x, y)$

and,

$v(x, y) = \sum_{i=1}^{N} v_i \phi_i(x, y)$

where u, v can be conveniently computed by some corresponding iterative solver Hackbusch (1994).

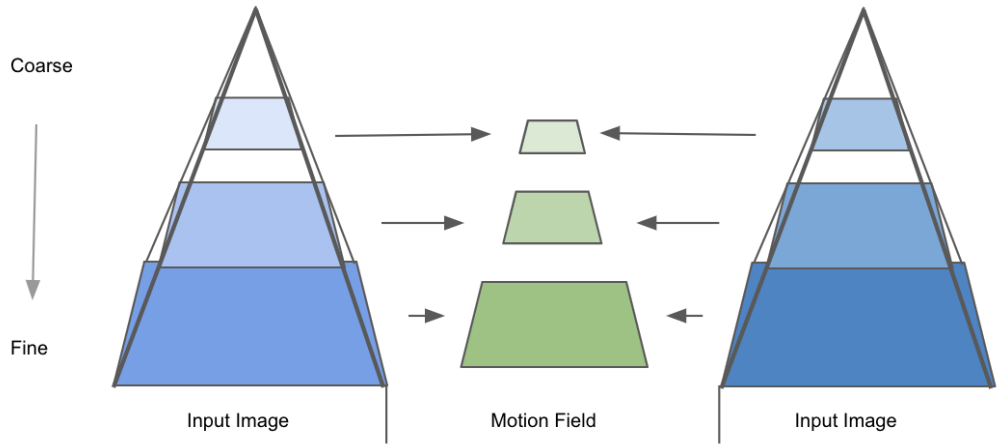


Figure 2.5: Coarse-to-fine Estimation

Coarse-to-Fine Motion Estimation

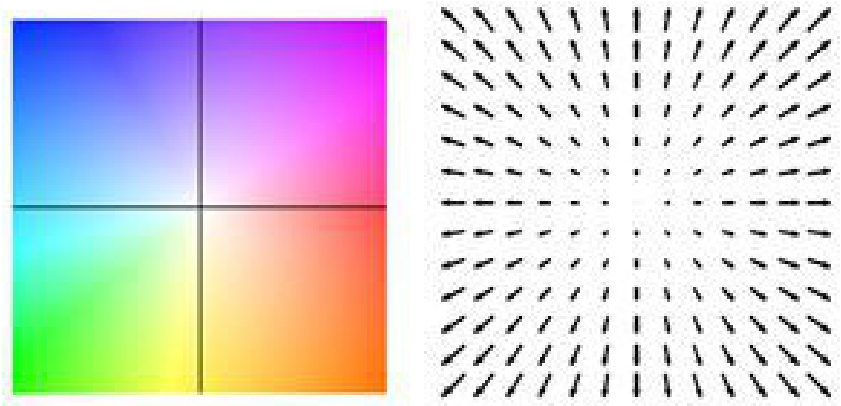
The magnitude of image motion has a significant impact on the accuracy of motion estimation. Depending on the spatial image frequency, very large motions even may cause aliasing along the time-frequency axis. As a solution, we first compute a coarse motion field using only low spatial frequency components to roughly stabilize the position of the image. Then the higher frequency subbands are used to estimate optical flow on the warped sequence. The coarse-to-fine approach, as is shown in Figure 2.5, yields a refined overall optical flow estimate. This process can be repeated at finer and finer spatial scales until the original image resolution is reached. In this context, constructing an image pyramid by recursively applying lowpass filtering and subsampling processes is a typical technique for generating multi-scale representations. It is important that the images at different scales are represented by different sampling rates. As a result, the same derivative filters can be utilized at each scale, eliminating the requirement to create multiple derivative filters, one for each scale.

Benchmarks

Concurrently with pursing better energy models, the establishment of public benchmark datasets for optical flow, such as the Middlebury Baker et al. (2011), MPI Sintel Butler, Wulff, Stanley, and Black (2012), and KITTI Menze, Heipke, and Geiger (2015) Optical Flow benchmark. Challenges

and limitations such as large displacements, dramatic lighting changes, and occlusions are properly revealed in the benchmarks. These public benchmarks have stimulated research on more faithful energy models that overcome some of the particular issues listed above, while also providing for a fair comparison of approaches on the same dataset.

Optical Flow Color Encoding



(a) (b)

Figure 2.6: Optical Flow Color Encoding

The flow field in HSV color space was designed in order to visualize the outcomes of optical flow estimation that is easy for people to understand. The color coding is shown in Figure 2.6. As seen in the image 2.6a, the displacement of each pixel is the vector from the center of the square to that pixel, which is encoded displayed in the multicolored image. This indicates that no optical flow is represented in the center of the image, i.e. the white area. For the flow colors displayed in the left and top corner of the blue quadrant, the darker the shade of blue, the larger the magnitude of the vector. This is also true for the other quadrants, where the stronger the hue, the larger the magnitude of flow to that position.

Average Endpoint Error (EPE/AEE)

The loss function for calculating optical flow is done by making use of the endpoint error. As shown diagrammatically in Figure 2.7, the end-to-end point error (EPE/AEE) is calculated by comparing the predicted optical flow vector with the ground truth (expected) optical flow vector and

is the Euclidean distance between the two vectors.

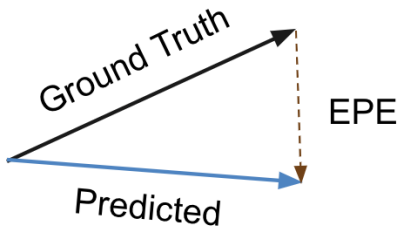


Figure 2.7: Average Endpoint Error (EPE/APE)

2.4.2 Learned Optical Flow Estimation

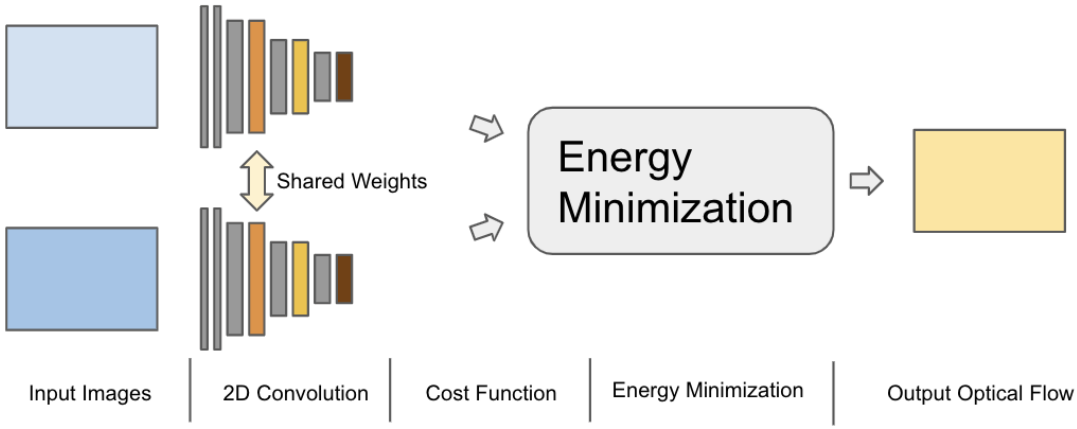


Figure 2.8: Using CNNs as a feature extractor in optical flow estimation

Like many subareas of computer vision, deep learning has made considerable impacts on the process of optical flow. Different from classical energy-based models, which formulate optical flow estimation as an energy minimization problem, the benefits of Convolutional Neural Networks (CNNs) over conventional methods have seen increased adoption in the context of motion estimation. Convolutional Neural Networks (CNNs) employing backpropagation on a large-scale image classification task Krizhevsky, Sutskever, and Hinton (2017) provided the foundation for CNNs to be used for optical flow estimation as well. As illustrated in figure 2.8, CNNs were employed as a feature extractor in early work Bailer, Varanasi, and Stricker (2017); Gadot and Wolf (2016) that

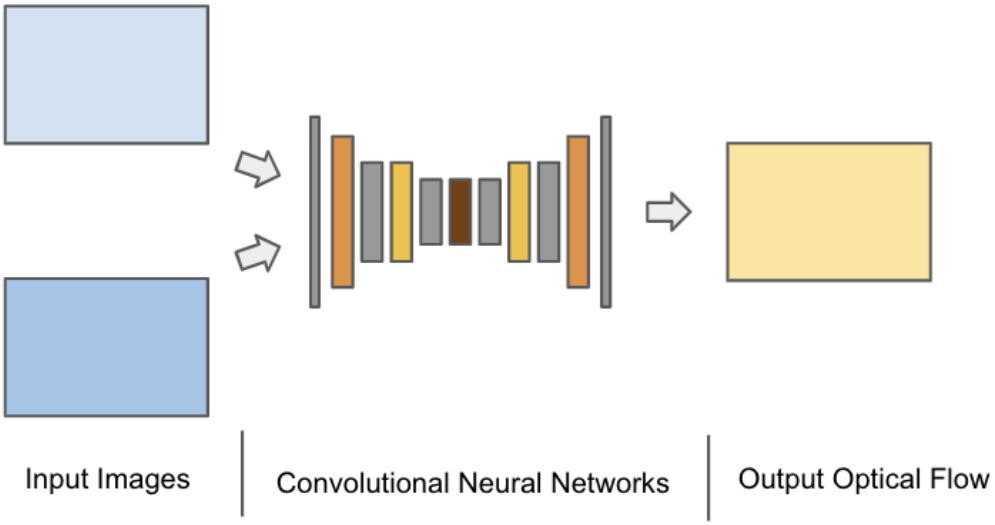


Figure 2.9: CNN regression architecture

applies them to optical flow, CNN-based feature encoders substitute the data term in the classical energy-based formulation. Instead of employing image intensities, image gradients, or other hand-crafted features, CNNs enable learning feature extractors to represent each pixel with a high-dimensional feature vector that combines a sufficient standard of distinctiveness and invariance, such as to appearance changes. Alternatively, shown in Figure 2.9, regression-based CNN architectures allow direct estimation of optical flow from a pair of input images and can be trained in an end-to-end fashion. CNNs are employed throughout the entire pipeline in the regression architectures to work as a function approximator, which effectively learns the relationship between the input images and the desired flow output given the labeled training dataset.

Optical Flow Warping

For CNNs that estimate the optical flow between a source image and a target image, the target image commonly shifts according to the optical flow field so as to try to match the input source image. The new target image is then fed to the following layers of the network, then the network in the stack focuses on the remaining increment between the image pair. The warping scheme (shown in Figure 2.10) is useful in video compression, and when an optical flow representation uses fewer parameters than an actual representation, the redundancy is significantly reduced.

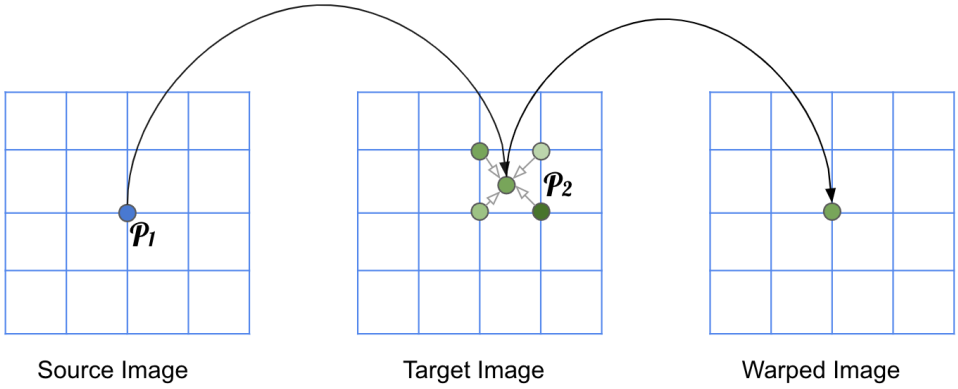


Figure 2.10: Optical Flow Warping

2.5 Structure-from-Motion

The goal of 3D modeling is to recover the 3D structure of the real world from its projection into 2D images. Structure-from-motion(SfM) recovers camera parameters, pose estimates, and sparse 3D scene geometry from image sequences. Modern methods usually solve the multiview SfM problem using bundle adjustment techniques, which aim to optimize a cost function known as the total reprojection error. The most widely used incremental pipeline is Bundler Agarwal, Snavely, Seitz, and Szeliski (2010). The majority of the SfM algorithms performs multiple bundle adjustments (BA) to rigid local structure and motion. However, some parts of the problem can be solved more efficiently by using optical flow. The input to the incremental reconstruction is the dense image matches, the outputs are the intrinsic and extrinsic calibration of the cameras along with the 3D pointcloud.

2.5.1 Camera Calibration

This thesis explores images collected using a perspective camera projection model, in which a 2D image plane receives light rays emitted by 3D scene points $X \in \mathbb{R}^3$. A pinhole camera is a camera without a lens but with a tiny aperture, it is used to simplify the projection process of perspective cameras. The idea of a pinhole camera is visualized in Figure 2.11, which the geometry of a pinhole camera with the local camera coordinate system defined by the axes (X, Y, Z), the scene point X is projected onto the image plane at x through the pinhole camera projection center

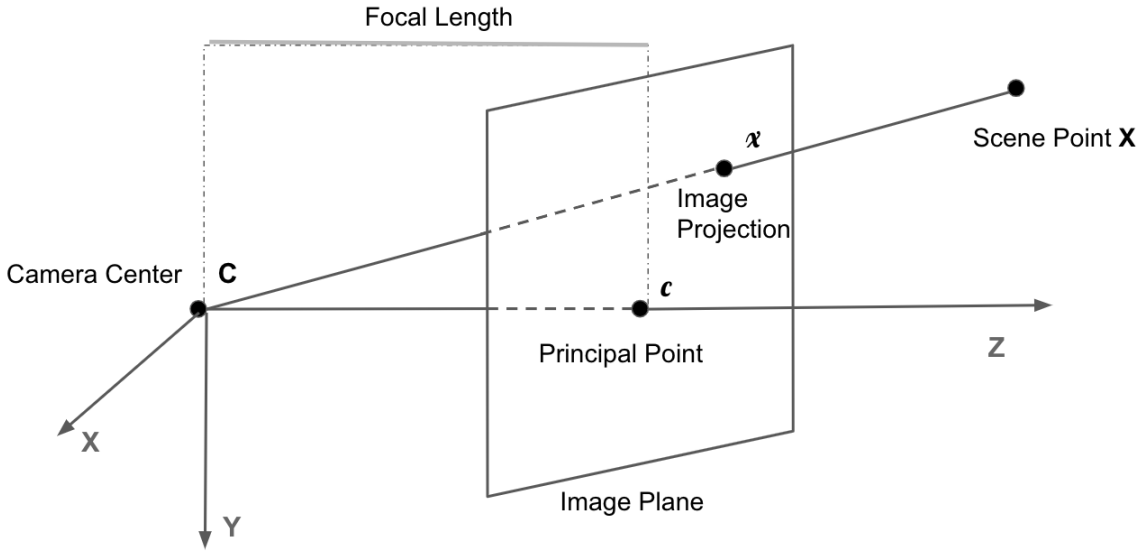


Figure 2.11: Model of a pinhole camera.

C, and the image plane is orthogonally offset to the plane (X, Y) by the distance of the focal length f. The geometry of the pinhole camera projection process is visualized in Figure 2.11. When using an ideal pinhole camera, all the captured light rays pass through a single center of projection $C \in \mathbb{R}^3$ (the aperture). Mathematically, this projection process can be formulated as:

$$P = K[R|t], t = -RC$$

where P is a 3×4 rank-3 matrix and defines the projection from the scene point $\hat{X}$ in homogeneous coordinates $X \in \mathbb{P}^3$ to an observation $X \in \mathbb{P}^2$ in the projective image plane. The 3×3 rotation matrix R and the translation vector t define the Euclidean transformation from world to camera coordinate system. These are known as the extrinsic camera calibration parameters, whereas the intrinsic camera calibration is encoded in the triangular matrix:

$$K = \begin{bmatrix} f_x & 0 & u \\ 0 & f_y & v \\ 0 & 0 & 1 \end{bmatrix}$$

where $(u, v) \in \mathbb{R}^2$ defines the location of the principal point and f_x and f_y is the horizontal and vertical focal length respectively.

Given 2D image observations x and their corresponding 3D scene points X, the 12 parameters of the projection matrix P can be estimated using the linear relation. Reordering the equations

with the direct linear transform allows to obtain the homogenous system of equations. This linear system can be solved from at least 6 2D image to 3D point correspondences, because every equation contributes 2 constraints for a total of 12 unknowns.

2.5.2 Triangulation

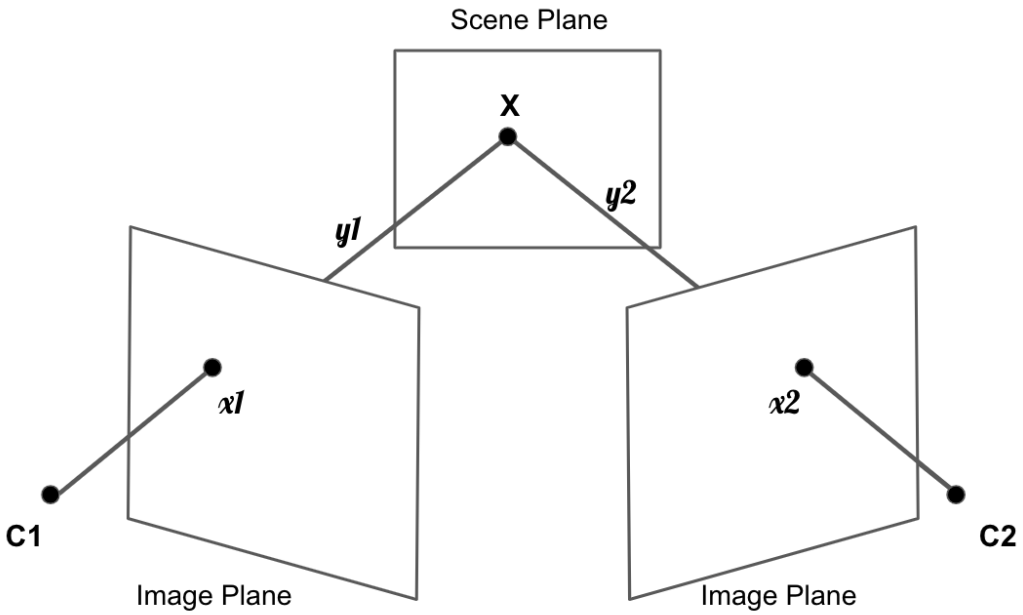


Figure 2.12: Triangulation of two viewing rays.

Triangulation is the process of triangulating the 3D position of 2D correspondences between several images. The input is the vectors of 2D points (the inner vector is per image), with the 3×4 projections matrices of each image, the output is the computed 3D points. With the 2D observations x in the image, the 3D structure X of the scene can be recovered by two points along the viewing ray $y = K^{-1}x \in \mathbb{P}^2$ are equal in projective space. As shown in Figure 2.12, point X from two corresponding image observations x_1 and x_2 by the intersection of their viewing rays y_1 and y_2. The 2D homography H is a projective transformation in $\mathbb{P}^2$ that preserves lines in projective space, which maps points x_1 on one plane to points x_2 on another plane. The homography matrix H has 8 degrees of freedom due to the projective ambiguity and can therefore be estimated from at least 4 image correspondences in two views.

2.5.3 Bundle Adjustment

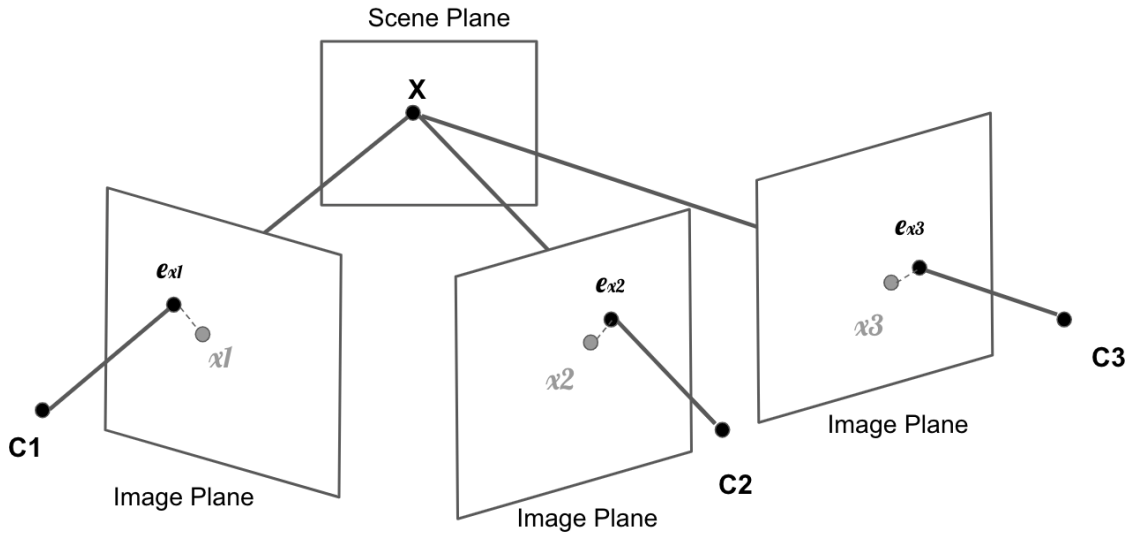


Figure 2.13: Overview of reprojection error

As shown in Figure 2.13, bundle adjustment is a common method for solving Structure-from-Motion problems. Bundle Adjustment (BA) is almost invariably used as the last step of every feature-based multiple view reconstruction vision algorithm to obtain optimal 3D structure and motion (i.e. camera matrix) parameter estimates.

Bundle adjustment can be defined as the problem of simultaneously refining the 3D coordinates describing the scene geometry, the parameters of the relative motion, and the optical characteristics of the camera(s) employed to acquire the images, according to an optimality criterion involving the corresponding image projections of all points. Provided with initial estimates, Bundle adjustment simultaneously refines motion and structure by minimizing the reprojection error between the observed and predicted image points. Perhaps the most popular variant of this method is the Sparse Bundle Adjustment which exploits the sparse nature of the matrix to efficiently store, process and solve for relatively large sets of parameters. This variant requires that all information about the matches, the 3D points corresponding to those matches, and the camera parameters are available in memory. When used with a sparse set of matches such as those produced by SIFT, this does not constitute a problem however, there is always an upper bound on the number of parameters one can solve for and is typically restricted to a finite set of sparse features. The second variant

of bundle adjustment uses an iterative approach where a non-linear optimization is used to solve for the unknown camera and structure parameters. Until recently this method also required that all information is available in memory which again limited the size of datasets one could process.

As illustrated in Figure 2.13, bundle adjustment is the joint refinement of scene structure X and camera calibration by minimizing the reprojection errors e_{xi} of the image observations x. Bundle adjustment involves the simultaneous optimization of 3D points and camera poses based on the reprojection error. Using an initial estimate for the camera poses from the decomposition of the fundamental matrix between pairs of images, the initial 3D points are estimated via triangulation. The optimization proceeds by updating the 3D points and camera poses such that the reprojection error E is minimized given by,

$E = \sum_i d_i(||Q(C_c, X_k), x_i||)^2$

where C_c are the camera parameters, X_k the points, and $Q(.,.)$ is a function which projects a 3D point onto the image plane corresponding to camera parameters C_c. d_i is a loss function that potentially down-weights outliers. A popular method for solving this type of problem is to store and factor the data as a sparse matrix or apply a non-linear optimization using Levenberg-Marquardt. Solving using a dense or sparse matrix requires N^2 memory space and has the complexity of $O(N^3)$ however for large-scale datasets it very often fails due to the large memory requirements imposed. On the other hand, solving using the iterative method has $O(N)$ time complexity and requires N memory space. However, the memory requirement can be reduced by computing the equation in batches.

2.5.4 Incremental Structure-from-Motion(SfM)

Incremental Structure-from-Motion(SfM) is one of the most popular strategies for the reconstruction of unordered photo collections, which have been proposed. Incremental SfM is the standard approach that initializes with a pair of images and adds one image at a time to grow the reconstruction by triangulation, filtering, and refinement using bundle adjustment to reduce accumulated errors. The initialization, image registration, and triangulation stages of an incremental sparse reconstruction pipeline is shown in Figure 2.14, Figure 2.15 and Figure 2.16 respectively.

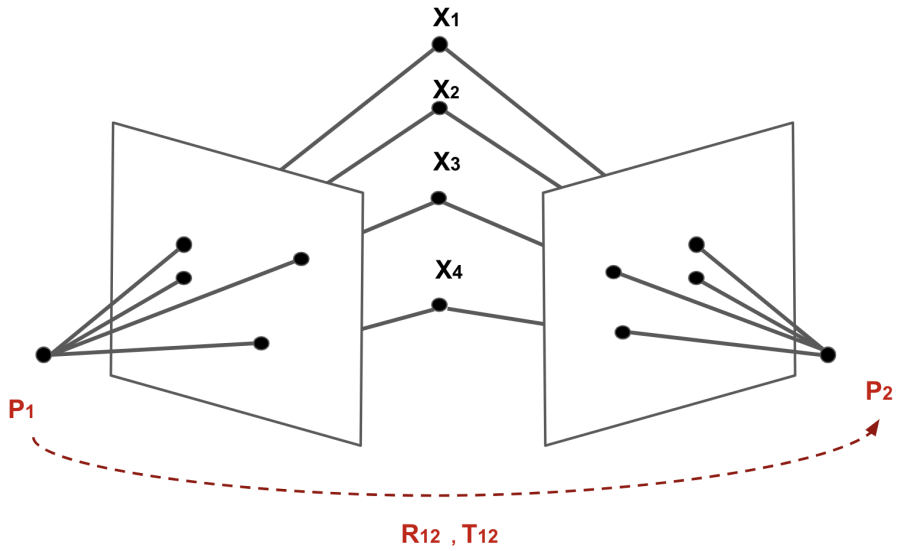


Figure 2.14: Overview of initialization in incremental Structure-from-Motion(SfM)

Photo Tourism Agarwal et al. (2010) presented an image-based modeling front-end that automatically computes the viewpoint of each photograph as well as a sparse 3D model of the scene and image-to-model correspondences. In particular, their SfM approach made several modifications to improve robustness over a variety of data sets. Including initializing new cameras using pose estimation, to help avoid local minima; a different heuristic for selecting the initial two images for SfM; checking that reconstructed points are well-conditioned before adding them to the scene; and using focal length information from image EXIF tags. The time complexity of incremental SfM is often known as $O(n^4)$ with respect to the number of cameras. By introducing multiple GPU to a BA strategy that provides a good balance between speed and accuracy, incremental SfM requires only $O(n)$ time on many major steps including BA, maintaining high accuracy by regularly re-triangulating the feature matches that initially fail to triangulate.

2.6 Surface Reconstruction

A surface mesh model is required for some practical uses as a pointcloud is an insufficient nor complete approximation. Because of the structure of the real scenes, mesh parameterization results in a compact representation for most scenarios. While a pointcloud requires multiple point

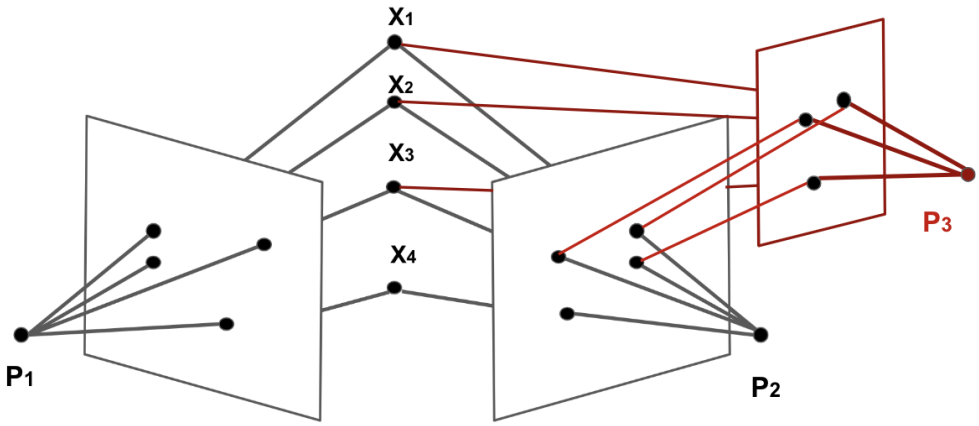


Figure 2.15: Overview of registration of image in incremental Structure-from-Motion(SfM)

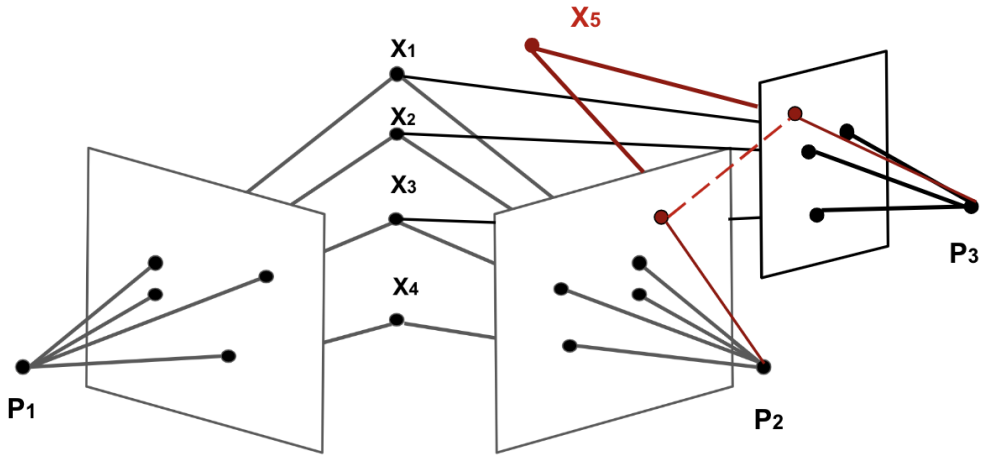


Figure 2.16: Overview of triangulation in incremental Structure-from-Motion(SfM)

instances to densely model the surfaces, a good surface mesh parameterization can theoretically model an entire object with a single entity. In most cases, a basic triangular mesh also may accurately represent the geometry of a scene with locally planar surfaces. When opposed to pointcloud renderings, creating surface texture maps from photos leads to improved rendering quality. Poisson Surface Reconstruction Kazhdan, Bolitho, and Hoppe (2006) is a widely used method for surface reconstruction. Given a set of 3D points, the oriented normals (denoted oriented points in the sequel) are sampled on the boundary of a 3D pointcloud, then it solves for an approximate indicator function, whose gradient best matches the input normals. The output scalar function, represented in

an adaptive octree, is then contoured using adaptive marching cubes.

2.7 End-to-End Deep Learning 3D Reconstruction

The goal of multi-view stereo (MVS) is to compute a depth (and normal) estimate for every pixel. The depth map refers to the depth of each pixel of the reference image, and pixel normal refers to a 3D plane in which the 3D point corresponding to a pixel in the reference image lies. This makes photometric values of corresponding regions/patches in two or more images of the same scene more similar through the accommodation of perspective projection. Depth maps can be directly projected into space by using extrinsic and intrinsic camera parameters to generate 3D point clouds. Given the estimated camera calibration, the challenge of retrieving a depth value for each pixel with a pair of images is simplified significantly, since the full and precise epipolar geometry between the views is already known. The multi-view stereo algorithms leverage the known epipolar geometry and compare every pixel in a reference image with all pixels along the corresponding epipolar line in the other image. In order to create more robust metrics of appearance similarity, a small patch around each pixel is commonly considered, the most similar patch along the epipolar line determines the location in the scene through triangulation. COLMAP Schönberger, Zheng, Frahm, and Pollefeys (2016) is one of the most widely used MVS pipelines, it is not deep learning based, but commonly used for comparison.

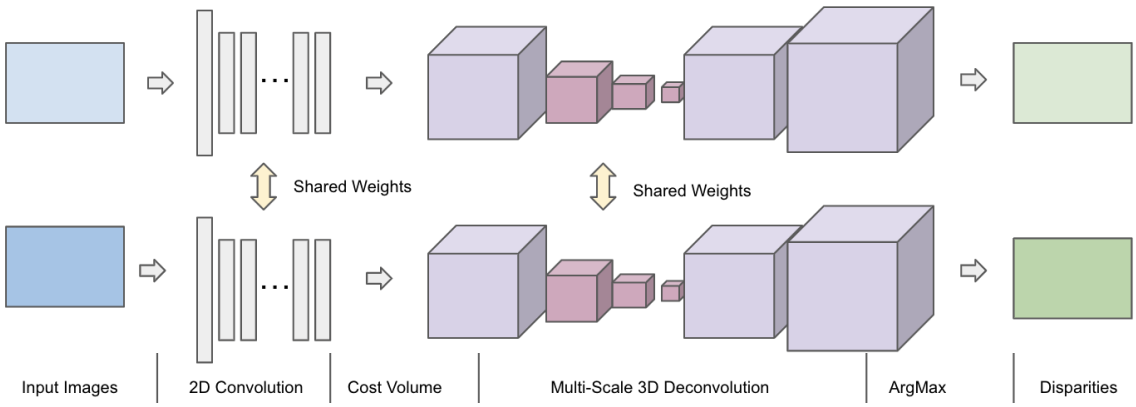


Figure 2.17: Overview of typical end-to-end deep stereo architecture.

The earliest end-to-end trainable networks for stereo-vision MVS normally employed convolutional neural networks (CNNs) to learn to regularize the cost volume. As the GC-Net Kendall et al. (2017) network structure shown in Figure 2.17, the network applies the soft $Argmin$ function to allow the model to regress sub-pixel disparity values from the disparity cost volume. As is standard, the features are generated via a 2D convolution network.

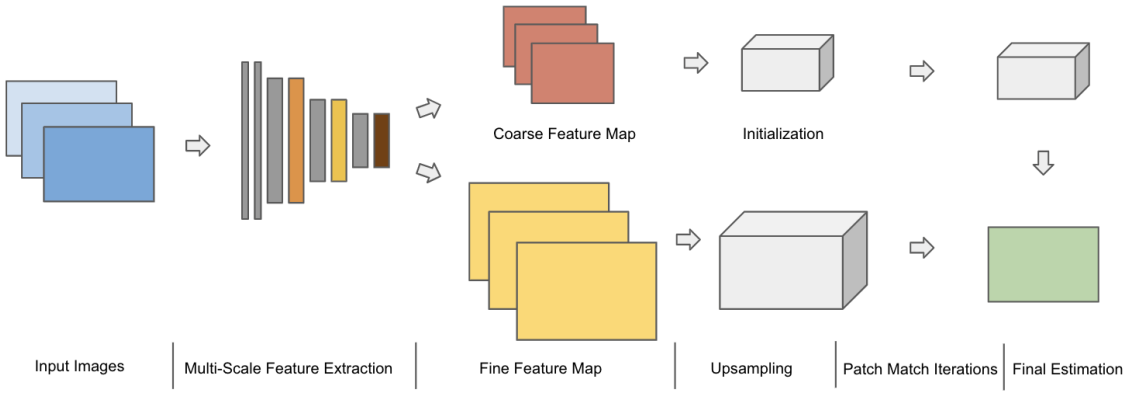


Figure 2.18: Overview of coarse-to-fine multi-view stereo network.

The later work PatchMatch-RL Lee, DeGol, Zou, and Hoiem (2021) is one of the representative coarse-to-fine neural networks. The authors extract multi-scale features using CNNs with shared weights, then perform coarse-to-fine estimation, with the correlation of feature maps at corresponding scales used to evaluate photometric costs and perform view selection. At the coarsest stage, pixelwise oriented points (depths/normals) are initialized and associated with hidden states per plane. Then, a series of PatchMatch iterations updates the points and hidden state maps.

The principles of the MVS neural networks are introduced as follows:

Plane Sweep

Plane sweep stereo Collins (1996) is an algorithm of multi-baseline stereo, in which a stack of planes with different depths corresponding to a reference image. It performs rectification of several cameras onto a common plane and resolves the problems with wide baselines and distortion after rectification. With homography, each target image is projected to the reference image for each depth plane, resulting in warped images. Then some metrics, such as Sum Squared Distance(SSD), Sum of Absolute Difference (sad), and Zero-mean Normalized Cross Correlation (ZNCC), are used to

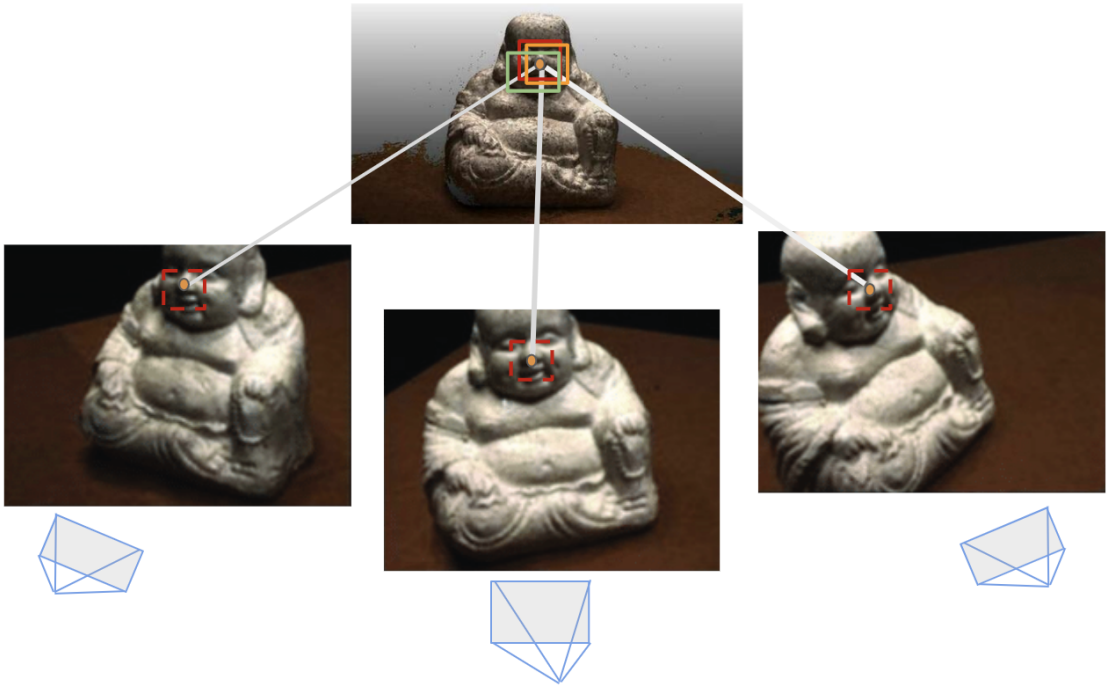


Figure 2.19: The concept of plane sweep stereo.

compare the reference image and each target image. The depth plane with the best match is selected. A visual representation is shown in Figure 2.19. As shown in Figure 2.19, the family of depth plane Π_m in the coordinate frame of the reference view is:

$\Pi_m = [\boldsymbol{n}_m^T - d_m]$

where $\boldsymbol{n}_m^T$ is the normal of the plane, and d_m is the depth from the reference camera. The mapping from the reference camera P_ref onto the plane Π_m and back to camera P_k is described by the homography introduced by the plane Π_m:

$H_{\Pi_m,P_k} = K_k(R_k - \boldsymbol{t}_k\boldsymbol{n}_m^t/d_m)K_{ref}^{-1}$

where K_k, K_{ref} is the camera intrinsic matrix and reference camera intrinsic matrix respectively, R_k is the camera rotation matrix, $\boldsymbol{t}_k$ is the transformation vector. The mapping from P_k to P_ref introduced by Π_m is the inverse homography H_{Π_m,P_k}^{-1}

Patch Match

Patch matching conceptually is a matching algorithm between a patch in one image to the approximate nearest one in another image. The Bruce-Force (i.e. exhaustive search) approach for

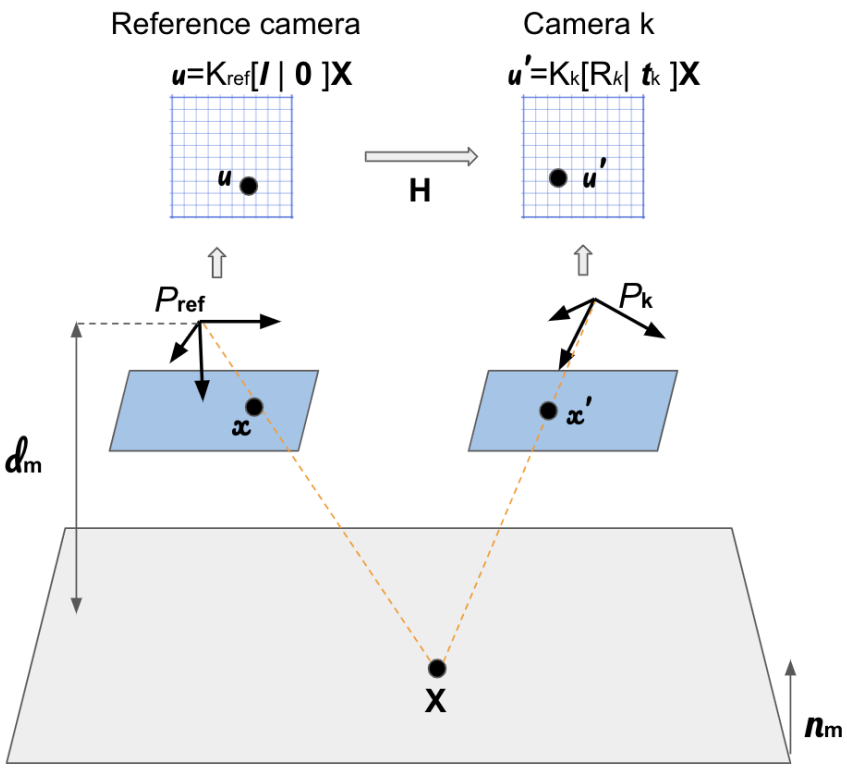


Figure 2.20: Overview of plane sweep algorithm.

such matching is — $\mathcal{O}(mM^2)$, where m and M are the number of pixels in the patch and image respectively. The M^2 comes from having to look at every scaling of the target image as the images need not be of the same scale. The phases of the randomized nearest neighbor algorithm Barnes, Goldman, Shechtman, and Finkelstein (2011) can be summarized as:

(a) patches initially have random assignments,

(b) the initial patch checks in the target image for the above and the left neighborhood, if the target patch improves the mapping, propagate the good matches,

(c) the patch searches randomly for improvements in the concentric neighborhoods.

The intuition behind the algorithm is that patches closer to the source image are also closer to the target image. Hence, it is worthwhile to check if a nearby neighbor has a better match. If so, take that. Step (c) prevents the algorithm from getting stuck in a local minimum by exploring random patches in a concentric route. Step (b) and (c) are called propagate and perturb respectively.

Cost Volume

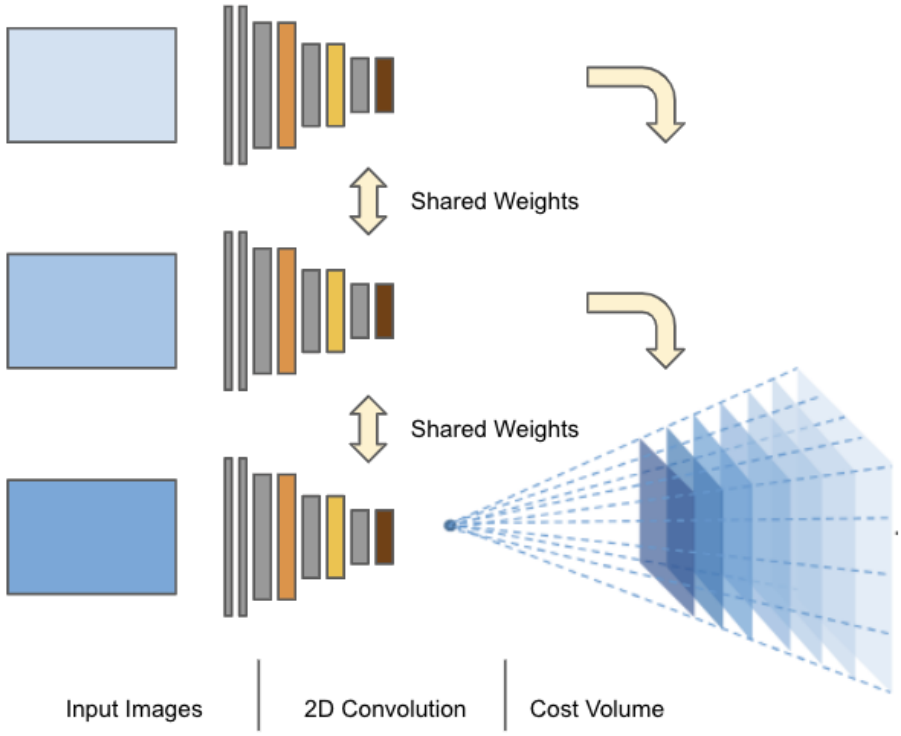


Figure 2.21: Overview of cost volume of multi-view stereo network.

Cost volume is a way to incorporate geometrical constraints and priors Kendall et al. (2017); Yang, Mao, Alvarez, and Liu (2020) into the deep learning MVS pipeline. The main concept is to use one of the images from the same scene as a reference and project the rest of the images onto it at a range of expected depths. As camera intrinsic and extrinsic matrices for each of the images are made available, the homography projection is available. This volume of $H \times W \times D$, where H, W, D are the image height, width, and depth, can be filled with a number of values that represent the cost or difference between a pixel in the reference image and the others. Figure 2.21 demonstrates an example of the cost volume pyramid of MVSNet Yao, Luo, Li, Fang, and Quan (2018). In this end-to-end deep learning architecture for depth map inference from multi-view images, the 3D cost volume is built upon the reference camera frustum via the differentiable homography warping. The cost volume regularization in the MVS context has a similar meaning as in general machine learning context — prevent overfitting, address the ill-posed problems, etc. In the simplest form, it is usually

represented as the minimization of

$$|A_z - y|^2 + \lambda |P_z|^2$$

where both A and P, called stabilizing function, are linear, the parameter λ determines the degree of regularization.

Chapter 3

Related Work

3.1 Data Input

The modeling of urban structures has been performed using several approaches, including procedural modeling, automatic creation mechanisms, and synthesis methods. There are various types of input data that could be considered sources for automatic urban reconstruction algorithms, such as imagery, and LiDAR. A principle goal of image-based reconstructions is to evoke a visceral sense of presence based on a collection of photographs of a scene. Because stereo systems use standard imaging components like cameras, they are potentially smaller, cheaper, and consume less power than systems using active devices. In addition, they can easily adapt to various scenes, including both outdoor and indoor, static or mobile environments. In a number of projects, the visual information has been recognized as a valuable source for large-scale urban reconstruction.

3.1.1 Active and Passive Acquisition Methods

Active devices usually follow the time-of-flight principle, i.e., the sensor sends out a pulse of light and measures the duration $\mathbf{T}$ required for the pulse to be reflected back. The distance between the sensor and the scene can be estimated as $(\mathbf{C} \times \mathbf{T}/2)$, where $\mathbf{C}$ is the light speed. It delivers semi-dense 3D pointclouds that are very precise, especially for long-distance acquisition.

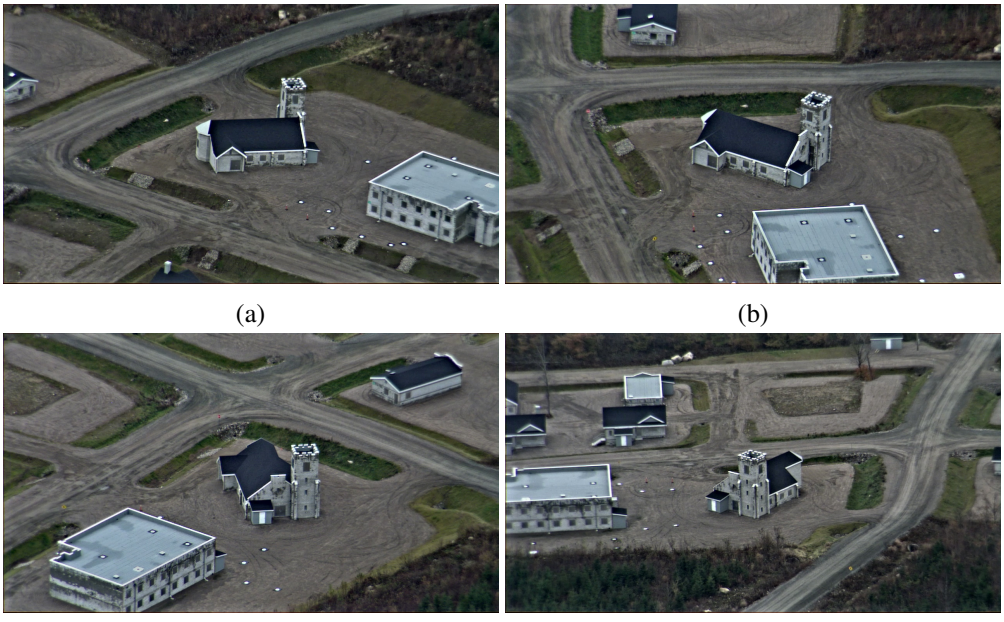
Other active devices use structured light to compute depth. Popular active devices such as Kinect, LIDAR, and SwissRanger Horaud, Hansard, Evangelidis, and Ménier (2016) can generate

depth-accurate point clouds with high quality. However, these devices have numerous drawbacks. LiDAR and SwissRanger are quite expensive; Kinect cannot function effectively in outdoor settings and has a limited effective range. LiDAR scanning technology is frequently used by land surveying offices and civil engineering firms for documentation purposes. Scanning devices are costly and not yet available for mass markets, making LiDAR data prohibitively expensive for scanning large cities and metropolitan areas.

Imagery (2D) is currently the most prevalent and accessible input source. Images obtained from the ground offer advantages in terms of the acquisition, storage, communication, and transmission. Stereo systems utilizing standard passive sensors, such as cameras, may be smaller, less expensive, and consume less power than systems employing active devices. In addition, they can easily adapt to a variety of scenes, both indoor and outdoor, static and dynamic. Stereo vision employs two (or more) images captured simultaneously by calibrated cameras. By matching the different images and computing the disparity between them, we can estimate the depth for each pixel in the camera frame, obtain the depth map, and create a 3D model. Figure 3.1 shows example frames of FMV, where images are taken from a helicopter circling around a church building.

3.1.2 Full Motion video (FMV)

Aerial images can be captured from different types of aircrafts, for example, the most common aircraft are drones (UAVs), helicopters, and airplanes. Full-motion video (FMV) is designed for synchronizing videos with maps, it is typically captured by a helicopter at an oblique aerial angle so that the rooftops and the facades of the buildings are visible in the images. The sensor systems collect camera pointing information, platform position and attitude, and other data and encode it into the video stream so that each video frame is associated with geopositional information. The ground sampling density is significantly higher than that of a satellite image, i.e. in the order of a few cms, and can vary according to the aircraft's flight height, depending on the area it is flying over. Full-motion video is utilized in numerous applications, including object recognition and tracking Pelapur et al. (2012); R. Wu et al. (2014), events and activities detection Kant (2012).



(a) (b)

(c) (d)

Figure 3.1: Example of Full Motion video (FMV) frames. Copyright of ©His Majesty the King in Right of Canada, as represented by the Minister of National Defence, 2022.

3.1.3 Wide-area motion imagery (WAMI)

Wide-area motion imagery (WAMI) is captured by an aircraft flying at over 10,000 ft and can cover areas of $10 - 20km^2$. The aircraft orbits around the area of interest during the flight, and an array of cameras captures and streams image data at about two frames per second. In the last decade, there have been significant developments in wide-area motion imagery (WAMI) Blasch and Seetharaman (2014). Wide area includes a geographical region larger than 50 square miles. Notionally, satellite imagery provides a large geographical area, but the challenges induced by satellite imagery include atmospheric transmission and pixel resolutions that limit surveillance resolutions to areas versus specific objects of interest. Satellite imagery could provide persistent surveillance, but is limited in that coverage consists of spot images versus constant monitoring. Furthermore, motion imagery includes visual imagery and video that does not include Synthetic Aperture Radar (SAR) imagery (typically done in spot model) in that resolutions are in the visual spectrum for object recognition. Finally, to further refine the historical developments, we are interested in motion analysis of moving targets.

Using the above details to refine the organization of developments, we note the rich history of

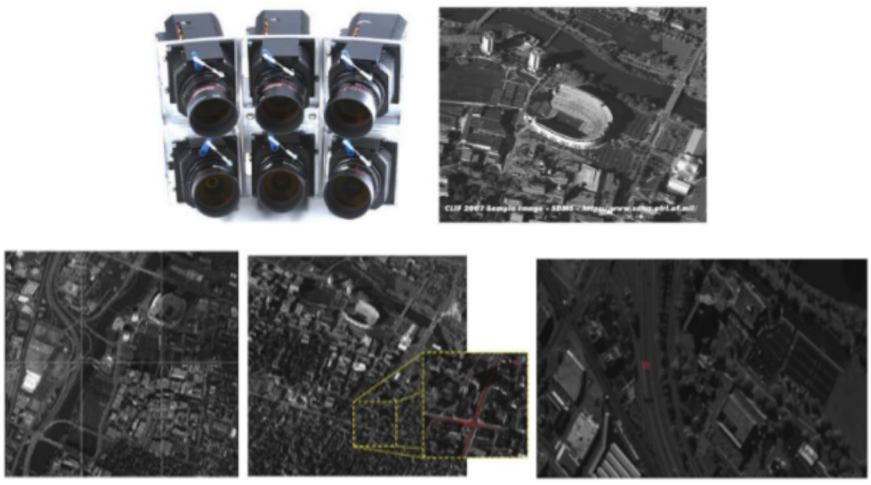


Figure 3.2: WAMI Summary: Sensors, Image Exploitation Blasch and Seetharaman (2014)

imagery surveillance, but focus on the aerial systems that motivated the developments in WAMI. In Blasch and Seetharaman (2014), the authors provide a brief history of WAMI hardware. An example of the WAMI sensors and image exploitation is shown in Figure 3.2. Some of the key challenges for the WAMI include low frame rates, extended camera coverage, multiple targets, weak target texture, and environment occlusions. The general advantages of using WAMI sensors include platform routing for persistent surveillance with constant coverage from overhead imagery, 3D processing for terrain analysis, and target tracking Porter, Fraser, and Hush (2010).

3.2 Image and Feature Descriptors

In image-based reconstruction, we are interested in gaining the geometric relationship between multiple images in order to reconstruct the 3D structure of the scene. One of the major problems is the robust and efficient geometric association of related input images. This process is typically solved in a two-stage approach: first, the contents of all the input images are independently described, then features are matched based on the descriptors. In the literature, the approaches can be categorized based on whether they describe the content globally (i.e., the image content) or locally (i.e., the region/pixel in the image). A global descriptor describes the whole image, while a local descriptor encodes an image based on the neighborhood of the given pixel. The local descriptor

extraction methods enable the reconstruction of the precise geometric relation between a pair of images.

3.2.1 Local Image Description

In image-based 3D modeling, we make use of image description approaches to describe the content of an image. Image matching based on local features has been extensively studied in the past decade. Local image description can be points, edges, or blobs, which are typically described using the image content in their direct neighborhood. Having interest points is better because it makes it possible to solve a geometric reconstruction problem in a reliable and efficient way. There are numerous different edge detection Canny (1986), corner detection Harris, Stephens, et al. (1988), and classic feature detection and their variations. Lowe D. G. Lowe et al. (1999) presented SIFT for extracting distinctive invariant features from images that can be invariant to image scale and rotation. Then it was widely used in image mosaics, matching, recognition, retrieval, etc. Among the standard hand-crafted feature descriptors, such as SIFT D. G. Lowe et al. (1999), SURF Bay et al. (2006), DAISY Tola, Lepetit, and Fua (2009) and ORB Rublee et al. (2011), SIFT D. G. Lowe et al. (1999) and its variants Arandjelovic and Zisserman (2012); Bursuc, Tolias, and Jégou (2015); Dong and Soatto (2015); Tuytelaars, Mikolajczyk, et al. (2008) is perhaps the most popular techniques of hand-crafted local features for more than a decade. For more efficient implementations, Rosten and Drummond Viswanathan (2009) proposed a point segment test for the features. Ke and Sukthankar Ke and Sukthankar (2004) used principal component analysis (PCA) to normalize gradient patches instead of histograms, that PCA-based local descriptors were also distinctive and robust to image deformations. Bay and Tuytelaars (2006) Bay et al. (2006) speeded up robust features and used integral images for image convolutions and Fast-Hessian detector(i.e. Harris-Laplace and the Difference of Gaussian (DoG) D. G. Lowe (2004)).

3.2.2 Learned Descriptors

In Chapter 2, we provided a comparative study of hand-crafted descriptors and learned local feature descriptors in the context of image matching. Learned descriptors are obtained as the intermediate representations of deep Convolutional Neural Networks(CNN). The most influential CNN

architectures in the state-of-the-art are thought to be AlexNet Krizhevsky et al. (2017), which competed in the Large Scale Visual Recognition Challenge (ILSVRC - Deng et al. (2009)) in 2012 and sparked many more papers employing CNNs and GPUs to accelerate deep learning. ImageNet Deng et al. (2009) is a dataset of over 15 million labeled high-resolution images with around 22,000 categories. ILSVRC uses a subset of ImageNet of around 1000 images in each of 1000 categories. In all, there are roughly 1.2 million training images, 50,000 validation images, and 150,000 testing images. The process of locating a discriminative embedding in a new space given an image patch is known as descriptor learning. PCA-SIFT Ke and Sukthankar (2004) applied principal component analysis (PCA) to embed a gradient image of each patch and Lepetit et al. Lepetit and Fua (2006) using randomized trees. The embedding process can also be used to turn existing descriptors into spaces with fewer dimensions (see Bursuc et al. (2015); Philbin, Isard, Sivic, and Zisserman (2010); Simonyan et al. (2014) for examples). More recently, DeepDesc Simo-Serra et al. (2015) learns compact discriminative feature point descriptors using a convolutional neural network (CNN). The approaches use a fixed set of filters and learn the pooling regions based on Convolutional Neural Networks (CNN), and this type of architecture is used in later work Balntas, Riba, Ponsa, and Mikolajczyk (2016); L.-C. Chen, Papandreou, Kokkinos, Murphy, and Yuille (2017). The methods discussed above compute the corresponding feature descriptor from an input image patch. They can therefore be utilized with any feature detector and are not restricted to a particular detector that provides the patch. By combining the DeepDesc descriptor with a Difference-of-Gaussians (DoG)-like detector, joint detector and descriptor learning Kushal and Agarwal (2012); D. G. Lowe (2004); Yi, Trulls, Lepetit, and Fua (2016) optimizes both the descriptor and the detector.

3.3 Feature Correspondences

In many computer vision applications, matching local image features is a vital step. It is fundamental for several stages of an image-based 3D modeling pipeline. In many applications, the quality of the initial feature matching stage has a significant impact on the overall system performance. Consequently, the computer vision community is very interested in learning which local feature descriptors offer the greatest ability to discriminate and the best performance in matching.

The objective of a matching algorithm is to establish as many precise point-wise correspondences, known as matches, as possible in order to discover shared visual content between two images.

3.3.1 Feature Matching

The goal of feature matching is to determine the correspondent keypoint pairs between the input frames using a distance metric. L_1 distance (i.e. Manhattan distance) and L_2 distance (i.e. Euclidean distance) are the most common distance metrics used for calculating the similarity between two descriptors. Fast Library for Approximate Nearest Neighbors (FLANN) Muja and Lowe (2009) and exhaustive search, commonly known as Brute Force (BF) matcher is a descriptor matcher that compares two sets of keypoint descriptors and generates a result that is a list of matches. The objective of Lowe's ratio test D. G. Lowe (2004), as one of the classic matching methods is to ensure that the best two distances are sufficient by matching the keypoints from the first image with those from the second image. The classic matcher works best for well-textured rigid objects but fails to match non-rigid objects and weakly textured regions. To handle the non-rigid deformations and repetitive textures, the deformation model Keysers, Deselaers, Gollan, and Ney (2007) proposed applying non-rigid 2D warping, Ecker et al. Ecker and Ullman (2009) present a pipeline to measure the similarity of small images, and Deep-Matching Revaud et al. (2016) computes dense correspondences relying on a hierarchical, multi-layer, correlational architecture. Another feature-based approach Wills, Agarwal, and Belongie (2006) calculated a non-rigid matching by robustly fitting smooth parametric models (homography and splines) to local descriptor matches in order to create dense correspondences between the images.

3.3.2 Optical Flow

The foundation of optical flow strongly resembles the original formulation of Horn and Schunck (HS) Horn and Schunck (1981). They combine a data term that assumes the constancy of some image properties with a spatial term that models how the flow is expected to vary across the image. A tremendous number of different robust functions have been explored; current best practices can be summarized as: coarse-to-fine estimation to deal with large motions Brox, Bruhn, Papenberg, and Weickert (2004) or high-order filter constancy Lempitsky, Rother, Roth, and Blake (2009) to

reduce the influence of lighting changes; warping with bicubic interpolation Roth, Lempitsky, and Rother (2009), median filtering after each incremental estimation step to remove outliers Wedel, Pock, Zach, Bischof, and Cremers (2009). Structure and motion can be computed either starting from the estimated flow or by inserting a suitable parameterization of the optical flow. Despite the recent success of learning-based approaches Dosovitskiy and Fischer (2015); Ilg et al. (2017), global energy-based methods are still the most accurate and robust techniques to solve the optical flow, especially in large displacement circumstances. Detailed related work will be reviewed in Chapter 4, where we present the proposed variational optical flow technique (HybridFlow) for dense feature matching.

3.4 Sparse Reconstruction

In most of the 3D reconstruction pipelines, there is the necessity of a two-step process to recover camera poses and sparse reconstruction using structure from motion (SfM), followed by multi-view stereo (MVS) to provide dense reconstruction.

3.4.1 Structure-from-Motion

Structure-from-Motion (SfM), aims at recovering camera parameters, pose estimates, and sparse 3D scene geometry from image sequences. Modern methods usually solve the multiview SfM problem using bundle adjustment techniques, which aim to optimize a cost function known as the total reprojection error. The most widely used incremental pipeline is Bundler Agarwal et al. (2010). The generic approach for these methods is to separately estimate the camera orientations based on the pairwise rotational measurements and then use these camera orientation estimates together with the pairwise translational measurements in order to solve for the camera locations. The majority of the algorithms described above perform multiple bundle adjustments (BA) to rigid local structure and motion. However, some parts of the problem can be solved more efficiently by using optical flow. Since image matching can be made more scalable, vocabulary tree techniques Nister and Stewenius (2006) are introduced to eliminate the searching time. Bundle adjustment can be optimized with sparse matrices Ceres Solver Google Inc. Agarwal, Mierle, and Team (2022) or using GPU

C. Wu et al. (2011). The number of variables can also be reduced by eliminating structure from the bundle adjustment Rodríguez, López-de Teruel, and Ruiz (2011). Some approaches use a divide-and-conquer approach on the epipolar graph to reduce computation graphs for efficient structure from motion Gherardi, Farenzena, and Fusiello (2010).

3.5 Dense Reconstruction

Passive stereo vision is an important 3D sensing technique that can produce dense point clouds with excellent depth resolution at high efficiency. It makes use of two (or more) images taken at the same time from separate calibrated cameras. By matching the features from different images and computing the disparity between them, we can estimate the depth for each pixel in the camera frame and thus build the depth map.

3.5.1 Multi-view Stereo (MVS)

Stereo vision is an important 3D sensing technique for producing dense point clouds, which allows for restoring the third dimension from multiple views. Stereo vision is a passive sensing technique for 3D measurement based on two or more images of a given scene. The multi-view stereo is more precise, robust, and provides a comprehensive structure of the scene. Most recent multi-view stereo vision approaches focus on dense correspondence, since this is required for most applications such as image-based rendering and robotic navigation. Most dense multi-view stereo algorithms follow the pipeline summarized in D. Scharstein Seitz, Curless, Diebel, Scharstein, and Szeliski (2006): given two or more images from separate cameras, we first compute the matching cost for each pair of pixels at a given disparity; and then, for each pixel, we aggregate the costs over its local neighborhood with the same disparity; finally, the disparity for each pixel is computed by performing the optimization over the aggregated values. Different choices can be made in each step of the pipeline, and this results in a rich category of various algorithms. For matching cost, different measures could be used, including squared intensity differences (SSD), sum of absolute intensity differences (SAD) Asari (2013), truncated quadratics and contaminated Gaussians Groenert and Bryski (2009) and mutual information Edelberg and Daniel (2012). A.Locher Locher, Perdoch,

and Van Gool (2016) enables immediate feedback on the reconstruction process in a user-centric scenario. With increasing processing time, the model improves in terms of resolution and accuracy. The algorithm explicitly handles input images with varying effective scales and creates dense point clouds. L. Schonberger Schönberger et al. (2016) core contributions are the joint estimation of depth and normal information, pixel-wise view selection using photometric and geometric priors, and a multi-view geometric consistency term for the simultaneous refinement and image-based depth and normal fusion.

3.5.2 Deep learning Methods

In contrast to procedural sparse-to-dense reconstruction pipelines, network-based techniques usually produce depth maps of the images. Based on the variation in the known information in the training datasets, we review the deep learning models in two categories: Structure from Motion (SfM) and Multi-view stereo(MVS), where camera poses are estimated in SfM.

Deep Structure from Motion

With the advancement of deep learning, numerous scholars have lately investigated neural network-based approaches related to Structure from Motion (SfM). Most of the methods can be categorized into two types: (a) depth and camera pose regression jointly optimized Yin and Shi (2018); T. Zhou, Brown, Snavely, and Lowe (2017), and (b) the camera pose and the depth are inferred from the image pair and are iteratively refined via multi-view geometry Tang and Tan (2018); Teed and Deng (2018). The joint optimization of monocular depth and pose regression models consists of a depth estimation network and a pose regression network. They often use the priors from the training data to predict depth for just one image. When the camera pose and the depth are inferred from the image pair, two (or more) images are required to estimate depth maps and camera poses at test time. As a result, most supervised deep learning methods fall into this category. Different from networks that estimate depth from a single image, DeMoN Ummenhofer et al. (2017) is the first deep network that has learned to estimate depth and camera motion from two unconstrained images; it takes advantage of the motion parallax, a powerful cue that generalizes to various types

of scenes and enables the estimation of ego-motion. Deepv2d Teed and Deng (2018) splits the camera posture from depth estimation, updating them iteratively while reducing geometric reprojection errors. DeepSFM Wei, Zhang, Li, Fu, and Xue (2020) initiates the pose estimation from DeMoN Ummenhofer et al. (2017) and samples nearby pose hypotheses to perform bundle adjustment of both poses and depth.

Deep Multi-view Stereo

Multi-view stereo (MVS) aims at recovering the dense 3D structure of a scene from a set of calibrated images. Instead of camera pose registration and optimization, MVS approaches are focused on patch representations, matching, and regularization. By using Deep Convolutional Neural Network (CNN), the performance of MVS can be further improved. Learning a similarity measurement between small image patches for matching cost computation was proposed in Hartmann, Galliani, Havlena, Van Gool, and Schindler (2017); Zbontar, LeCun, et al. (2016). SurfaceNet Ji, Gall, Zheng, Liu, and Fang (2017) constructs Colored Voxel Cubes (CVC) in preparation, combining all camera and image pixel color information into a single volume for the network's input. MVSNet Yao et al. (2018) proposed an end-to-end deep learning architecture for generating depth maps that use the differentiable homography warping operation to implicitly encode camera geometries in the network to build the 3D cost volumes. A later work, RMVSNet Yao et al. (2019) uses Convolutional GRU for cost volume regularization and to avoid using memory-intensive 3D CNNs. In contrast, Point-MVSNet R. Chen, Han, Xu, and Su (2019) employs a coarse-to-fine strategy in which a low resolution depth map is initially predicted, and then the depth map is repeatedly upsampled and refined. The coarse-to-fine strategy achieves state-of-the-art 3D reconstruction quality with higher efficiency and overcomes the memory problem in MVS.

Chapter 4

Motion Estimation for Large Displacements and Deformations

The first step in the pipeline is finding dense correspondences between pairs of images. This chapter presents the paper "Motion Estimation for Large Displacements and Deformations" Q. Chen and Poullis (2022b), published by Nature Scientific Reports 2022.

4.1 Abstract

Large displacement optical flow is an integral part of many computer vision tasks. Variational optical flow techniques based on a coarse-to-fine scheme interpolate sparse matches and locally optimize an energy model conditioned on colour, gradient and smoothness, making them sensitive to noise in the sparse matches, deformations, and arbitrarily large displacements. This chapter addresses this problem and presents HybridFlow, a variational motion estimation framework for large displacements and deformations. A multi-scale hybrid matching approach is performed on the image pairs. Coarse-scale clusters formed by classifying pixels according to their feature descriptors are matched using the clusters' context descriptors. We apply a multi-scale graph matching on the finer-scale superpixels contained within each matched pair of coarse-scale clusters. Small clusters that cannot be further subdivided are matched using localized feature matching. Together, these

initial matches form the flow, which is propagated by an edge-preserving interpolation and variational refinement. Our approach does not require training and is robust to substantial displacements and rigid and non-rigid transformations due to motion in the scene, making it ideal for large-scale imagery such as aerial imagery. More notably, HybridFlow works on directed graphs of arbitrary topology representing perceptual groups, which improves motion estimation in the presence of significant deformations. We demonstrate HybridFlow's superior performance to state-of-the-art variational techniques on two benchmark datasets and report comparable results with state-of-the-art deep-learning-based techniques.

Key Words – optical flow, variational energy minimization, dense correspondence, large displacement, coarse-to-fine

4.2 Introduction

Dense motion estimation from optical flow is an essential component in many diverse computer vision applications ranging from autonomous driving Y. Wang et al. (2019), multi-object tracking and segmentation Porzi et al. (2020), action recognition Piergiovanni and Ryoo (2019), to video stabilization Yu and Ramamoorthi (2020), to name a few. Consequently, optical flow estimation directly contributes to the performance and accuracy of these applications.

Research in dense motion estimation techniques has been ongoing since the 1950s when Gibson first proposed it in Gibson (1950). Despite the active research, to this day, the estimation of optical flow remains an open research problem. This is primarily attributed to the following two challenges: occlusions and large displacement.

Occlusions can appear in several forms; self-occlusion, inter-object occlusion, or background occlusion. Typical solutions based on a variational approach employ a robust penalty function, and regularizers that aim to reduce the occlusion errors Hur and Roth (2019); Luo et al. (2019). However, they still fail in cases where the pixels vanish between consecutive frames. More recently, many deep-learning-based techniques were proposed Bar-Haim and Wolf (2020); P. Liu, Lyu, King, and Xu (2019). In many cases where ground truth is available, their performance surpasses that of variational techniques on benchmark datasets; however, applying these networks on real image

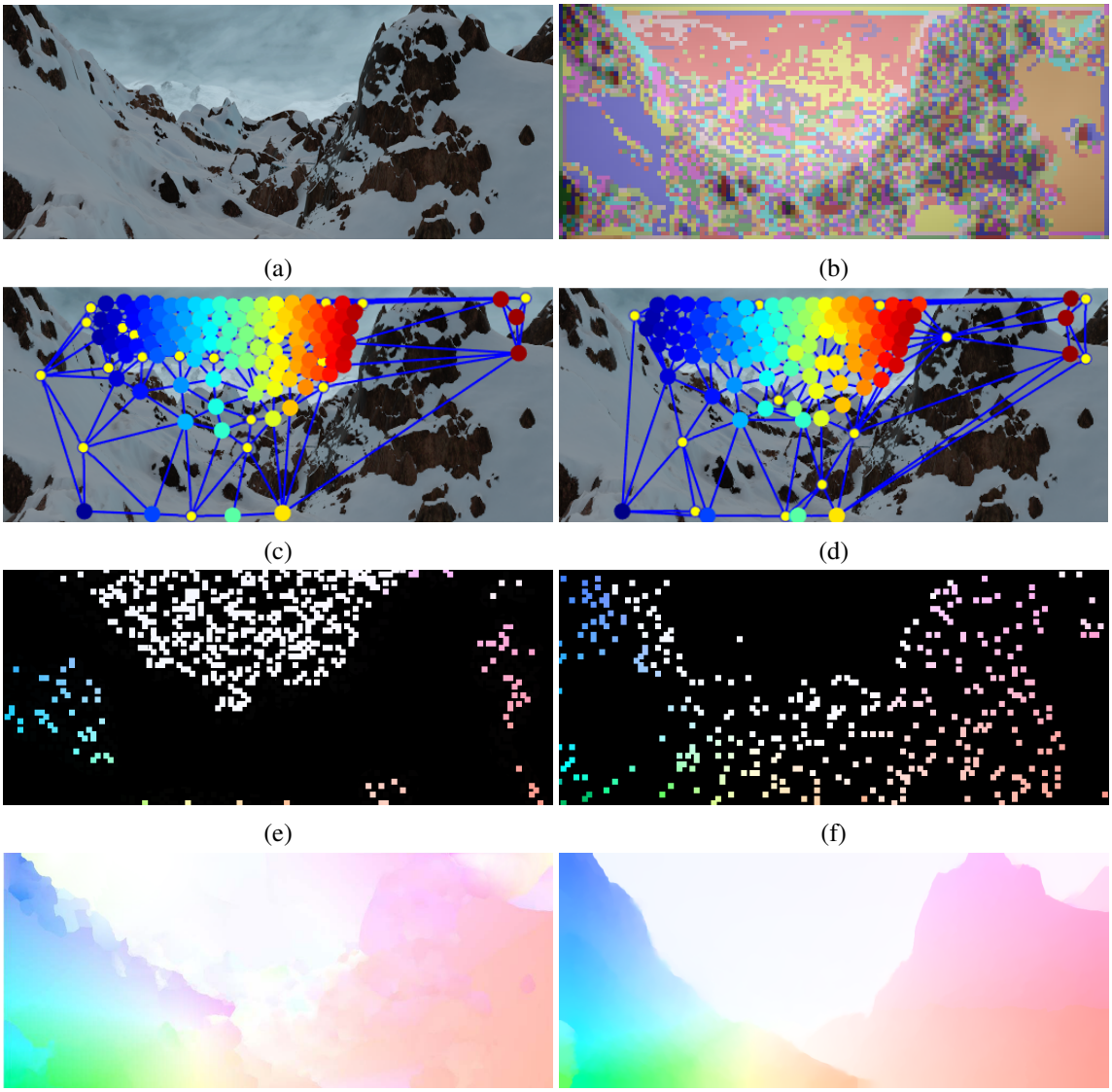


Figure 4.1: (a) Input image frame. (b) Coarse-scale clusters from pixels' feature descriptors. (c) Color-coded graph matches of one coarse-scale cluster; first frame (a). (d) Color-coded graph matches for (c); second frame. (e) Motion vectors from graph-matching of superpixels at finest-scale. (f) Motion vectors from pixel feature matching in small clusters. (g) Interpolated flow from the combined initial motion vectors (e) + (f). (h) Final optical flow after variational refinement. Average Endpoint Error (EPE) = 0.157. Note: The pixels in (c) and (d) are magnified by 10×10 for clarity in the visualization.

sequences is a non-trivial task that requires re-training, fine-tuning and often manual annotation.

On the other hand, for large displacements, solutions follow a coarse-to-fine model that introduces additional errors due to the coarse scales' upsampling and interpolation. To alleviate some of

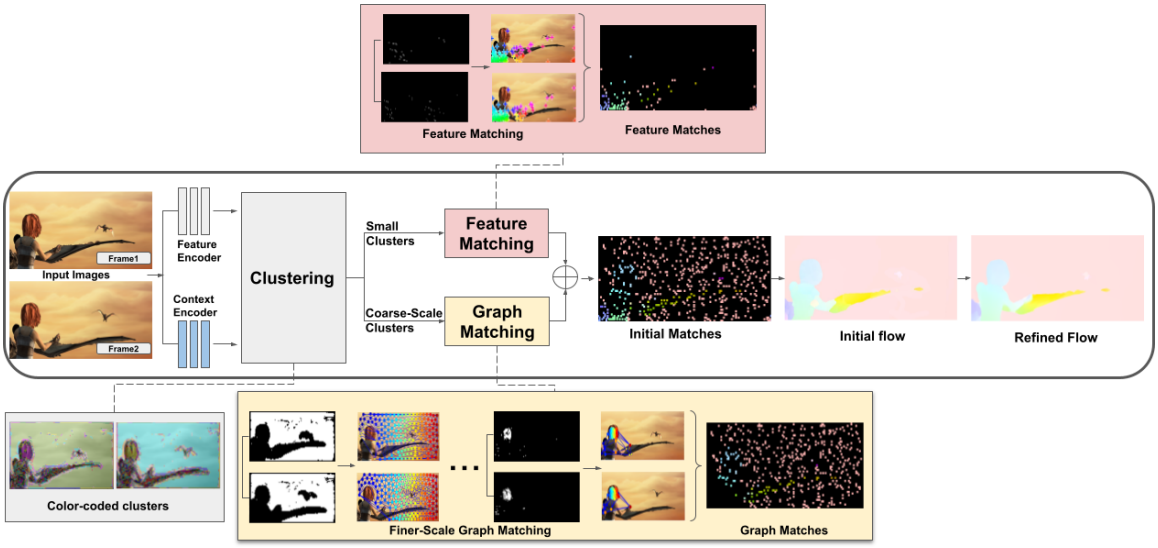


Figure 4.2: HybridFlow: A multi-scale hybrid matching approach is performed on the image pairs. Uniquely, HybridFlow, leverages the strong discriminative nature of feature descriptors, combined with the robustness of graph matching on arbitrary graph topologies. Coarse-scale clusters are formed based on the pixels' feature descriptors and are further subdivided into finer-scale SLIC superpixels. Graph matching is performed on the superpixels contained within the matched coarse-scale clusters. Small clusters that cannot be further subdivided are matched using localized feature matching. Together, these initial matches form the flow, which is propagated by an edge-preserving interpolation and variational refinement.

the interpolation errors, Revaud et al. Revaud et al. (2015) proposed EpicFlow, an edge-preserving interpolation of sparse matches used to initialize the optical flow motion estimation in a variational approach. Several techniques employing EpicFlow have since been proposed Hu et al. (2017); Hu, Song, and Li (2016), which address the sensitivity to noise in the sparse matches. The result is reduced interpolation errors in the estimated optical flow at the cost of over-smoothing the fine structures and failure to capture small-scale and fast-moving objects in the image. Thus, the accuracy of the initial sparse matches has a detrimental effect on the accuracy of the optical flow.

This chapter presents HybridFlow (Figure 4.2), a robust variational motion estimation framework for large displacements and deformations based on multi-scale hybrid matching. Uniquely, HybridFlow leverages the strong discriminative nature of feature descriptors, combined with the robustness of graph matching on arbitrary topologies. We classify pixels according to the $argmax$ of their context descriptor and form coarse-scale clusters. We follow a multi-scale approach, and

fine-scale superpixels resulting from the perceptual grouping of pixels contained within the parent coarse-scale cluster form the basis of subsequent processing. Graph matching is performed on the graphs representing the fine-scale superpixels by simultaneously estimating the graph node correspondences based on the first and second-order similarities and a smooth non-rigid transformation between nodes. Graph matching is an NP-hard problem; thus, the graphs' factorization into Kronecker products ensures tractable computational complexity. This process can be repeated at multiple scales to handle arbitrarily large images. At the finest-scale, the pixels' feature descriptors are matched based on their $\mathcal{L}_2$ distance. Pixel-level feature matching is also performed on clusters that are too small to be subdivided into superpixels. We combine both sets of pixel matches to form the initial sparse motion vectors from which the optical flow is interpolated. Finally, variational refinement is applied to the optical flow. HybridFlow is robust to large displacements and deformations and has a minimal computational footprint compared to deep-learning-based approaches. A significant advantage of our technique is that using multi-scale graph matching reduces the computational complexity from $\mathcal{O}(n^2)$ to $\sum_{i=0}^{k} \mathcal{O}(k^2)$ where k is always smaller than the superpixel size $|s|$ and significantly smaller than n, i.e. $k < |s| << n$. Our experiments demonstrate the effectiveness of our technique in optical flow estimation. We evaluate HybridFlow on two benchmark datasets (MPI-Sintel Butler et al. (2012), KITTI-2015Menze et al. (2015)) and compare it against state-of-the-art variational techniques. Hybridflow, outperforms all other variational techniques and, on average, gives comparable results with deep-learning-based methods.

To summarize, our contributions are:

- A hybrid matching approach that uniquely combines the robustness of feature detection and matching with the invariance to rigid and non-rigid transformations of graph matching. The combination results in high tolerance to large displacements and deformations when compared to other techniques.

- An objective function based on first and second-order similarities for matching graph nodes and edges, which results in improved matching as showcased by our experiments.

- A complete variational framework for estimating optical flow that does not require training and is robust to large displacements and deformations caused due to motion in the scene while

providing superior performance to state-of-the-art variational techniques and comparable performance to state-of-the-art deep-learning-based techniques on benchmark datasets.

4.3 Related Work

Optical flow is a 2D vector field describing the apparent motion of the objects in the scene. This optical flow field can be very informative about the relations between the viewers' motion and the 3D scene.

Over the years, many techniques have been proposed following the predominant way of estimating optical flow using variational methods Horn and Schunck (1981). The optical flow is estimated via optimization of an energy model conditioned on image brightness/colour, gradient, and smoothness. This energy model fails when dealing with large displacements due to motion in the scene because its solution is approximate and locally optimizes the function.

To address this challenge, Anandan Anandan (1989) proposed a coarse-to-fine scheme. Coarse-to-fine techniques upsample and interpolate the flow from the finer-scale of the pyramid to the coarser. These techniques can deal with large displacement; however, it comes at the cost of over-smoothing any fine structures and failing to capture small-scale and fast-moving objects.

At the same time, researchers explored the integration of feature matching in optical flow estimation. Revaud et al. Revaud et al. (2016) recently presented one of the most promising variational techniques where a HOG descriptor was used as a feature matching term in the energy function. Their technique can deal with deformations and is robust to repetitive textures. In subsequent work, the authors proposed EpicFlow, which performs a sparse-to-dense interpolation on the correspondences and estimates optical flow while preserving edges Revaud et al. (2015). Hu et al. Hu et al. (2017) built upon this work and proposed a robust interpolation technique to address the sensitivity of EpicFlow to noise in the initial matches by enforcing matching neighbourhood flow in the two images and fitting an affine model to the sparse correspondences. Up to now, this improvement produced superior performance than the previous best, which was based on a coarse-to-fine technique using PatchMatch Hu et al. (2016).

More recently, several techniques were proposed based on convolutional neural networks (CNN).

These estimate the optical flow in an end-to-end fashion using supervised learning Ilg et al. (2017); Ranjan and Black (2017); Sun, Yang, Liu, and Kautz (2018) or unsupervised learning Meister, Hur, and Roth (2018); Z. Ren et al. (2017); Yin and Shi (2018). One of the recent top-performing CNN-based approaches is SelFlow P. Liu et al. (2019). SelFlow is a self-supervised learning approach for optical flow that, until lately, produced the highest accuracy among all unsupervised learning methods. The authors achieved this by creating synthetic occlusions from perturbing superpixels. The current state-of-the-art CNN-based technique is RAFT Teed and Deng (2021), in which per-pixel features are employed in a deep network architecture of recurrent transforms. RAFT and its variants such as GMA Jiang, Campbell, Lu, li, and Hartley (2021) currently achieve the best performance reporting the lowest average endpoint error for all significant optical flow benchmark datasets.

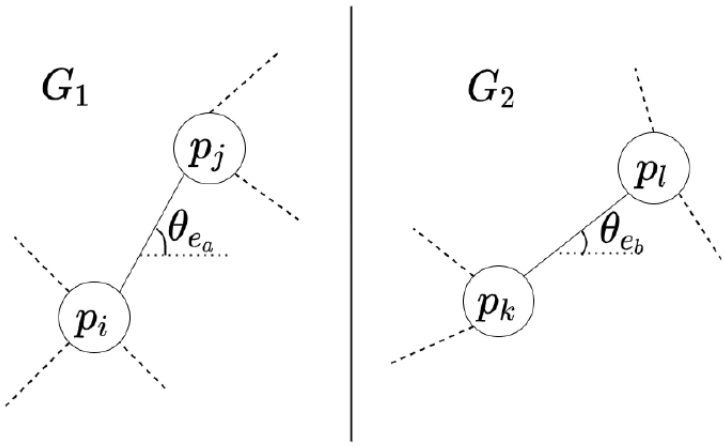
Currently, the average endpoint error (AEE/EPE) reported on Sintel-final for the top-performing deep-learning technique (CRAFT) is 2.424, and for the top-performing variational technique (Hybridflow -ours) is 5.121; a difference of fewer than 2.7 pixels over the entire imageset of 562 images of 1024x436. Although deep learning techniques beget superior performance to the variational methods on benchmark datasets for which ground truth is available, they are unusable on real image sequences that seldom have associated ground truth, and training and fine-tuning become impossible. Moreover, even in cases where ground-truth may be available, the training and fine-tuning are time-consuming, offline operations that render them unsuitable in scenarios requiring real or interactive time performance.

For these reasons, we propose a variational optical flow technique that is independent of the content of the image sequences and does not impose additional requirements for training and fine-tuning. Our method follows a hybrid approach for matching to eliminate errors in the initial sparse matches introduced from large displacements and deformations. HybridFlow leverages the strong discriminative nature of feature descriptors combined with the robustness of deformable graph matching. In contrast to variational state-of-the-art, which employs a regular grid structure in their coarse-to-fine matching scheme, HybridFlow operates at only a single image scale and multiple scales of clustering, eliminating over-smoothing and handling small-scale and fast-moving objects better. More notably, our method does not restrict deformations by enforcing smooth neighbourhood matching but instead employs deformable graph matching, which allows for rigid and non-rigid

transformations between neighbouring superpixels.

4.4 Graph Model and Matching

Model. A graph $G = \{P, E, T\}$ consists of nodes P inter-connected with edges E. A node-edge incidence matrix T specifies the topology of the graph G. The nodes are represented in matrix form as $P = [\vec{p_1}, \vec{p_2}, \ldots, \vec{p_N}[\in \mathbb{R}^{dim(\vec{p}) \times N}$, where $dim : \vec{v} \longrightarrow \mathbb{R}$ is a function that returns the cardinality of a vector $\vec{v}$. Similarly, the edges are represented in matrix form as $E = [\vec{e_1}, \vec{e_2}, \ldots, \vec{e_M}[\in \mathbb{R}^{dim(\vec{e}) \times M}$. An edge-weight function $w : E \times E \longrightarrow \mathbb{R}$ assigns weights to edges. Given the above definitions, the incidence matrix is defined as $T \in \{0, 1\}^{N \times M}$ where $T_{(i,k)} = T_{(j,k)} = 1$, if an edge $e_k \in E$ connects the nodes $p_i, p_j \in P$, otherwise it is set to 0.



$$\mathbf{K}_{(p_i p_j, p_k p_l)} = \lambda^E(e_a, e_b)$$
$$\mathbf{K}_{(p_i p_i, p_k p_k)} = \lambda^P(p_i, p_k) \quad \mathbf{K}_{(p_i p_i, p_l p_l)} = \lambda^P(p_i, p_l)$$
$$\mathbf{K}_{(p_j p_j, p_k p_k)} = \lambda^P(p_j, p_k) \quad \mathbf{K}_{(p_j p_j, p_l p_l)} = \lambda^P(p_j, p_l)$$

Figure 3. Two nodes in G_1 and G_2. The element values in $\mathbf{K}$ are calculated according to Equations 11 and 12.

Matching. Matching two graphs $G_1 = \{P_1, E_1, T_1\}$ and $G_2 = \{P_2, E_2, T_2\}$ is an NP-hard problem for which exact solutions can only be found if the number of nodes and edges are significantly small e.g. $N, M < 15$. Proposed solutions typically formulate graph matching as a Quadratic Assignment Problem(QAP) and provide an approximation to the solution Dokeroglu, Sevinc, and Cosar (2019). This requires the calculation of two affinity matrices: $A^P_{1,2} \in \mathbb{R}^{N \times N}$ which encodes the similarities between nodes in G_1 and G_2, and $A^E_{1,2}\mathbb{R}^{M \times M}$ which encodes the similarities between edges in G_1 and G_2. The functions $\lambda^P : P \times P \longrightarrow \mathbb{R}$ and $\lambda^E : E \times E \longrightarrow \mathbb{R}$ measure the similarities between nodes and edges, respectively. Therefore for two corresponding nodes $p_i \in P_1$ of G_1 and $p_k \in P_2$ of G_2, the node affinity matrix element is $A^P_{i,k} = \lambda^P(p_i, p_k)$. Similarly, for edges $e_a \in E_1$ of G_1 and $e_b \in E_2$ of $G2$ the edge affinity matrix element is $A^E_{a,b} = \lambda^E(e_a, e_b)$.

Given the above definitions, the solution to matching G_1 and G_2 is equivalent to finding the correspondence matrix $C_{1,2} \in \{0,1\}^{N_1 \times N_2}$ between the nodes of G_1 and G_2, that maximizes,

$$\underset{C_{1,2} \in \{0,1\}^{N_1 \times N_2}}{\arg\max} \mathbf{1}^T_{C_{1,2}} \mathbf{K} \mathbf{1}_{C_{1,2}} \tag{8}$$

where $\mathbf{1}_{C_{1,2}} \in \{0,1\}^{N_1 \times N_2}$ is the characteristic function, and $\mathbf{K} \in \mathbb{R}^{N_1 N_2 \times N_1 N_2}$ is a composite affinity matrix that combines the node affinity matrix $A^P_{1,2}$ and the edge affinity matrix $A^E_{1,2}$. The element of $\mathbf{K}((p_i p_j)_1, (p_k p_l)_2)$ for the nodes $p_i, p_j \in P_1$, $p_k, p_l \in P_2$, and the edges connecting these nodes $e_a \in E_1$, $e_b \in E_2$ respectively, is calculated as,

$$K((p_i p_j)_1, (p_k p_l)_2) = \begin{cases} \lambda^P(p_i, p_k) & \text{if } p_i = p_j \text{ and } p_k = p_l, \\ \lambda^E(e_a, e_b) & \text{if } p_i \neq p_j \text{ and } p_k \neq p_l, \\ 0 & \text{otherwise} \end{cases} \tag{9}$$

An example is shown in Figure 4.4. Intuitively, if the two nodes considered in each graph are co-located, i.e. there is *no edge* connecting them, then the element's value is the similarity of the function $\lambda^P(.,.)$ for the nodes. If the two nodes are different, i.e. there *is* an edge connecting them, then the element's value is the similarity of the function $\lambda^E(.,.)$ for the connecting edges; otherwise,

it is set to 0.

4.5 Method

Figure 4.2 and Algorithm 1 summarize the steps of the proposed technique. HybridFlow is the refined flow resulting from the interpolation of the combined initial flows calculated from the sparse graph matches from superpixels and feature matches of pixels in small clusters, as explained below.

Algorithm 1: HybridFlow

Result: Optical flow $\mathcal{O}$ between image-pair I_1, I_2
1. initialize optical flow $\mathcal{O} = \{\}$;
2. pixel classification (Eq. 10), clustering (Sec. 4.5.1), and matching of $\{\mathcal{C}_1^1, \ldots, \mathcal{C}_n^1\} \in I_1$, $\{\mathcal{C}_1^2, \ldots, \mathcal{C}_n^2\} \in I_2 \rightarrow \mathcal{M}_{cluster}$;
3. **for** $(\mathcal{C}_i^1, \mathcal{C}_i^2)$ *in* $\mathcal{M}_{cluster}$ **do**
    if $|\mathcal{C}_i^1| > 10,000$ *and* $|\mathcal{C}_i^2| > 10,000$ **then**
        a. $(\mathcal{C}_i^1, \mathcal{C}_i^2) \rightarrow \mathcal{M}_{coarse}$;
        b. fine-scale clustering with SLIC $\rightarrow \mathcal{S}_i^1 \subset \mathcal{C}_i^1 \in I_1, \mathcal{S}_i^2 \subset \mathcal{C}_i^2 \in I_2$ (Sec. 4.5.1);
        c. graph matching of superpixels in $\mathcal{S}_i^1, \mathcal{S}_i^2 \rightarrow \mathcal{M}_{fine}$ (Sec. 4.5.2);
        d. sparse flow $\mathcal{O}_c$ from pixel matching within each matched pair $(\mathcal{S}_i^1, \mathcal{S}_i^2) \in \mathcal{M}_{fine}$;
    else
        $(\mathcal{C}_i^1, \mathcal{C}_i^2) \rightarrow \mathcal{M}_{small}$;
        a. sparse flow $\mathcal{O}_s$ from pixel matching within matched pair $(\mathcal{C}_i^1, \mathcal{C}_i^2) \in \mathcal{M}_{small}$;
end
4. initial sparse flow $\mathcal{O} = \mathcal{O} \cup \{\mathcal{O}_c, \mathcal{O}_s\}$;
5. interpolation of sparse flow $\mathcal{O}$ and variational refinement (Sec. 4.5.3);

4.5.1 Perceptual Grouping and Feature Matching

Feature descriptors encode discriminative information about a pixel and form the basis of the perceptual grouping and matching. We conduct experiments with three different feature descriptors: rootSIFT proposed in Arandjelovic and Zisserman (2012), pretrained DeepLab on ImageNet, and pretrained encoders with the same architecture as in Teed and Deng (2021). As discussed later in the experimental results and Section 4.6.2, the latter descriptor results in the best performance. Next, we cluster pixels based on their feature descriptors to replace the rigid structure of the pixel grid as shown in Figure 4.1b. Specifically, we classify each pixel as the argmax value of its N-dimensional

feature descriptor and aggregate them into clusters. Thus, a pixel p is assigned a cluster index i_p given by,

$$i_p = \arg\max(Softmax(ReLU(F_c(p)))) \tag{10}$$

where $\mathcal{F}_c$ is the feature descriptor. Hence, this results in an arbitrary number of coarse-scale clusters in each image matched according to their cluster indices. A cluster may be non-contiguous. Since the index is calculated from the feature descriptor as in Equation 10, it specifies the class of the object and is used during graph matching to match clusters of the same class, as explained in the following section.

Pixels contained in clusters with an area less than $10,000$ are matched according to the similarity of their feature descriptors using the sum of squared differences (SSD) with a ratio-test. Outliers in the initial matches are removed from subsequent processing using RANSAC, which finds a localized fundamental matrix per cluster.

The initial sparse flow resulting from this step consists of the flow calculated from each of the inlier features. Figure 4.1f shows the initial flow resulting from the sparse feature matching of the pixels contained within all small clusters. The size of pixels is magnified by 10×10 for clarity in the visualization.

Coarse-scale clusters with a larger area than $10,000$ pixels are further clustered by a simple linear iterative clustering (SLIC) which adapts k-means clustering to group pixels into perceptually meaningful atomic regions Achanta et al. (2012). The parameter κ is calculated based on the image size and the desired superpixel size and is given by $\kappa = \frac{|I|}{|s|}$ where $|s| \approx 2223, s \in \mathcal{S}$, and $|I|$ is the size of the image. This restricts the number of the approximately equally-sized superpixels $\mathcal{S}$; in our experiments discussed in Section 4.6.2, the optimal value for $\kappa \approx 250$ to 300. For the finer-scale superpixels $\mathcal{S}$, a graph is constructed where each node corresponds to a superpixel's centroid, and edges correspond to the result Delaunay triangulation as explained in the following Section 4.5.2.

4.5.2 Graph Matching

The two sets of superpixels contained in the matched coarse-scale clusters of images I_1, I_2 are represented with the graph model described in Section 4.4. For each superpixel S, the nodes P are

a subset of all the pixels p in S i.e. $P \subseteq \{p : \forall p \in S \in I\}$. The edges E and topology T of each graph are derived from a Delaunay triangulation of the nodes P. The graph is undirected, and the edge-weight function $w(.,.)$ is symmetrical w.r.t. edges $\vec{e_a}, \vec{e_b} \in E$, such that $w(\vec{e_a}, \vec{e_b}) = w(\vec{e_b}, \vec{e_a})$. The similarity functions $\lambda^P(.,.)$ and $\lambda^E(.,.)$ are also symmetrical; for $p_i, p_j \in P_1$, $p_k, p_l \in P_2$, and edges $e_a \in E_1$, $e_b \in E_2$, the similarity functions are given by,

$$\lambda^P(p_i, p_k) = e^{-|d^P(f(p_i), f(p_k))|} \tag{11}$$

$$\lambda^E(e_a, e_b) = e^{-\frac{1}{2}\left[\Phi^\circ + |d^E(\theta_{e_a}, \theta_{e_b})| + |d^L(e_a, e_b)|\right]} \tag{12}$$

where Φ° is given by,

$$\begin{aligned}\Phi^\circ = \Phi^1_{gradient}(f(p_i), f(p_j), f(p_k), f(p_l)) + \Phi^2_{gradient}(f(p_i), f(p_j), f(p_k), f(p_l)) + \\ \Phi^1_{color}(C(p_i), C(p_j), C(p_k), C(p_l)) + \Phi^2_{color}(C(p_i), C(p_j), C(p_k), C(p_l))\end{aligned} \tag{13}$$

$$\begin{aligned}\Phi^1_{gradient} &= |d^P(f(p_i), f(p_k))| + |d^P(f(p_j), f(p_l))| \\ \Phi^1_{color} &= |d^{\mathcal{C}}(f(p_i), f(p_k))| + |d^{\mathcal{C}}(f(p_j), f(p_l))|\end{aligned} \tag{14}$$

$$\begin{aligned}\Phi^2_{gradient} &= |d^P(f(p_i), f(p_j))| - |d^P(f(p_k), f(p_l))| \\ \Phi^2_{color} &= |d^{\mathcal{C}}(f(p_i), f(p_j))| - |d^{\mathcal{C}}(f(p_k), f(p_l))|\end{aligned} \tag{15}$$

$f : P \longrightarrow S$ is a feature descriptor with cardinality S for a node $p \in P$, $\mathcal{C} : P \longrightarrow 6$ is a function which calculates the 6-vector $< \mu_r, \mu_g, \mu_b, \sigma_r, \sigma_g, \sigma_b >$ containing color distribution means and variances (μ, σ) at p modeled as a 1D Gaussian for each color channel, $d^P : S \times S \longrightarrow \mathbb{R}$ is the $\mathcal{L}^1$-norm of the difference between the feature descriptors of two nodes in $p_i, p_j, p_k, p_l \in P$, $d^E : \mathbb{R} \times \mathbb{R} \longrightarrow \mathbb{R}$ is the difference between the angles $\theta_{e_a}, \theta_{e_b}$ of the two edges $e_a \in E_1, e_b \in E_2$ to the horizontal axes, and $d^{\mathcal{C}} : 6 \times 6 \longrightarrow \mathbb{R}$ is the $\mathcal{L}^1$-norm of the difference between the two 6-vectors containing color distribution information for the two nodes in $p_i, p_j, p_k, p_l \in P$.

Φ^1_* signify first-order similarities and measures similarities between the nodes and edges of the two graphs. In addition to the first-order similarities Φ^1_*, the functions in the above equations define additional second-order similarities Φ^2_* which have been shown to improve the performance

of the matching Cho, Lee, and Lee (2010). That is, instead of using only similarity functions that result in small differences between similar gradients/colours and large otherwise, e.g. first-order, we additionally incorporate the second-order similarities defined above, which measure the similarity between the two gradients and colours using the *distance between their differences* Tian et al. (2019). For example, the first-order similarity $\Phi^1_{gradient}$ calculates the distance between the two feature descriptors in the two graphs i.e. $\lambda^P(p_i, p_k)$ in Equation 11, whereas the second-order similarity calculates the *distance between the feature descriptor differences of the end-points in each graph* i.e. $\Phi^2_{gradient}$ and Φ^2_{color} in Equations 11 and 15. A descriptor $f(s_i)$, as defined in Equation 13, is calculated for each centroid-node representing superpixel $s_i \in \mathcal{S}$ as the average of the feature descriptors of all pixels contained within it $f(s_i) = \frac{1}{|s_i|} \sum_{\forall p \in s_i \subset I} \phi_p$ where $|s_i|$ is the number of pixels in superpixel s_i, and ϕ_p is the feature descriptor of pixel $p \in s_i \subset I$.

Given the above function definitions, graph matching is solved by maximizing Equation 8 using a path-following algorithm. $\mathbf{K}$ is factorized into a Kronecker product of six smaller matrices which ensures tractable computational complexity on graphs with nodes $N, M \approx 300$ F. Zhou and De la Torre (2012). Furthermore, robustness to geometric transformations such as rotation and scale is increased by finding an optimal transformation at the same time as finding the optimal correspondences and thus enforcing global rigid (e.g. similarity, affine) and non-rigid geometric constraints during the optimization F. Zhou and De la Torre (2013).

The result is superpixels matches within the matched coarse-scale clusters. Assuming a piecewise rigid motion, we use RANSAC to remove outliers from the superpixel matches. For each superpixel s having at least three matched neighbours, we fit an affine transformation. We only check whether the superpixel s is an outlier, in which case it is removed from further processing. This process is repeated for all small clusters and graph-matched superpixels. We proceed by matching the pixels contained within the matched superpixels based on their feature descriptors. Similar to earlier in Section 4.5.1, we remove outlier pixel matches contained in the superpixels using RANSAC to find a localized fundamental matrix.

The initial sparse flow resulting from graph matching consists of flow calculated from every pixel contained in the matched superpixels. Figure 4.1b shows the result of the clustering of the feature descriptors for the image shown in Figure 4.1a. Clusters having a large area are further

divided into superpixels. The graph nodes correspond to each superpixel's centroid, and the edges result from the Delaunay triangulation of the nodes, as explained above. Figure 4.1c and Figure 4.1d show the result of graph matching superpixels within matched coarse-scale clusters. The matches are colour-coded, and unmatched nodes are depicted as smaller yellow circles. Examples of unmatched nodes appear in the left part of the left image in Figure 4.1c. The images shown are from the benchmark dataset MPI-Sintel Butler et al. (2012).

4.5.3 Interpolation and Refinement

The combined initial sparse flows (Figure 4.1f, 4.1e) calculated from sparse feature matching and graph matching, as described above in Sections 4.5.1 and 4.5.2 respectively, are first interpolated and then refined. For the interpolation, we apply an edge-preserving technique Revaud et al. (2015). This results in dense flow as shown in Figure 4.1g. In the final step, we refine the interpolated flow using variational optimization on the full-scale of the initial flows, i.e. no coarse-to-fine scheme, with the same data and smoothness terms as used in Revaud et al. (2015). The final result is shown in Figure 4.1h.

4.6 Experimental Results

In this section, we report on the evaluation of HybridFlow on benchmark datasets and compare it with state-of-the-art variational optical flow techniques. In Section 4.7, we present two applications of the proposed technique on large-scale image-based reconstruction where ground truth is unavailable. Specifically, we use large-scale aerial imagery, and Full-Motion Video (FMV) captured from aerial sensors and demonstrate how our technique easily scales to ultra-high resolution images, in contrast to deep learning alternatives.

4.6.1 Datasets and evaluation metrics

We evaluate HybridFlow on the two widely used benchmark datasets for motion estimation:

- MPI-Sintel Butler et al. (2012) — a synthetic data set for the evaluation of optical flow derived from the open source 3D animated short film, Sintel. It includes image sequences with large

displacements, motion blur, and non-rigid motion.

- KITTI-2015 Menze et al. (2015) — a real data set captured with an autonomous driving platform. It contains dynamic scenes of real world conditions and features large displacements and complex 3D objects.

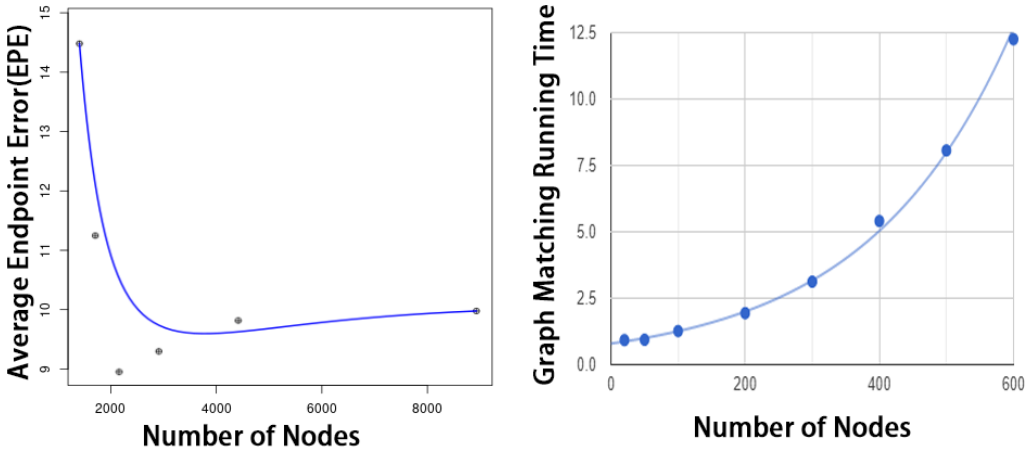
The quantitative evaluation is performed in terms of the average endpoint error(EPE) for MPI-Sintel, and percentage of optical flow outliers(FI) for KITTI-2015.

4.6.2 Implementation details

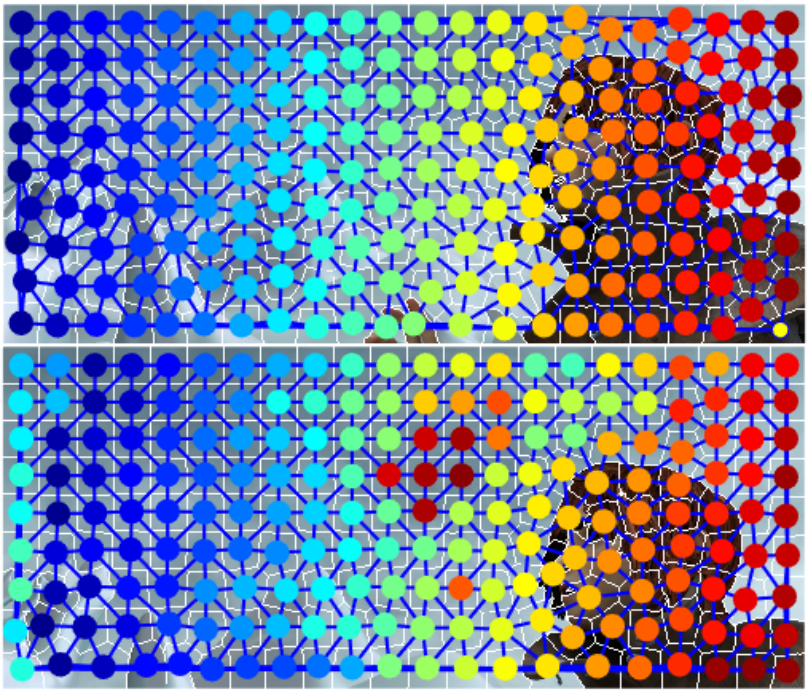
The proposed approach was implemented by Q. Chen in Python. All experiments were run on a workstation with an Intel i7 processor. We extract the features descriptors using the approach introduced in RAFT Teed and Deng (2021). Perceptual grouping using SLIC superpixels is performed using the method in Achanta et al. (2012). We factorize graphs into Kronecker products as presented in F. Zhou and De la Torre (2012) and perform deformable graph matching following the approach in F. Zhou and De la Torre (2013). Finally, we interpolate the combined initial flows from sparse feature matching and graph matching using the edge-preserving interpolation and variational refinement in EpicFlowRevaud et al. (2015).

Superpixel size. We empirically determined the optimal size of the superpixels which subsequently determined the number of superpixels κ as defined in Section 4.5.1. Figure 4.4 shows an example from the experiments on different superpixel sizes. The rows correspond to the superpixel sizes $|s|= 22323$ (20 superpixels), $|s|= 2232$ (200 superpixels), $|s|= 1116$ (400 superpixels) and $|s|= 223$(2000 superpixels) respectively. The first and second columns show the colour-coded matches using only the graph matching technique described in Section 4.5.2. Figure 4.3a shows a graph of the average endpoint error (EPE) of the final optical flow as a function of the superpixel size performed on the training image sequences of the MPI-Sintel dataset. In Figure 4.3b we show the increase of the graph matching's computational time as a function of the number of nodes in the graphs.

Initial coarse-scale clustering. The initial coarse-scale clusters are formed by clustering the pixels' feature descriptors. This is a crucial part of the process, which increases robustness to large



(a) (b)



(c)

Figure 4.3: (a) Average end-point error (EPE) w.r.t number of graph nodes per image(1024×436). (b) Average graph-matching time complexity (seconds) w.r.t. number of graph nodes. We empirically determine the optimal number of superpixels by performing graph matching using different superpixel sizes and calculate the EPE of the resulting optical flow. Optimal size is found to be $|s| \approx 300$. (c) Ablation: Graph matching using SLIC clusters as the initial coarse-scale-clusters instead of clustering the feature descriptors. Superpixel clustering results in a near-rigid pixel grid that, as can be seen, is not robust to occlusions. The number of superpixels is set to 200. The first and second columns show the colour-coded matches of the graph nodes using graph matching based on an initial coarse-scale clustering of superpixels (SLIC).

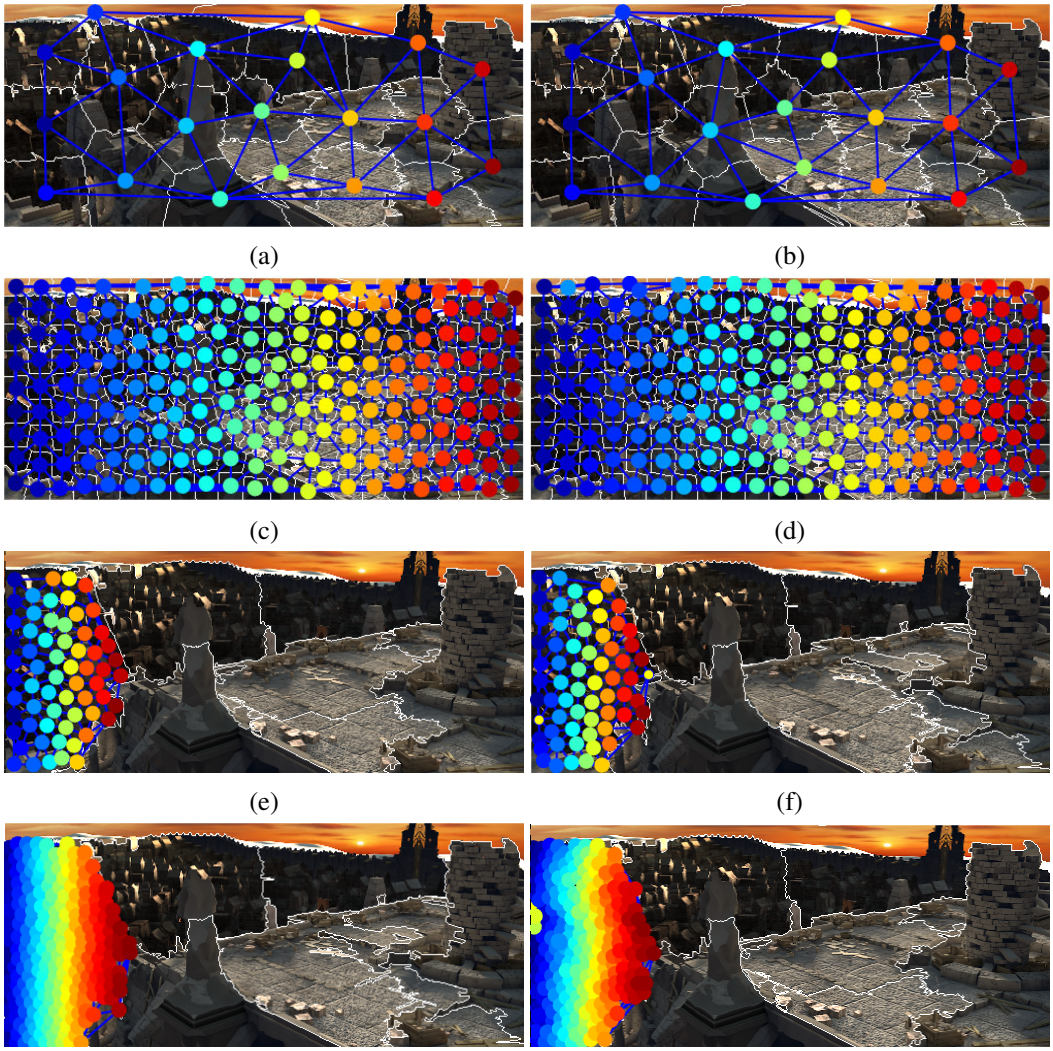


Figure 4.4: Superpixel size. Graph-matching using different superpixel sizes. The images correspond to the examples of superpixel sizes $|s|=$ 22323 (20 superpixels, Figures (a),(b)), $|s|=$ 2232 (200 superpixels, Figures (c),(d)), $|s|=$ 1116 (5 clusters subdivided into 80 superpixels, Figures (e),(f)) and $|s|=$ 223 (5 clusters subdivided into 400 superpixels, Figures (g),(h))respectively. The figures show the colour-coded graph node matches using *only* graph matching as explained in Section 4.5.2.

displacements. As shown in Figure 4.3c, using SLIC superpixels on the entire image results in a near-rigid rectangular pixel grid and consequently failures in graph matching. This is evident from the mismatching of the dark red circles in the middle of the right image. Our experiments show that an irregular pixel grid based on features descriptors increases the robustness in the presence of large displacements and deformations.

		Sintel – final pass									Sintel-Clean		KITTI-20015		
	Method	EPE all	EPE-noc	EPE-occ	d0-10	d10-60	d60-140	s0-10	s10-40	s40+	EPE all	EPE-noc	Fl-all Noc/est	Fl-fg Noc/Est	Fl-bg Noc/Est
DL	GMA Jiang et al. (2021)	**2.470**	**1.241**	**12.501**	**2.863**	**1.057**	**0.653**	**0.566**	**1.817**	**13.492**	**1.388**	**0.582**	**2.94**	**3.69**	3.07
	RAFT Teed and Deng (2021)	2.855	1.405	14.680	3.112	1.133	0.770	0.634	1.823	16.371	1.609	0.623	3.07	3.98	**2.87**
	ScopeFlow Bar-Haim and Wolf (2020)	4.098	1.999	21.214	4.028	1.689	1.180	0.725	2.589	24.477	3.592	1.400	4.45	4.49	4.44
VM	SfM-PM Maurer, Marniok, Goldluecke, and Bruhn (2018)	5.466	2.683	**28.147**	4.963	2.186	1.782	1.031	3.182	32.991	**2.910**	1.016	**9.30**	19.94	**6.94**
	RicFlow Hu et al. (2017)	5.620	2.765	28.907	5.146	2.366	1.679	1.088	3.364	33.573	3.550	1.264	10.29	14.88	9.27
	CPM Hu et al. (2016)	5.960	2.990	30.177	5.038	2.419	2.143	1.155	3.755	35.136	3.557	1.189	13.85	18.71	12.77
	CPM2 Y. Li, Hu, Song, Rao, and Wang (2017)	6.180	3.012	32.008	5.059	2.399	2.126	1.212	3.625	37.014	3.253	0.980	–	–	–
	EpicFlow Revaud et al. (2015)	6.285	3.060	32.564	5.205	2.611	2.216	1.135	3.727	38.021	4.115	1.360	16.69	24.34	15.00
	HybridFlow (SIFT)	8.082	4.966	33.445	7.513	4.907	3.983	2.635	5.401	41.585	7.018	4.086	23.57	19.19	24.32
	HybridFlow (DeepLab)	7.677	4.507	33.471	7.281	4.592	3.279	2.214	5.043	41.139	5.788	2.815	18.55	14.36	19.28
	HybridFlow	**5.121**	**1.999**	30.531	**4.087**	**1.598**	**1.213**	**0.871**	**2.483**	**32.559**	3.791	**0.962**	16.96	**14.18**	16.54

Table 4.1: Benchmark datasets results. The top half of the table (*DL*) are the top-performing deep learning methods; the bottom half of the table (*VM*) are the top-performing variational methods. For MPI-Sintel results, EPE-noc is the EPE on the non-occluded areas, and EPE-occ is the EPE on occluded areas. s0-10 is the EPE for pixels whose motion speed is between 0-10 pixels, similarly for s10-40 and s40+; d0-10 is the endpoint error over regions between 0 and 10 pixels apart from the nearest occlusion boundary, similarly for d10-60 and d60-140. For the KITTI-2015 test-set non-occluded pixels, FI-bg is the percentage of optical flow outliers for background, FI-fg is the percentage of optical flow outliers for the foreground, FI-all/Est is the percentage of outliers averaged over all non-occluded ground truth pixels.

4.6.3 Comparison of clustering techniques

We compared initial coarse-scale clusters formed by (a) Delaunay triangulation of rootSIFT features, (b) SLIC superpixels, (c) Felsenszwalb's Felzenszwalb and Huttenlocher (2004) graph-based image segmentation technique, and (d) our proposed clustering of feature descriptors. As shown in Figure 4.5, initial coarse-scale clustering using SLIC, Felsenszwalb's graph-based technique and Delaunay triangulation of rootSIFT features cause erroneous results in graph matching, which accumulate in the finer-scales. However, coarse-scale clusters based on clustering feature descriptors provide consistent and robust performance. The average endpoint error (EPE) for the Sintel images in Figure 4.5 are 2.33, 2.12, 1.95 and 1.08 respectively. The last column shows the ground truth and below the resulting optical flow using each technique.

4.6.4 Quantitative Evaluations

On Synthetic Data (MPI-Sintel) Table 4.1 shows the average endpoint error (EPE) on the MPI-Sintel 'clean' and 'final' (realistic rendering effect) image dataset for HybridFlow and other state-of-the-art variational optical flow techniques. We present our results using three types of pixel-wise descriptors: (i) rootSIFT descriptors, named as HybridFlow (SIFT), (ii) features descriptors

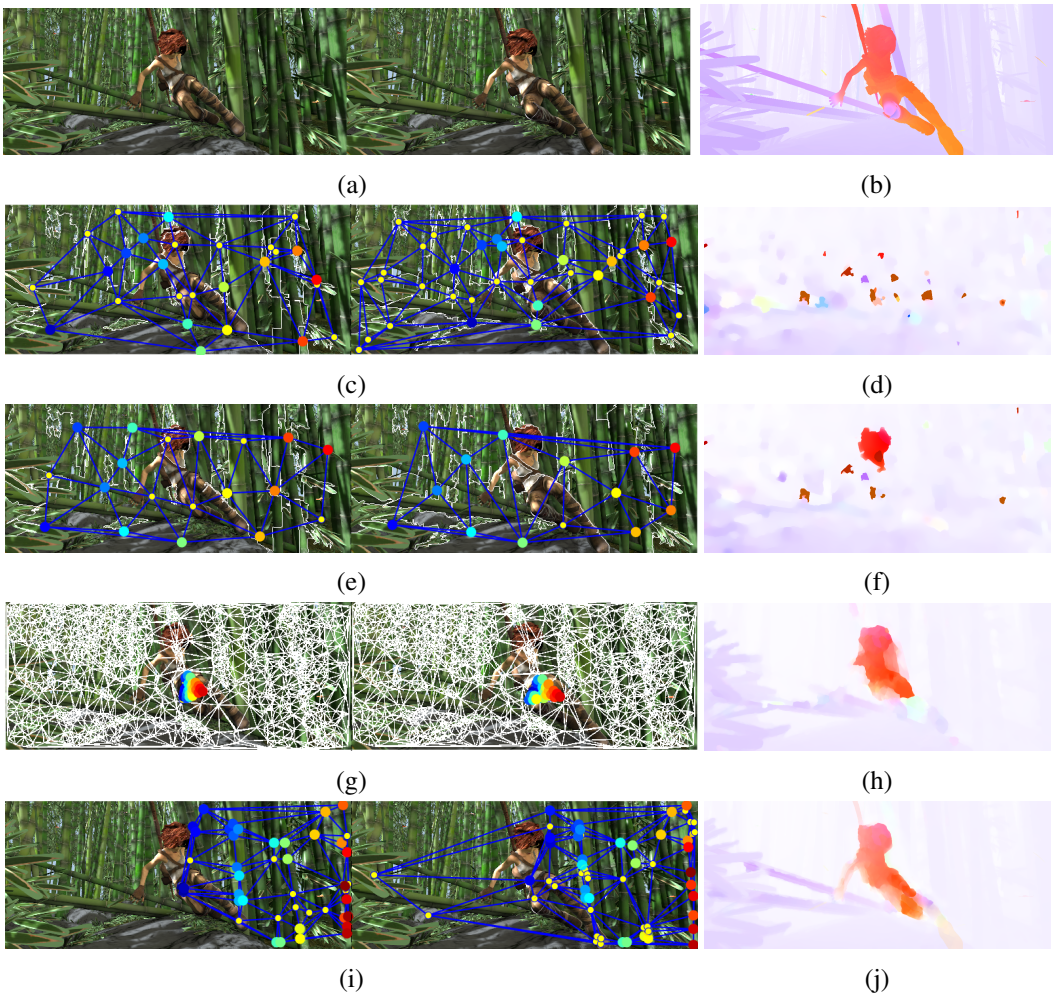


Figure 4.5: Graph Matching with different initial coarse-scale clustering methods on the pair of images shown in Figure (a). The initial coarse-scale clusters are resulting from Felsenszwalb's Felzenszwalb and Huttenlocher (2004) graph-based segmentation (Figure (c), SLIC superpixels Achanta et al. (2012) (Figure (e)), Delaunay triangulation of rootSIFT features (Figure(g)) and clustering of feature descriptors(Figure(i)). Figure (d),(f),(h),(j) shows the optical flow results corresponding to each technique; ground truth shown in Figure (b).

extracted from a pre-trained ResNet He, Zhang, Ren, and Sun (2015) trained on Sintel, named as HybridFlow (DeepLab), and (iii) descriptors learned by feature and context encoder as in RAFT Teed and Deng (2021), name as HybridFlow. HybridFlow outperforms all other state-of-the-art variational techniques and gives comparable results to the deep-learning-based techniques with an average overall EPE of 5.121 in MPI-Sintel 'final' datasets.

On Real Data (KITTI-2015). Table 4.1 shows the results for HybridFlow and other non-stereo-based optical flow methods on the 200 KITTI-2015 test images. Although HybridFlow does

not have the best overall performance, it outperforms all variational techniques on the non-occluded test-set and has comparable performance for the other categories. Specifically, the percentage of background, foreground, and overall outliers are 31.06%, 17.25%, and 29.27%, respectively. The percentages of outliers for non-occluded areas are 16.96%, 14.18%, and 16.54%.

Failure cases

Graph matching is robust to texture variations, illumination variations, and deformations. However, erroneous matches can be introduced when large occluded areas fall inside the convex graph, as shown in the example in Figure 4.3c. Mismatches in the graph matching can lead to the wrong matching of the finer-scale superpixels, and consequently, significant errors in the optical flow. This is clearly evident from the results in Table 4.1 for Sintel and KITTI-2015, where for the non-occluded test-sets, HybridFlow outperforms all state-of-the-art variational methods and matches the performance of deep-learning techniques such as ScopeFlow.

4.7 Application: Large-scale 3D reconstruction

The motivation for our work is large-scale 3D reconstructions from airborne images. In particular, we focus on full-motion video (FMV) and large-scale aerial imagery, typically captured by a UAV/helicopter and an airplane, respectively. Deep learning techniques are not applicable since they have a fixed input size. Thus, a very high-resolution image must be scaled-down to typically less than $1K \times 1K$ to be used as input to the network. This significant reduction in resolution leads to low-resolution optical flow and significantly low-fidelity 3D models. Most notably, there is no ground truth dataset for real scenarios to train the deep learning models.

On the other hand, the state-of-the-art variational methods considered in this work also impose restrictions on the input image size. For example, RicFlow and EpicFlow use a hierarchical structure employed by DeepMatching, which on an 8GB GPU can only handle $1K \times 1K$ resolutions. HybridFlow can handle arbitrary-sized resolutions with a low memory footprint. In this section, we present the results of the application of HybridFlow on the use case of large-scale 3D reconstruction from airborne images. We reiterate that there is no ground truth data for training models in such

scenarios, and the resolutions can be significantly higher than $1K \times 1K$.

4.7.1 Image-based Large-scale Reconstruction

Image-based reconstruction involves three main components: (1) Structure-from-Motion (SfM) for camera pose estimation, (2) Bundle Adjustment optimization, and (3) Multi-View Stereo (MVS). In contrast, we reformulate the reconstruction as a single-step process. Using HybridFlow allows us to triangulate directly the dense matches without MVS as a post-processing step, therefore achieving faster reconstructions.

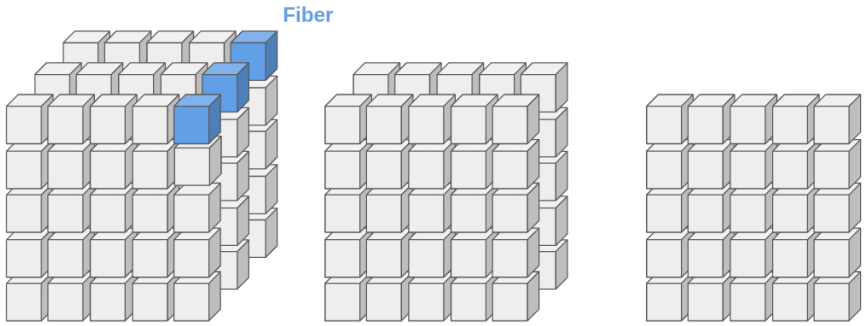


Figure 4.6: On-disk dynamic tensor-shaped data structure. For each image, we store a tensor with layers containing pixel-level matches to subsequent images based on the HybridFlow. Unmatched pixels in the second image are stored in the tensor data structure for the second image, which contains layers with pixel-level matches to the third image and onward. A fiber is shown in blue. Each cell contains the match of that pixel, i.e. the top right corner in all subsequent images. Reconstruction is reduced to triangulating the matches contained within each fiber.

We design a specialized off-memory, on-disk data structure for storing the matches. As shown in Figure 5.12, at every image, we keep a tensor with layers containing pixel-level matches to subsequent images based on the HybridFlow. Unmatched pixels in the second image are stored in the tensor data structure for the second image, which contains layers with pixel-level matches to the third image and onwards. The data structure can scale up dynamically to arbitrary-sized datasets (subject to the disk limits) and allows for efficient outlier removal and validation, i.e. multiple pixels in the same image cannot be matched to the same pixel in the following image. A simple look-up at a fiber of the tensor gives the matches for that pixel in all subsequent images. Hence, reconstruction is reduced to traversing all fibers in each tensor and triangulating to get a 3D position.

We demonstrate the effectiveness of HybridFlow on large-scale reconstruction from images and

Method	Features	# matches	SfM Variant	time-SfM (min)	# points	MVS Variant	# points	time-MVS (min)	Run-time	Reprojection Error
VisualSFM Schonberger and Frahm (2016)	SIFT G. Lowe (2004)	35,0161	Bundler Snavely, Seitz, and Szeliski (2006)	1.10	4,863	PMVS Furukawa and Ponce (2007)	111,189	1.289	2.389	0.880
COLMAP Schonberger and Frahm (2016)	RootSIFT Arandjelovic and Zisserman (2012)	704,127	iterative BA	11.051	11,274	Kazhdan Schönberger et al. (2016)	287,205	23.965	35.016	0.810
Our method	Epic-Flow Revaud et al. (2015)	1,576,705	iterative BA	12.533	139,606	-	139,606	-	12.533	1.004
Our method	HybridFlow	8,144,093	iterative BA	16.052	6,512,324	-	6,512,324	-	16.052	0.820

Table 4.2: The comparison of number of points reconstructed and reprojection error.

present result on two different types of datasets: full-motion video, and large-scale aerial imagery. We followed the single step process described above employing the dynamic tensor-shaped data structure for the efficient processing of the matches calculated by HybridFlow.

4.7.2 Full-motion Video

Full-motion video (FMV) is typically captured by a helicopter at an oblique aerial angle so that the rooftops and the facades of the buildings are visible in the images. The ground sampling density is significantly higher than that of a satellite image, i.e. in the order of a few cms, and can vary according to the aircraft's flight height, depending on the area it is flying over.

We ran experiments on a full-motion video dataset containing images taken from a helicopter circling an area containing a few mockup buildings. Our test dataset contains 71 images with resolution 1280×720 with unknown camera calibrations or EXIF information. We report results using the (i) single-step reconstruction using HybridFlow matches, the (ii) same single-step reconstruction using EpicFlow matches, (iii) and the state-of-the-art incremental SfM techniques Bundler Snavely et al. (2006), VisualSFM C. Wu et al. (2011), COLMAP Schonberger and Frahm (2016).

Perhaps the most popular feature extraction methods used in SfM is SIFT G. Lowe (2004). In COLMAP Schonberger and Frahm (2016), they use a modified version called RootSIFT Arandjelovic and Zisserman (2012) for extracting and matching each image. The first comparison focuses on the density of the matches. Figure 4.7c shows the SIFT matches, Figure 4.7d the RootSIFT matches, Figure 4.7e the EpicFlow matches, and Figure 4.7f the HybridFlow matches for the input images shown in Figures 4.7a and 4.7b. The latter two show the matches as colour-coded optical flows for visualization clarity, otherwise drawing the matches will cover the entire image. Table 4.2 presents the total number of matches per technique. As expected, SIFT and RootSIFT have the lowest number of matches since they only extract scale-space extrema. On the other hand, the dense

optical flow technique EpicFlow results in eight times lower number of matches than HybridFlow.

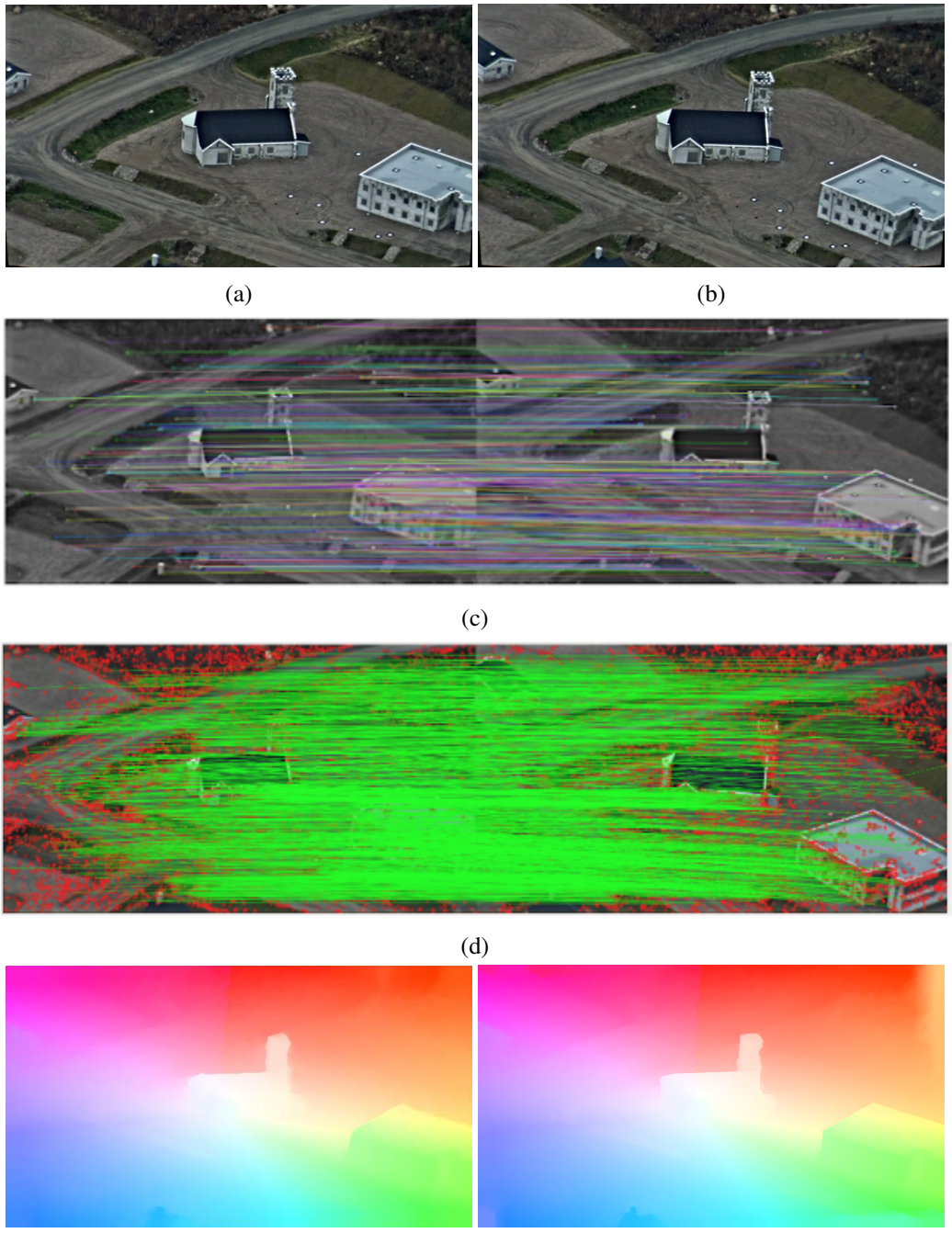
The reconstruction can serve as a proxy for the accuracy of the matches in cases where ground truth is not available. We proceed with the evaluation of the reconstruction in terms of the reprojection error. Figure 4.8 shows the reconstructed pointcloud of (a) COLMAP's sparse (SfM) reconstruction, (b) COLMAP's dense (MVS) reconstruction, (c) our single-step reconstruction using HybridFlow matches, and (d) our single-step reconstruction using EpicFlow matches. The reconstructed point clouds are rendered from the same viewpoint and camera intrinsics. The reprojection error using our single-step method with HybridFlow achieves the highest number of reconstructed points in the lowest time per point, while the reprojection error is comparable with COLMAP for almost 60x more points.

4.7.3 Large-scale Aerial Imagery

Large-scale Aerial Imagery is captured by an aircraft flying at over 10,000ft and can cover areas of $10 - 20km^2$. The aircraft orbits around the area of interest during the flight, and an array of cameras captures and streams image data at about two frames per second.

Figure 4.9a shows an example of large-scale aerial imagery capturing a downtown urban area. The resolution is 6600×4400 is considered average amongst large-scale aerial imagery, since some of the larger resolutions can reach sizes of up to 14000×12000. Deep learning techniques can be applied only (i) by rescaling the image to the fixed input size expected by the neural network, or (ii) tiling the image, calculating flows per tile, and then merging the results. In the first case, rescaling reduces the resolution and subsequently the final number of reconstructed points. Furthermore, essential details such as cars and trees are completely removed. In the latter case, there is no one-to-one mapping between tiles. For example, a tile may contain areas appearing in two or more different tiles in the second image. Furthermore, the deep optical flow techniques always return a match for every pixel. That means that even if an area is not present in a tile, this will nevertheless be matched to another area in the second image. For these reasons, deep learning techniques cannot be applied in these use cases.

Competing variational methods such as RicFlow Hu et al. (2017), EpicFlow Revaud et al. (2015) cannot be applied either since hierarchical structure employed by DeepMatching Revaud et al.



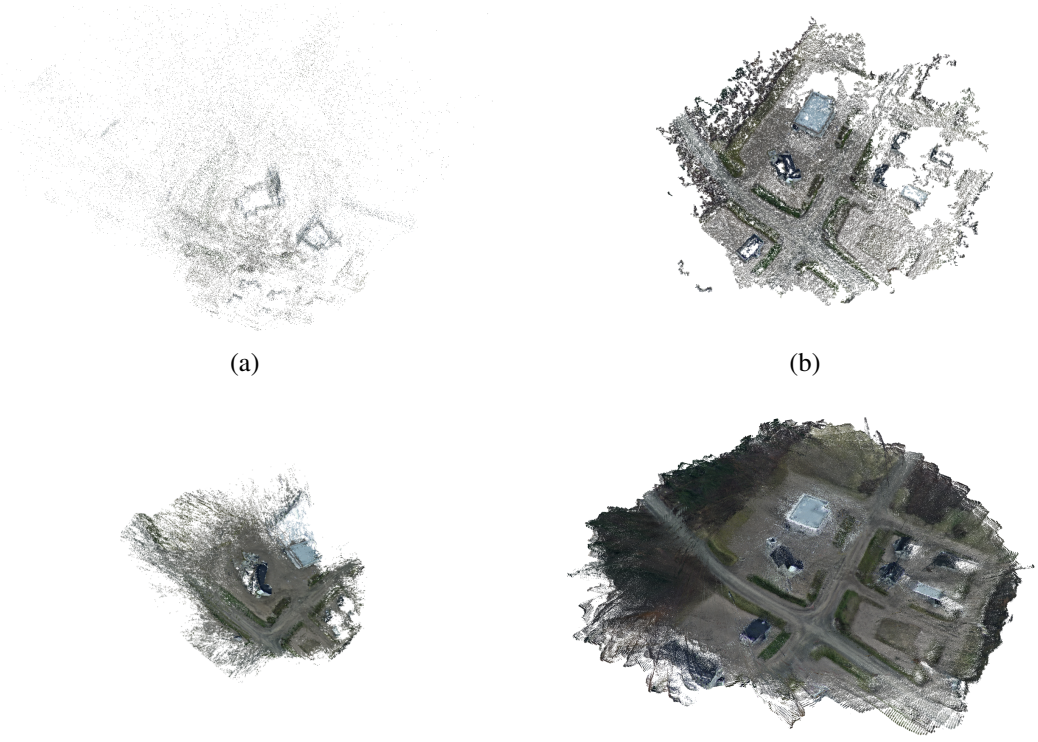
(a) (b)

(c)

(d)

(e) (f)

Figure 4.7: Density of matches. The first row (a) and (b) shows an example of the input image frames, copyright of ©His Majesty the King in Right of Canada, as represented by the Minister of National Defence, 2022. (c) shows SIFT G. Lowe (2004) matches, (d) shows RootSIFT Arandjelovic and Zisserman (2012) matches, (e) and (f) shows EpicFlow Revaud et al. (2015) and HybridFlow results.

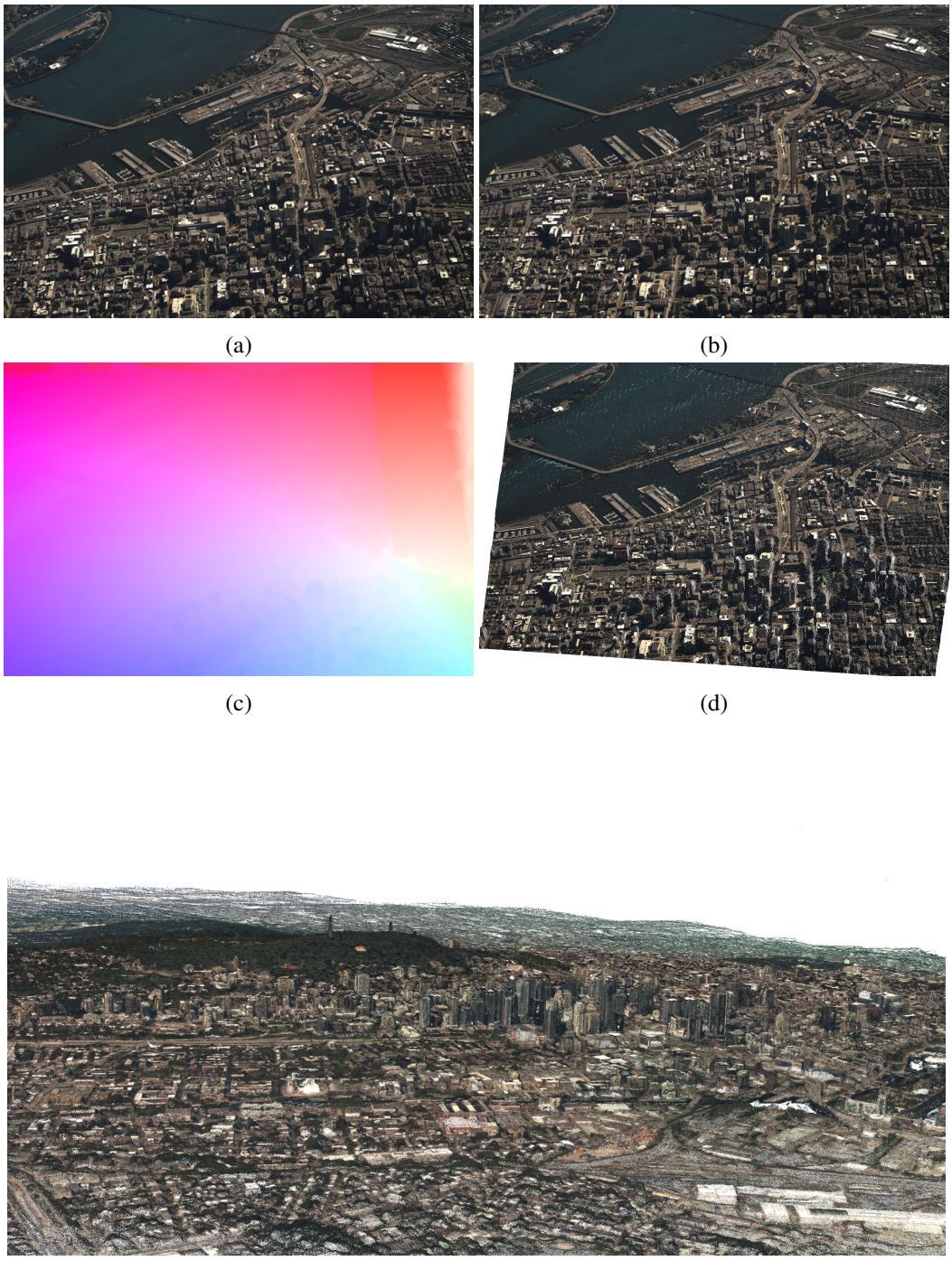


(a) (b)

(c) (d)

Figure 4.8: The reconstruction serves as a proxy to the accuracy of the matches. We calculate and compare reprojection errors for the techniques shown in Table 4.2. (a) shows COLMAP's sparse (SfM) reconstruction, (b) shows COLMAP's dense (MVS) reconstruction Schonberger and Frahm (2016), (c) shows our single step reconstruction using dense matches from Epicflow Revaud et al. (2015), and (d) shows our single step reconstruction with Hybridflow. HybridFlow produces 60x more matches than COLMAP and 47x more matches than EpicFlow. The reprojection error is comparable with COLMAP (for 60x more points) while the runtime is less than half.

(2016), which on an 8GB GPU can only handle $1K \times 1K$ resolutions. In contrast, HybridFlow is the only top-performing variational method that can handle arbitrary-sized images such as large-scale aerial imagery. Figure 4.9a and 4.9b shows two consecutive images capturing a downtown urban area having a resolution of 6600×4400. HybridFlow is the only top-performing variational method that can handle high-resolution images as shown in Figure 4.9c. Deep learning techniques cannot be applied due to the fixed input size of the networks. Similarly, competing state-of-the-art variational methods cannot be applied for this size of images as explained above. Figure 4.9d shows the resampled image from Figure 4.9b using the HybridFlow matches in Figure 4.9c and the matched pixels in Figure 4.9a. Figure 4.9e shows a render of the reconstructed pointcloud for the



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

Figure 4.9: (a) and (b) are two consecutive large-scale aerial images of a downtown urban area with resolution 6600×4400. Copyright of ©His Majesty the King in Right of Canada, as represented by the Minister of National Defence, 2022. (c) HybridFlow is the only top-performing variational method that can handle high-resolution images. Deep learning techniques cannot be applied due to the fixed input size of the networks as explained in the text. (d) Image resampled from (a) using HybridFlow flows in (c) to form (b). (e) Reconstructed pointcloud using 320 images.

downtown urban area generated using 320 images of the same size.

4.8 Conclusion

We addressed the problem of large displacement optical flow and presented a hybrid approach based on sparse feature matching using feature descriptors and graph matching, named HybridFlow. In contrast to state-of-the-art, it does not require training, and the use of sparse feature matching is robust and can scale up to arbitrary image sizes. This makes our technique applicable in use-cases such as reconstruction or object tracking where ground-truth is unavailable, and processing must be performed in interactive time. We match initial coarse-scale clusters based on a clustering of context features. We employ graph matching to match perceptual groups clustered using SLIC superpixels within each initial coarse-scale cluster, and perform pixel matching on smaller clusters. Based on the combined feature matches and the graph-node matches, we calculate the initial flow which is interpolated using an edge-preserving interpolation and refined using variational refinement. The proposed technique has been evaluated on two benchmark datasets (Sintel, KITTI), and we compared it with the current state-of-the-art variational optical flow techniques. We show that HybridFlow surpasses all other state-of-the-art variational methods in non-occluded test sets. Specifically, for Sintel, HybridFlow has the lowest overall EPE, while for KITTI, it gives comparable results.

Data availability statement

The datasets generated and analysed during the current study are available online: Sintel Butler et al. (2012) http://sintel.is.tue.mpg.de/, and KITTI Menze and Geiger (2015) http://www.cvlibs.net/datasets/kitti/ benchmark datasets.

Chapter 5

Single-shot Dense Reconstruction

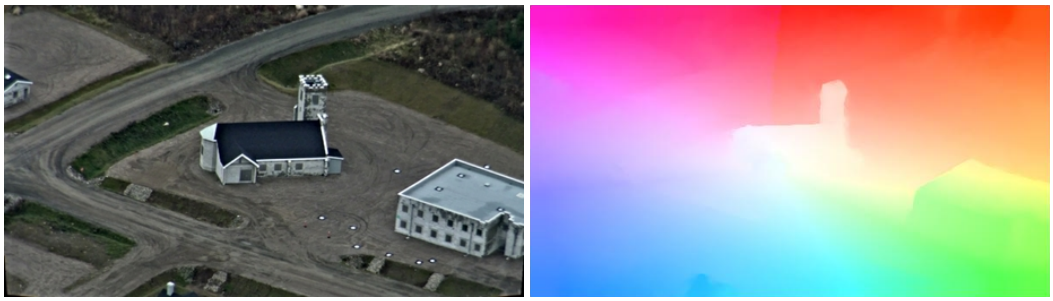
Traditional image-based modeling techniques follow a two-step process. First, the camera poses are recovered using sparse correspondences, and in a second step dense correspondences are used to reconstruct dense geometry. This chapter is based on the paper "Single-shot Dense Reconstruction with Epic-flow" Q. Chen and Poullis (2018), published in IEEE 3DTV, 2018. The proposed single-shot process employs dense flow-fields to recover correspondences which drastically reduces the computational complexity. Furthermore, the out-of-core handling of the correspondences makes it invariant to the size of the dataset.

5.1 Abstract

In this chapter, we present a novel method for generating dense reconstructions by applying only structure-from-motion(SfM) on large-scale datasets without the need for multi-view stereo as a post-processing step. A state-of-the-art optical flow technique is used to generate dense matches. The matches are encoded such that verification of their correctness becomes possible, and are stored in a database on-disk. The use of this out-of-core approach transfers the requirement for large memory space to disk, therefore allowing for the processing of even larger-scale datasets than before. We compare our approach with the state-of-the-art and present the results which verify our claims.

Index Terms – 3D reconstruction, dense reconstruction, structure-from-motion (SfM), multi-view stereo (MVS), urban reconstruction, large-scale

5.2 Introduction



(a) (b)

Figure 5.1: (a) A frame from a video captured from a helicopter circling a church building, copyright of ©His Majesty the King in Right of Canada, as represented by the Minister of National Defence, 2022. (b) Epic-flow Revaud et al. (2015) of two consecutive frames.

The automatic reconstruction of large-scale urban areas has always been of great interest to the computer graphics and vision communities. Image-based reconstructions rely on structure from motion (SfM) to recover the camera poses using bundle adjustment Schonberger and Frahm (2016); Snavely et al. (2008); C. Wu et al. (2011), followed by multi-view stereo (MVS) Furukawa and Ponce (2007); Martinec and Pajdla (2007) to generate a dense pointcloud. In recent years, many variants of these techniques have been proposed which result in impressive reconstructions.

However, dealing with remote sensor images covering large-scale areas introduces certain challenges which very often cause failures in SfM and/or MVS techniques. Firstly, remote sensor images cover large areas which contain thousands of geospatial features e.g. buildings, roads, trees, cars, etc, which from an oblique aerial or nadir direction look identical and repetitive i.e. consider a satellite image where the roads, the roofs of the building, etc, have the same texture and similar shapes. One of the main limitations of the existing state-of-the-art feature extraction and matching techniques is that they cannot handle repetitive textures, leading to erroneous matches and subsequently erroneous or failed reconstructions. Secondly, remote sensor images typically have a large size and capture the object from all around similar to an inverted turn-table i.e. consider the single frame in Figure 5.1a part of a video captured from a helicopter circling the church building. The symmetry occurring in man-made structures such as this one often leads to erroneous results since features from opposing sides of the building can be easily mistakenly matched i.e. the cameras are

facing each other.

In this chapter, we propose a method for single-shot dense reconstruction using only SfM. Unlike existing techniques, we rely on the state-of-the-art optical flow technique EpicFlow Revaud et al. (2015) to extract robust dense matches. The matches are encoded and stored on-disk, therefore, transferring the requirement for large memory to disk which is easily met. An advantage of the encoding is the fact that verification for correctness can be easily performed and ambiguous matches for which the transitivity property fails are removed. This process is explained in Section 5.4.1. An iterative bundle adjustment is used which allows for the optimization of an arbitrary number of parameters as explained in Section 5.4.2. Finally, Section 5.5 presents the experiments and comparisons with other state-of-the-art techniques which verify our claims.

Our technical contributions are:

- A novel method of generating dense reconstructions using only SfM while producing similar or better results with other state of the art, in terms of accuracy and time.
- An encoding for the matches which allows the easy identification and elimination of ambiguous matches for which the transitivity property does not hold. This ensures the robustness of the dense matches used for SfM.

5.3 Related Work

We present related work in terms of (a) dense matching and, (b) 3D reconstruction.

5.3.1 Dense Matching

Optical flow is the apparent motion between two consecutive frames caused by the movement of the object or the camera. A number of different robust techniques have already been proposed for recovering the optical flow which can be better categorized in terms of the underlying technique they use i.e. block-matching, feature tracking, and energy-based methods. Differential methods of estimating optical flow are based on computing the partial derivatives of the image and the flow field, such as LucasKanade Lucas, Kanade, et al. (1981) or BuxtonBuxton Murray and Buxton

(1987). The majority of current optical flow methods strongly resemble the original formulation of Horn-Schunck Horn and Schunck (1981). They combine a data term that assumes constancy of some image property with a spatial term that models how the flow is expected to vary across the image. Current state-of-the-art can be better categorized as follows: coarse-to-fine estimation to deal with large motions Brox et al. (2004), texture decomposition Wedel, Pock, Braun, Franke, and Cremers (2008) or high-order filter constancy Lempitsky et al. (2009) to reduce the influence of lighting changes, warping with bicubic interpolation Roth et al. (2009), graduated non-convexity to minimize non-convex energies Sun, Roth, Lewis, and Black (2008), median filtering after each incremental estimation step to remove outliers Wedel et al. (2009). FlowNet2.0 Ilg et al. (2017) re-casts the optical flow estimation as a learning problem and make an improvement over learning optical flow in terms of quality and speed.

Perhaps the most popular state of the art optical flow technique which has already been used in many successful vision systems is Epic-Flow Revaud et al. (2015). Epic-flow is an edge-preserving interpolation of correspondences for optical flow which leverages recent advances in matching algorithms and introduces an edge-aware geodesic distance that handles motion discontinuities and occlusions. We choose to use this method to compute our dense matches because of its ability to handle deformations and repetitive textures.

5.3.2 3D Reconstruction

Given a set of matches, SfM can produce a sparse reconstruction of the scene. Typically SIFT is used for detecting and matching features G. Lowe (2004) followed by camera pose estimation Martinec and Pajdla (2007); Snavely et al. (2008), and finally bundle adjustment Schonberger and Frahm (2016); Snavely et al. (2008); C. Wu et al. (2011). Modern SfM approaches showed great success in reconstructing 3D models for large scale areas from community photo collections shared on the internet, such as in Furukawa and Ponce (2007); Snavely et al. (2008). Another variant is incremental SfM which surpasses traditional SfM techniques in terms of robustness, accuracy, completeness, and scalability. Perhaps the closest work to ours is COLMAP Schonberger and Frahm (2016) where a general-purpose SfM system is proposed which incorporates an iterative bundle adjustment, retriangulation, and an outlier filtering strategy that improves completeness and accuracy

for large scale datasets.

5.4 Methodology

Image matches extracted using dense optical flow are encoded, verified for correctness, and stored in a database as explained in Sections 5.4.1, 5.4.1 Image Matching, 5.4.1 Feature Match Encoding and Verification, respectively. The reconstruction is performed using the matches as explained in Section 5.4.2.

5.4.1 Pre-processing

During the pre-processing step dense features are extracted and matched between the images. An out-of-core process performs redundancy checks in order to eliminate ambiguous matches [group of matches where the transitive relation does not hold, duplicates] and transforms the validated data into the internal representation used. Finally, the data is encoded and used to populate the database.

Image Matching

Dense features are extracted and matched in an N^2 fashion (complexity is $(N-1) \times (N-2)$ between pairs of images. Although any dense feature extractor and matching technique can be used e.g. SiftFlow C. Liu, Yuen, and Torralba (2010), FlowNet2.0 Ilg et al. (2017), etc, we employed Epic-flow Revaud et al. (2015). Epic-flow computes dense optical flow using a hierarchical, multi-layer, correlational architecture inspired by deep convolutional networks Revaud et al. (2016) even in the presence of large displacements. This matching algorithm can handle complex cases such as non-rigid deformations and repetitive textures, it efficiently determines dense correspondences in the presence of significant changes between images, and it has bidirectional validity checks.

Feature Match Encoding and Verification

Bundle adjustment is the common method for solving Structure-from-Motion problems. Perhaps the most popular variant of this method is the Sparse Bundle Adjustment which exploits the

(a) (b)

Figure 5.2: (a) A frame from our dataset, copyright of ©His Majesty the King in Right of Canada, as represented by the Minister of National Defence, 2022. (b) A rendered image from verified matches of its previous frame

sparse nature of the matrix to efficiently store, process, and solve for relatively large sets of parameters. This variant requires that all information about the matches, the 3D points corresponding to those matches, and the camera parameters are available in memory. When used with a sparse set of matches such as those produced by SIFT G. Lowe (2004), SURF Bay et al. (2006), ORB Rublee et al. (2011), etc, this does not constitute a problem however, there is always an upper bound on the number of parameters one can solve for and is typically restricted to a finite set of sparse features.

The second variant of bundle adjustment uses an iterative approach where a non-linear optimization is used to solve for the unknown camera and structure parameters. Until recently this method also required that all information is available in memory which again limited the size of datasets one could process. In COLMAP Schonberger and Frahm (2016), the authors presented for the first time how incorporating a database allows the processing and solving of larger sets of parameters however, this was again limited to a set of sparse features, though albeit larger than before, but which almost always contained ambiguous matches. In order to address these problems and ensure that only validated and unambiguous dense matches are used we represent the data in a format which allows for the efficient identification of redundancies. By redundancies we refer to (a) duplicate matches, and (b) matches where the transitivity property does not hold as shown in Figure 5.2a, if 2 or more feature points in one image are matched to the same feature point or its match, they will be removed. We achieve this by keeping two maps for each image in the dataset, an index map, and a conflict map. The conflict maps keep track of whether a feature point has a match in the next image, and the index maps store the information about the indices of feature points in each of the feature

tracks. This reduces the complexity of checking a feature point whether it is ambiguous, to 1. We render images for verification from verified matches of its previous frame, as shown in the example in Figure 3b, and compare to its original image, as shown in the example in Figure 5.2b.

Populating the Database

The internal data representation and redundancy check ensures that there are no duplicates and that for all matches the transitivity property holds. Next, we populate the database using this information. To speed up the recall time we encode the information as a single number and use a single table for storage instead of multiple tables Schonberger and Frahm (2016). This eliminates complex queries involving joins which are computationally expensive. A feature $f(i, x, y)$ contained in image I_i at pixel (x, y) is encoded as a single number $e(i, x, y) = i \star w \star h + y \star w + x$, where w, h are the width and height of the image respectively. Similarly the decoding of a number into the three tuple is performed as $x = code\%w$, $y = (code - x)\%(w \star h)/(w)$, $i = (code - x - y \star w)/(w \star h)$, where (x, y) are the image coordinates and i is the image index.

5.4.2 Bundle Adjustment

Bundle adjustment involves the simultaneous optimization of 3D points and camera poses based on the reprojection error. Using an initial estimate for the camera poses from the decomposition of the fundamental matrix between pairs of images, the initial 3D points are estimated via triangulation. The optimization proceeds by up- dating the 3D points and camera poses such that the reprojection error E is minimized Snavely et al. (2008) given by,

$E = \sum_i d_i(||Q(C_c, X_k), x_i||)^2$

where C_c are the camera parameters, X_k the points, and $Q(.,.)$ is a function which projects a 3D point onto the image plane corresponding to camera parameters C_c. d_i is a loss function that potentially down-weights outliers. A popular method for solving this type of problem is to store and factor the data as a sparse matrix or apply a non-linear optimization using Levenberg-Marquardt. Solving using a dense or sparse matrix requires N^2 memory space and has the complexity of $O(N^3)$ however for large-scale datasets it very often fails due to the large memory requirements imposed. On the other hand, solving using the iterative method has $O(N)$ time complexity and requires N

Method	VisualSFM C. Wu et al. (2011)	COLMAP Schonberger and Frahm (2016)	Proposed approach
Feature	SIFT G. Lowe (2004)	RootSIFT Arandjelovic and Zisserman (2012)	Epic-Flow Revaud et al. (2015)
# matches	-	704,127	1,576,705
SfM Variant	Bundler Agarwal et al. (2010)	Iterative BA Schonberger and Frahm (2016)	Iterative BA Schonberger and Frahm (2016)
# points	4.863	11,274	139,606
time-SfM(min)	1.10	11.051	12.533
MVS-Variant	PMVS	Kazhdan Kazhdan et al. (2006)	-
# points	111,189	287,205	-
time-MVS(min)	1.289	23.965	-
Run-time(min)	2.389	35.016	12.533

Table 5.1: A comparison of results of VisualSfM C. Wu et al. (2011), COLMAP Schonberger and Frahm (2016) and proposed approach in numberof matches, number of points and run time

memory space. However, the memory requirement can be reduced by computing the equation in batches.

Inspired by COLMAP Schonberger and Frahm (2016) and retriangulation, we use the iterative bundle adjustment method, since the generated dense flow between pairwise images leads to vast amount of points. This scheme is more efficient in our case, because the number of cameras is much smaller than the number of points, and we avoid performing large-scale matrix computations in memory.

5.5 Experiments

We run experiments on a dataset containing images taken from a helicopter circling around a church building. Our test dataset contains 71 images with resolution 1280×720 with unknown camera calibrations or EXIF information. We use the same dataset to evaluate our proposed method and compare it to the state of the art incremental SfM techniques, namely Bundler Snavely et al. (2008), VisualSFM C. Wu (2013), COLMAP Schonberger and Frahm (2016). The reconstructions are compared and evaluated by computing the distance of points set, as well as performing surface reconstruction and comparing the meshes.

One of the most popular feature extraction methods in Structure from Motion is SIFT G. Lowe (2004), in COLMAP Schonberger and Frahm (2016), all experiments use RootSIFT Arandjelovic and Zisserman (2012) features and match each image. With our dataset, each pair of the matched images have less than 1000 feature matches using SIFT, there are twice as many using RootSIFT.

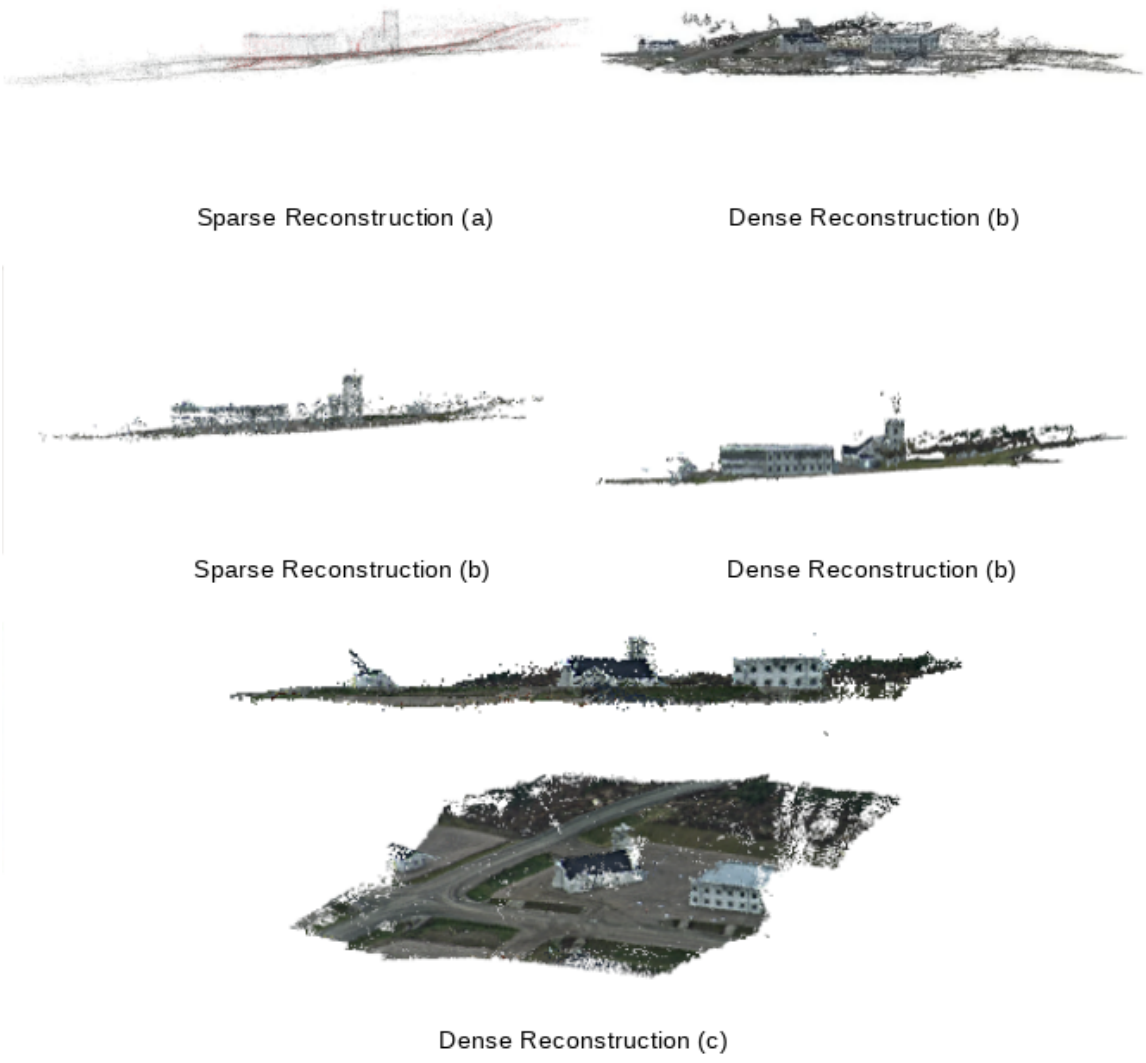


Figure 5.3: A comparison of results: 1. sparse reconstruction of SfM Sparse Reconstruction(a) and dense reconstruction of PMVS Dense Reconstruction(b); 2. Side-view of COLMAP SfM Sparse Reconstruction(a) and surface reconstruction Dense Reconstruction(b); 3. Side-view and top-view of direct SfM result ours (Dense Reconstruction (c)).

However, in our case, each of the images has almost the same amount of feature points as the resolution in the second image, thus an on-disk database for computing feature tracks becomes essential because of memory restrictions. Each of the rows in our database is converted to one feature track, then with COLMAP's Schonberger and Frahm (2016) iterative bundle adjustment we generate dense 3D reconstruction using only SfM and in a shorter time, as shown in Table 5.1. Rather than having a two steps process, i.e. SfM + MVS, which is time consuming and restricted by memory limitations, the proposed method generates a dense model efficiently and directly from SfM.

Figure 5.3 shows a comparison of our result with the results of both SfM sparse and, MVS dense reconstructions of state-of-the-art VisualSFM C. Wu (2013) and, COLMAP Schonberger and Frahm (2016).

We also performed a comparison between the sparse pointcloud produced by our approach and COLMAP Schonberger and Frahm (2016) by computing the nearest distance between points, and computing the mean distance of 0.01089, RMS 0.04825, with an overlap of 80%. Another comparison was performed between the reconstructed surfaces of the dense point clouds and computing the Hausdorff Distance of the two meshes.

Bundler Agarwal et al. (2010) generated disjoint groups of images (based on the matches) which led to multiple reconstructions of different scales, therefore, we were unable to quantitatively compare the results because considerable user interaction is required to manually align the reconstructions which introduced errors/bias.

Finally, We compared our reconstructed surface to Kazhdan's Kazhdan et al. (2006) and computed a mean distance of 0.013398 and RMS of 0.021225, respectively. As it can be seen, the proposed approach produces similar or better reconstructions (in terms of accuracy and density) with the dense techniques at a fraction of the time, using only a single step of SfM. It produces better results than all techniques (sparse or dense) except from Kazhdan et al. (2006) which takes three times longer to generate a result.

5.6 Conclusion

In this chapter, we have presented an improved method for performing single-shot reconstructions for large-scale datasets using only SfM. The method relies on a state-of-the-art optical flow technique to generate robust matches. The matches are further refined by verifying the correctness. An iterative bundle adjustment method is used to reconstruct the scene which is similar or better than other dense reconstruction state-of-the-art techniques.

5.7 Appendix A: Ablations of Single-shot Dense Reconstruction

5.7.1 Proposed Single-shot Dense Reconstruction

In our approach, we propose the method to conduct Structure-from-Motion (SfM) from optical flow, the proposed solution utilizes an on-disk data structure for organizing all the keypoints/matches, therefore eliminating the restrictions imposed by memory. The result of the SfM process is a dense pointcloud representing the scene. We compute optical flows from image pairs, which are pixel-wise matched in image space, then take all of the matches as the input to the next stage of Structure from motion and get our dense reconstruction. We proposed an approach of optical flow motion estimation for large displacement and deformations, namely HybridFlow, and applied it in the 3D reconstruction application. We use an out-of-core encoding approach to deal with a large amount of data, an advantage of the encoding is the fact that verification for correctness can be easily performed and ambiguous matches for which the transitivity property fails are removed. Lastly, an iterative bundle adjustment SfM is used which allows for the optimization of an arbitrary number of parameters and reconstructs the 3D scenes.

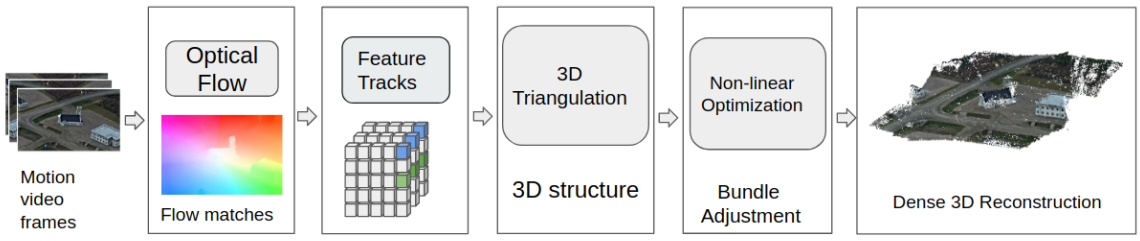


Figure 5.4: Proposed 3D reconstructions with optical flow pipeline

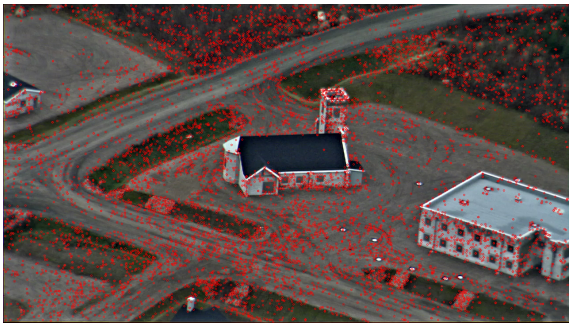
5.7.2 Comparison of Feature Matching

Sparse correspondences and Epic-flow

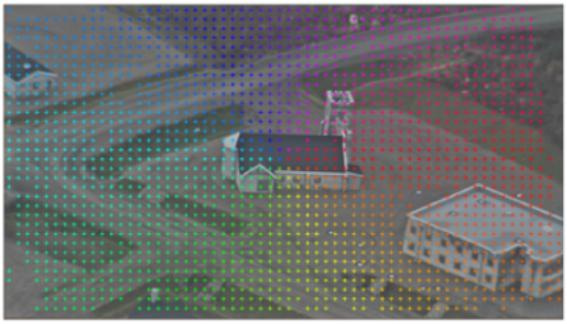
As from our review, typically the most popular descriptor for images is Scale Invariant Feature Transforms (SIFT) as shown in Figure 5.5a feature extractor, which was investigated for WAMI data, SIFT has been proven to be one of the most robust local invariant feature descriptors. The computation of SIFT contains four stages: scale-space extreme detection; keypoint localization; orientation assignment; and keypoint descriptor. COLMAP Schonberger and Frahm (2016) SfM system uses RootSIFT (show in Figure 5.5b) features and gets correspondences. Deep Matching Revaud et al. (2016), as an alternative of performing dense wide-baseline matching by first matching a few feature points and triangulating them, then locally rectifying the images, however, in this case, if some matches are mistakenly computed and are even not detected gross reconstruction errors will occur and cause incorrect reconstruction results. When it comes to the WAMI images, the unique features of multiple targets, weak target texture, image scale, and environment occlusions, lead to the failure of image matching. DeepMatching (shown in Figure 5.5c) computes dense correspondences between images, Which relies on a hierarchical, multi-layer, correlational architecture designed for matching images and was inspired by deep convolutional approaches. The proposed matching algorithm can handle non-rigid deformations and repetitive textures and efficiently determines dense correspondences in the presence of significant changes between images.

Despite the sparse feature matching, dense optical flow techniques compute the dense matches as input to the later reconstruction. Figure 5.6 shows a comparison of feature matching. As the state of the art of most recent works, Epic-Flow Revaud et al. (2015) is an edge-preserving interpolation of correspondences for optical flow, it leverages recent advances in matching algorithms and introduces an edge-aware geodesic distance that handles motion discontinuities and occlusions. We choose to use this method in the single-shot dense reconstruction Q. Chen and Poullis (2018) (Chapter 5) to compute our dense matches because of its ability to handle deformations and repetitive textures. Epic-flow computes the dense optical flow using a hierarchical, multi-layer, correlational architecture inspired by deep convolutional networks even in the presence of large displacements. This matching algorithm can handle complex cases such as non-rigid deformations and repetitive

(a) SIFT



(b) RootSIFT



(c) Deep-matching

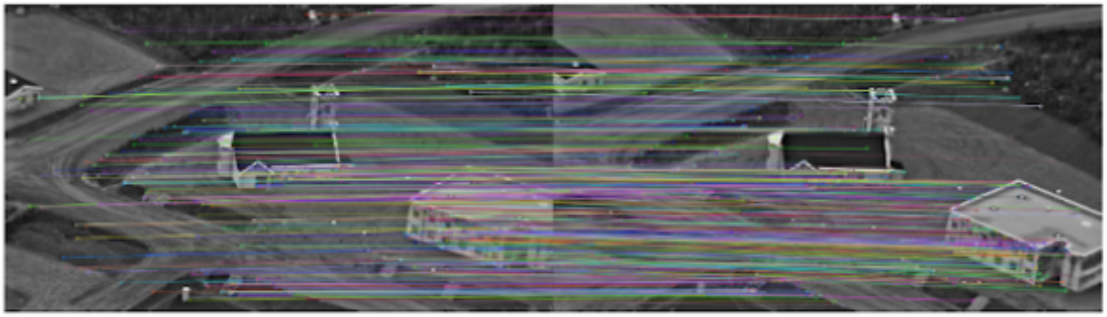
(d) Epic-flow

Figure 5.5: A comparison of features: (a) SIFT G. Lowe (2004), (b) RootSIFT Arandjelovic and Zisserman (2012) , (c) Deep-matching Revaud et al. (2016) and Epic-flow Revaud et al. (2015)

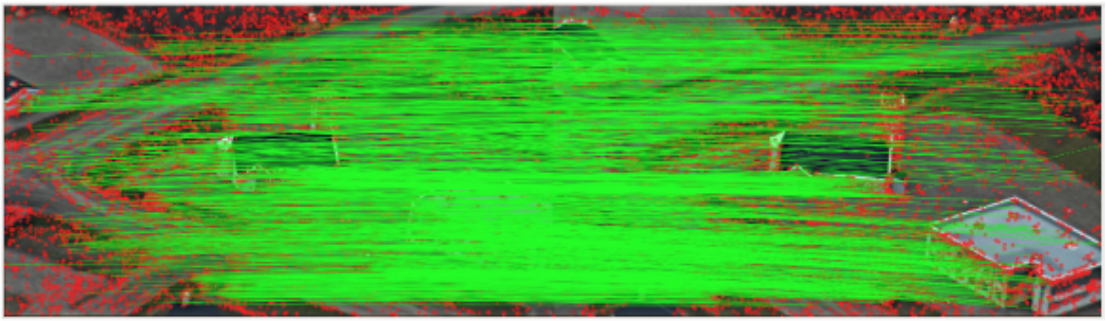
textures, it efficiently determines dense correspondences in the presence of significant changes between images, and it has bidirectional validity checks. Ideally, instead of getting sparse reconstructions from sparse features, the robust and accurate flow estimation, i.e. pixel-wise motion matches give us denser correspondences which lead to a dense 3D reconstruction in the end.

HybridFlow and Other Optical Flow

HybridFlow Q. Chen and Poullis (2022b) is proposed to improve the robustness and accuracy in the presence of large displacements and deformations, its superiority among the variational optical flow techniques is demonstrated in detail in Chapter 4. The comparison of HybridFlow and other state-of-the-art variational optical flow techniques are shown in Figure 5.7 shows a qualitative comparison between HybridFlow, EpicFlow and RicFlow, as can be visualized and shown from Chapter 4 Table 4.1, the matching accuracy of HybridFlow is higher than the other methods. Chapter 4 Table 4.2 and Figure 4.8 also demonstrate the effectiveness of HybridFlow on large-scale 3D



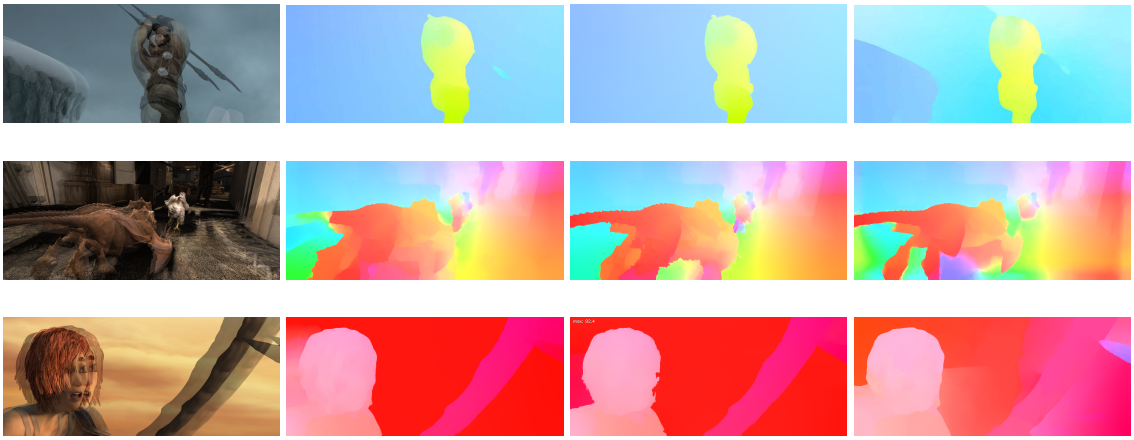
SIFT (a)



RootSIFT (b)

Epic-flow (c)

Figure 5.6: A comparison of matching: 1. SIFT G. Lowe (2004) (SIFT(a)); 2. RootSIFT Arandjelovic and Zisserman (2012) (RootSIFT(b)); 3. Epic-flow Revaud et al. (2015) (Epic-flow(c)).



(a) Images (b) EpicFlow Revaud et al. (2015) (c) RicFlow Hu et al. (2017) (d) **ours**

Figure 5.7: Results on MPI-Sintel. The first column is the combined left-right image, the second is EpicFlow, the third RicFlow, and the last column is HybridFlow.

reconstruction.

Initial Matches

Interpolation and refinements are essential steps for variational optical flow techniques. In HybridFlow, we apply a standard interpolation and refinement method also used by EpicFlow Revaud et al. (2015) and RicFlow Hu et al. (2017). The interpolation uses distances between the initial matches and applies a coarse-to-fine scheme, which yields an initial edge-preserving estimate of the optical flow. The final optical flow estimation is performed by variational energy minimization using the interpolated dense matches for initialization. The outcome of these two steps solely depends on the accuracy and sparsity of the initial matches.

To demonstrate how critical the interpolation and refinement steps to our approach, we conducted an additional experiment to quantitatively compare the results of our approach and two state-of-the-art without using the interpolation and refinement steps i.e. comparison of the sparse matches with the same matches in the ground truth. We compare with EpicFlow and RicFlow which rely on Deep Matching for the initial sparse matches. In this context, precision is defined as the percentage of matches with an error lower than 10 pixels. Although large, this threshold was used to be consistent with the Deep Matching paper. For lower thresholds, our approach vastly surpasses

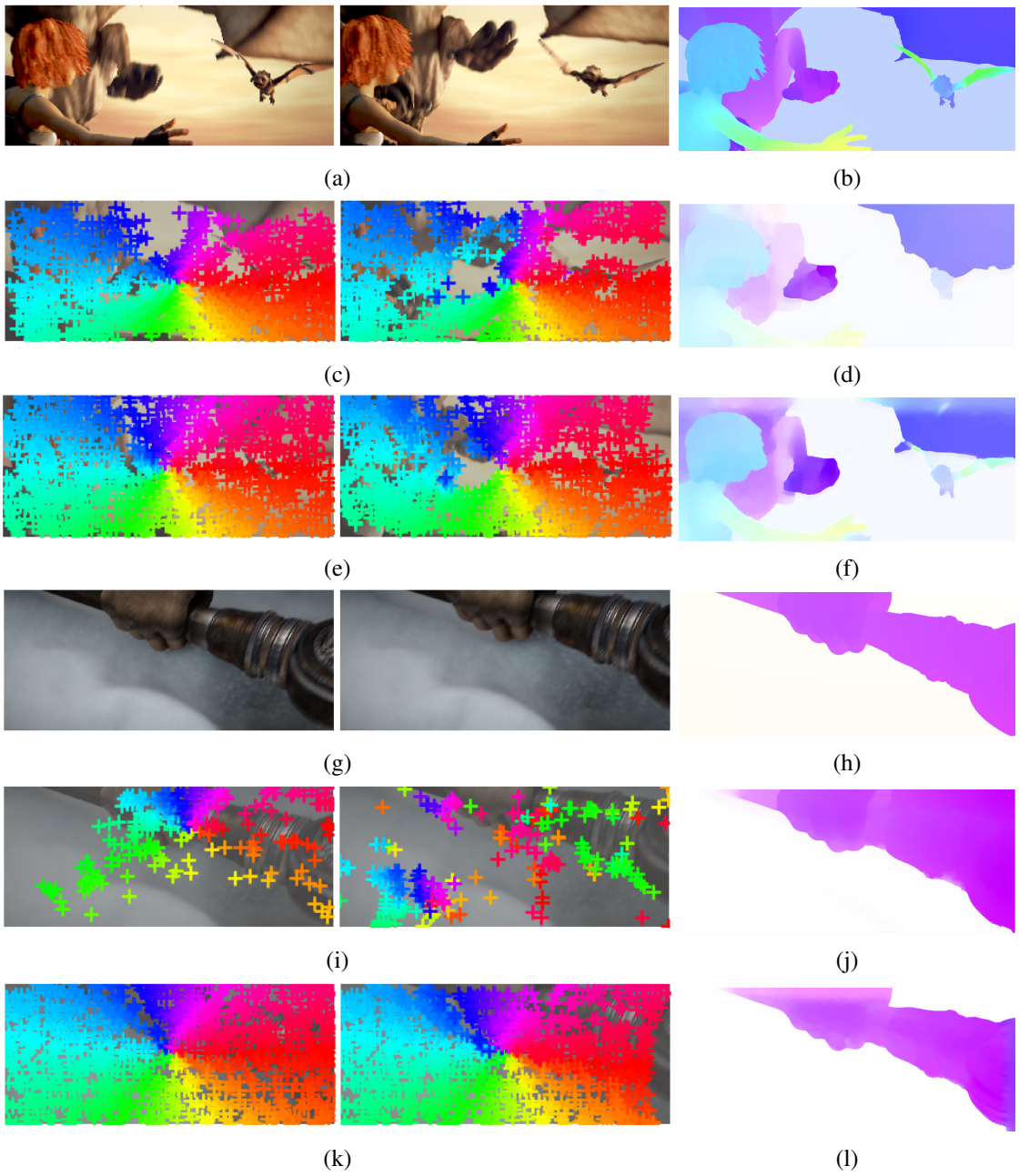


Figure 5.8: (a)(g) shows the examples of image input, (b)(h) shows the correspondent ground truth flow; (c)(i) shows the Deep Matching results, where the number of matches is 3996 and 260, precision is 84.57% and 65.21% respectively, (d)(j) shows the Epicflow results; (e)(k) shows the initial matches of proposed Hybridflow, where the number of matches is 9052 and 8278, precision is 96.15% and 90.30% respectively, (f)(l) shows the HybridFlow results. The overall quantitative optical flow comparison can be found in Chapter 4 Table 4.1.

Deep Matching with and without interpolation and refinement. In our approach, we use a per-pixel feature descriptor, however, to ensure a fair comparison we have also used every 4^{th} pixel on each

row and column, similar to Deep Matching. The experiment was conducted on 2 sets of MPI-Sintel Butler et al. (2012) containing 50 images each. The average precision of Deep Matching on these two sets is 89.11%, whereas the average precision of HybridFlow on the same datasets is 93.18% clearly demonstrating the superiority of our approach. Figure 5.8 shows a qualitative comparison of the results.

5.8 Appendix B: Extending to Large-scale Image Processing

5.8.1 Image Tiling

Algorithm 2: Image tiling approach of matching images in pseudo code

```
Data: Input images
Result: Every pair of images matched using DeepMatching for each of the image tiles
Initialization tile every image (1280pixels × 720pixels)into 100pixels × 100pixels
  pieces;
Function GetNeighbor image (i):
    tiles = Tile In Next Image;
    line = CALLComputeCorrespondEpilines();
    while tile IN tiles do
        if intersect (line, tile ) == True then
            matchedTiles.pushback(tile);
        else
            Continue;
        end
    end
    return matched Tiles ;
;
Function Main(img1, img2):
    tile = Tile from Img1;
    tileList = CALL Flood-fill(tile) Getting all the tiles in sequence;
    while tile IN tileList do
        matchList = CALLf function GetNeighbor;
        Matched Tile = RANSAC(matchList) ;
    end
    return Best Tile IN Matched Tile ;
return
```

WAMI images are large-scale in both camera coverage and image size, and most of the optical flow techniques Hu et al. (2017); Revaud et al. (2015) are not capable of processing the full-sized

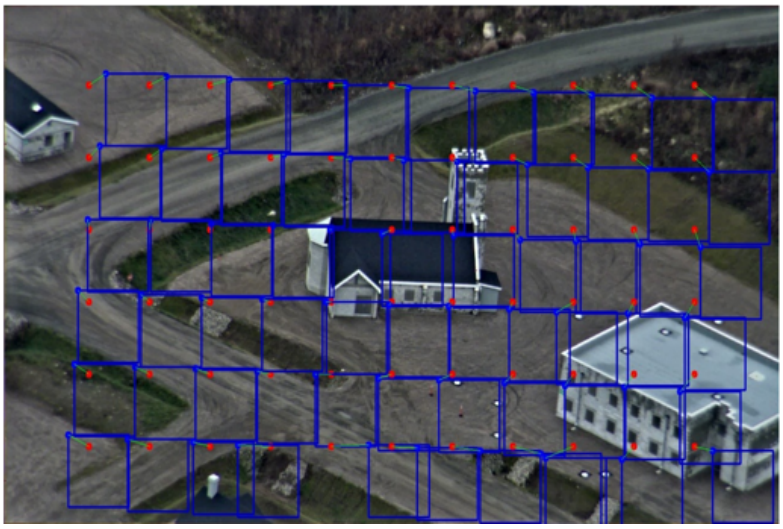


Figure 5.9: Motion of tiles

Figure 5.10: Regenerated image

($6K \times 8K$) WAMI images due to memory limitations.

Given the WAMI (Wide Area Motion Imagery) data input, the experiments are carried out mainly focused on dense 3D reconstruction from large-scale 2D images. For dense matching methods, especially flow estimations, the cost of computation is quite large. And dealing with large-scale images the usage of memory and consumption of time is sometimes not affordable for most of the scenarios. Our first approach of the WAMI imagery pre-processing is to cut the images into smaller-sized tiles. The advantages of this approach are:

1) previous algorithms and methods can be applied to our cases,

2) less computation and space complexity.

However, using WAMI images in small parts of a regular grid cause the trouble of:

1) inconsistency of the targets,

2) loss of global perspective

Considering the size of the imagery and resolution required for the dense reconstructions, we perform pre-processing of tiling the images and search for the best match of each of the tiles. The current approach can be described as in the Algorithm 2:

Firstly, getting the center (in image space) of the image and matching it with the tile from the other image's center at the same location and its neighbor tiles. Secondly, collecting all the matching points and getting the sum score for each of the tiles, and getting the highest scored tiles. Then, using all the points from the highest scored tiles for RANSAC, to compute the translation of the matched tiles. Lastly, based on the center tile's translation, each of its neighbor tiles applies the same translation and computes their corresponding translation. The global Fundamental matrix helps to filter out tiles going too far away.

By tiling the full-sized images, traditional optical flow methods are able to handle images of any size without loss of the consistency of the targets. This experiment performed using the test dataset contains 71 images with resolution 1280×720, as an example of tiles shown in Figure 5.9 and recreated frame from its matched tiles as shown in Figure 5.10.

Learning-based Feature Extraction

The image tiling process described in section 5.8 can possibly fail in settings where the lighting effects are inadequate or overexposed, there are atmospheric effects, motion and defocus blur, or the tiles are texture-less. The alternative implementation is to generate pixel-wise feature descriptors, i.e. dividing the image into a regular grid and performing feature extraction per pixel. Since hand-crafted feature descriptors are not designed for this, feature descriptors such as SIFT G. Lowe (2004), SURF Bay et al. (2006) or ORB Rublee et al. (2011) etc. will have problems of generating same descriptors per patch or meaningless feature descriptors for most of the pixels in the image. HybridFlow integrates three types of pixel-wise descriptors: (i) rootSIFT descriptors, named as HybridFlow(SIFT), (ii) features descriptors extracted from a pre-trained ResNet He et al. (2015) trained on MPI-Sintel Butler et al. (2012), named as HybridFlow(DeepLab), and (iii) descriptors learned by feature and context encoder as in RAFT Teed and Deng (2021), name as HybridFlow. HybridFlow outperforms all other state-of-the-art variational techniques and gives comparable results to the deep-learning-based feature extraction techniques.

5.9 Appendix C: Feature Tracking

5.9.1 On-disk Matches

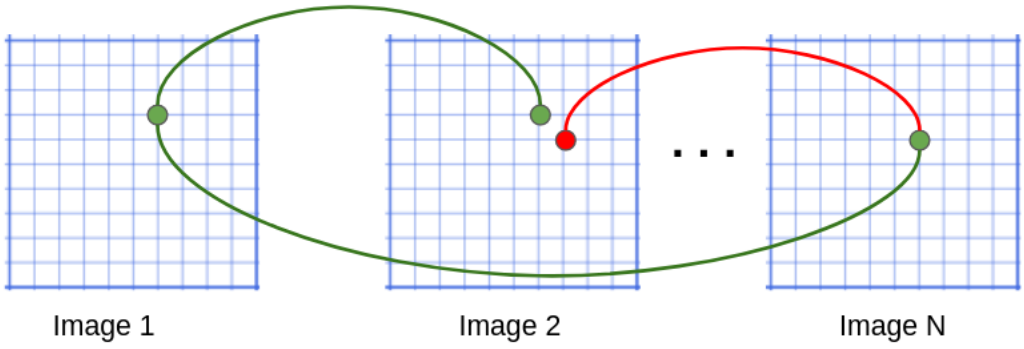


Figure 5.11: Transitivity of matches

After pair-wise feature matching, an essential step in reconstruction is to do the feature points tracking. Feature point tracking means following up on the position of a characteristic point in a

set of images. These multi-view correspondences are called tracks. Track identification in a set of images is an important task in computer vision. Given a set of matches, SfM can produce a sparse reconstruction of the scene. During the pre-processing step, dense features are extracted and matched between the images. An out-of-core process performs redundancy checks in order to eliminate ambiguous matches [group of matches where the transitive relation does not hold, duplicates] and transforms the validated data into the internal representation used. Finally, we carried out experiments on encoding and populating the data to (a) the database (published in the paper Q. Chen and Poullis (2018)); (b) Dynamic Data structure to process the match tracks (applied in the paper Q. Chen and Poullis (2022b)). Dense features are extracted and matched in an N^2 fashion (complexity is $(N-1) \times (N-2))$) between pairs of images, N indicates the number of images.

In order to address these problems and ensure that only validated and unambiguous dense matches are used, we represent the data in a format that allows for the efficient identification of redundancies. By redundancies, we refer to (a) duplicate matches, and (b) matches where the transitivity property does not hold as shown in Figure 5.11, if a point A in $Image1$ is matched to point B in $Image2$ and point D in $ImageN$, while another point C in $Image2$ is also matched to point D, they are defined as ambiguous match, i.e. if 2 or more feature points in one image are matched to the same feature point or its match, they will be removed. We achieve this by keeping two maps for each image in the dataset, an index map, and a conflict map. The conflict maps keep track of whether a feature point has a match in the next image, and the index maps store the information about the indices of feature points in each of the feature tracks. This reduces the complexity of checking whether a feature point is ambiguous. Alternatively, we propose combining the super-pixel constraint with the dense match validation. To improve the robustness of determining whether a feature point is ambiguous, we label feature tracks that possess transitivity as inliers and conflicted feature tracks as outliers. We proposed a tracking system of super-pixels and the validation of matches by keeping a record of pixel motion within every super-pixel and validating based on the inliers and the outliers. Within each of the super-pixels, the percentage of inliers should be higher than the percentage of outliers; otherwise, the super-pixel is not correctly matched by definition.

5.9.2 Tensor Storage and Data Validation

The database approach solves the problems of validation and ambiguity of dense matches, however, for each transaction of adding of matches, the time complexity of connection to the database and queries are considerably heavy. We come up with the dynamic multi-dimension array structure for the dense matches are used we represent the data in a format which allows for the efficient identification of redundancies. This reduces the complexity of populating the database and checking a feature point whether it is ambiguous.

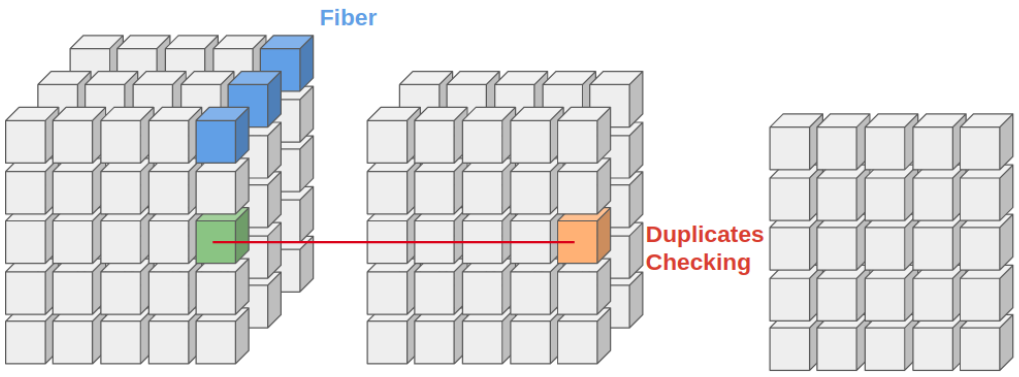


Figure 5.12: Dynamic tensor data structure

For each of the images, we create a dynamic 3D structure to store the information of its feature locations in 2D image space, as well as the matches in other images. For example, a feature point (x, y) from the first image is stored in location from structure $S_0(x, y, 0)$, its matches from the N frame will be at location $S_0(x, y, n)$, and the 1 to n dimension is what we named fiber, as an example is shown in Figure 5.12. From each fiber, we can find all the 2D coordinates in image space, and reconstruct a point in 3D space. While a fiber is presented in an image, we'll have the duplication checking and not creating it from other tensors.

Chapter 6

End-to-End Multi-View Structure-from-Motion

Convolutional Neural Networks (CNNs) are increasingly utilised in the context of 3D modelling, as their practical advantages over conventional methods have been apparent in a variety of computer vision-related areas and beyond. The current era may be characterised by the prevalence of deep learning techniques, which set the standard for precision and efficacy. This chapter is based on the work in the paper "End-to-End Multi-View Structure-from-Motion" Q. Chen and Poullis (2022a) published in the IEEE International Conference on Signal Processing, Sensors, and Intelligent Systems (SPSIS 2022).

6.1 Abstract

Image-based 3D reconstruction is one of the most important tasks in Computer Vision with many solutions proposed over the last few decades. The objective is to extract metric information i.e. the geometry of scene objects directly from images. These can then be used in a wide range of applications such as film, games, virtual reality, etc. Recently, deep learning techniques have been proposed to tackle this problem. They rely on training on vast amounts of data to learn to associate features between images through deep convolutional neural networks and have been shown

to outperform traditional procedural techniques. In this chapter, we improve on the state-of-the-art two-view structure-from-motion (SfM) approach of J. Wang et al. (2021) by incorporating 4D correlation volume for more accurate feature matching and reconstruction. Furthermore, we extend it to the general multi-view case and evaluate it on the complex benchmark dataset DTU Jensen, Dahl, Vogiatzis, Tola, and Aanæs (2014). Quantitative evaluations and comparisons with state-of-the-art multi-view 3D reconstruction methods demonstrate its superiority in terms of the accuracy of reconstructions.

Index Term – Deep Learning, Structure-from-Motion, Multi-View-Stereo, 3D Reconstruction

6.2 Introduction

3D perception is an important ability of the visual system that can improve scene understanding. With the rapid development of computer technology, 3D reconstruction techniques are playing an increasingly important role in all aspects of industry and production, such as preservation of cultural heritage, virtual reality and other fields. Perhaps the most popular and successful 3D reconstruction technique in recent years is based on Structure-from-Motion and multi-view stereo. This is due to the low equipment cost, high operational flexibility, and good reconstruction accuracy it provides.

Recently, deep learning models have been proposed to address vision tasks including 3D depth estimation from images. Given a large amount of annotated data, a model can learn the nonlinear mapping between source and target domains. It has been shown that deep learning models can also perform feature detection, pose estimation, landmark localization, and image recognition tasks. The results achieved are remarkable; in many problems related to expression learning, scholars have also applied it to multi-view learning, resulting in a number of works on multi-view stereo such as Yao et al. (2018, 2019).

In this chapter, we provide the research background on multi-view reconstruction and explore related concepts, including an overview of 3D reconstruction methods, the concept of multi-view features, and Structure-from-Motion reconstruction and multi-view stereo. Next, we present an end-to-end model for multi-view Structure-from-Motion with hypercorrelation volume which results in high accuracy feature matching and improved 3D reconstructions. Lastly, we evaluate our technique

on the complex multi-view stereo benchmark dataset DTU Jensen et al. (2014) and present a quantitative evaluation and comparison with state-of-the-art 3D reconstruction methods demonstrating the superiority of our approach.

6.3 Related Work

6.3.1 3D reconstruction

3D reconstruction has experienced a long process of improvement, produced rich results, and played an important role in manufacturing and production nowadays. In some traditional fields, such as in industrial manufacturing, reverse engineering based on 3D reconstruction technology can help producers effectively improve product quality; for example, in cultural heritage protection, building 3D models is the digital preservation of cultural relics. In some emerging fields, such as virtual reality, high-precision 3D reconstruction technology brings users a more immersive experience. In addition, 3D reconstruction is also widely used in the fields of smart medical care, urban planning, and autonomous driving. With the continuous improvement of science and technology, the application scope and application requirements of 3D reconstruction are also expanding. Therefore, it is particularly important to study accurate and robust 3D reconstruction algorithms.

3D reconstruction can be divided into active and passive reconstruction based on the way of acquiring 3D models. Active reconstruction is scanning the object from all directions using a three-dimensional scanning device, and directly obtaining the three-dimensional model of the object to be reconstructed. Using the energy source emitted by the 3D scanning device itself, the 3D scanning device mainly refers to a laser rangefinder or a structured light scanner. Structured light scanners have lower requirements for the scanning environment, and the scanning results can be used as auxiliary tools for other applications. However, active reconstruction using structured light has its disadvantages. For example, it cannot directly and accurately obtain texture information related to the application scene, and it cannot be directly applied to a wide range of open scenes. Moreover, the cost while using the 3D scanning equipment, as well as the price of the equipment itself, is also very expensive. It is only used in specific situations, and the active reconstruction has more limitations. In passive reconstruction, images are obtained directly through the traditional RGB cameras, and then

a 3D reconstruction is performed on the images using algorithms such as Structure-from-Motion. In the actual application process, passive reconstruction directly uses the camera to capture the image, performs feature matching between the images, and then triangulates to calculate the depth information of the reconstructed scene to restore the three-dimensional structure of the objects or the scene. Compared with active reconstruction, passive reconstruction relies on calculating 3D points rather than measuring them, it does not require expensive acquisition equipment and has the advantages of a large measurement range and high variability, which is convenient for manual operation. The simplest method only requires cameras and computers.

6.3.2 Structure-from-Motion

Structure-from-Motion(SfM) is a feature-based 3D scene reconstruction algorithm in the field of multi-view geometry reconstruction in computer vision. The algorithm recovers camera parameters and scene structure information by analyzing the motion process of the camera relative to the target scene. By inputting images of a target from different perspectives, the overall three-dimensional structure of the scene is recovered. At present, the algorithm is mainly used to recover the camera motion trajectory through video successively tilt or recover the three-dimensional structure information of the scene through the multi-view picture set of the same scene. Snavely et al Snavely et al. (2008) and Furukawa et al Furukawa, Curless, Seitz, and Szeliski (2010) proposed the modern Structure-from-Motion and dense patch matching reconstruction from unstructured 2D images. Schonberger et al Schonberger and Frahm (2016) revisited this problem and proposed a state-of-the-art procedural Structure-from-Motion pipeline, which is widely used in other applications and even used to generate ground truth in deep learning approaches.

6.3.3 Deep Learning Multi-View 3D Reconstruction

With recent active research on deep learning, an increasing number of neural network-based 3D reconstruction methods have been proposed. Most deep learning neural networks employ a convolutional neural network (CNN) Krizhevsky, Sutskever, and Hinton (2012) and can be categorized into two types. In the first category T. Zhou et al. (2017), similar to Structure-from-Motion, the problem can be summarized as a joint optimization task of monocular depth and camera poses; in

the second category Yao et al. (2019), similar to multi-view stereo, the camera poses are known and the depths are iteratively refined via multi-view geometry. The main difference is the recovered 3D information density and multi-view stereo techniques perform reconstruction on implicit surfaces and radiance fields.

In this chapter, we extend a deep two-view Structure-from-Motion method to the more general task of multi-view reconstruction. Different from the two-view camera and depth estimation, the multi-view stereo tasks require both, camera recovery and depth estimation. In this work, we show deep learning-based multi-view reconstruction on a complex dataset i.e. DTU benchmark dataset Jensen et al. (2014) and carry out the comparison of the 3D models between this approach and state-of-the-art multi-view stereo techniques.

6.4 Methodology

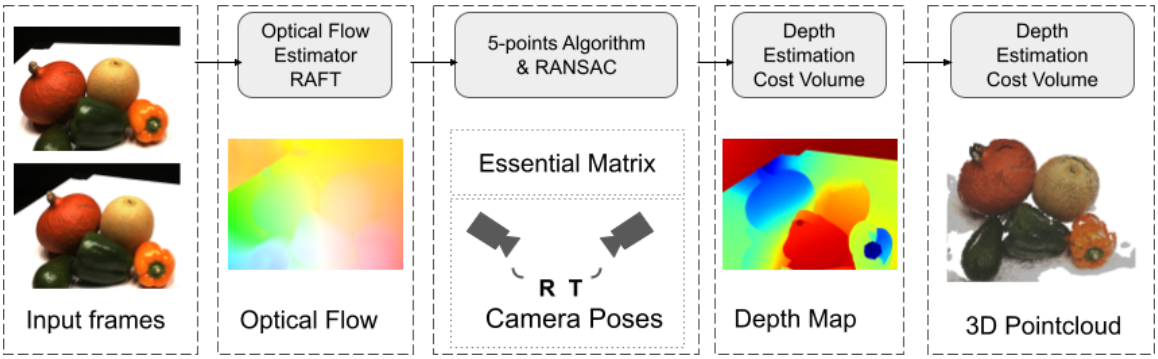


Figure 6.1: Overview of the 3D reconstruction pipeline.

The proposed framework follows the classic pipeline for 3D reconstruction from multi-view images where the features in images are matched, followed by the camera pose and the relative depth map estimation. In the last step, homography warping generates the final output. The pipeline is shown in Figure 6.1. Below, we explain the two main steps: (i) the Structure-from-Motion based on deep learning that results in the relative depth maps between pairs of images, and (ii) multi-view reconstruction with homography warping which results in the final pointcloud.

6.4.1 Structure-from-Motion with Deep Learning model

Structure-from-Motion involves the three main steps, feature matching, camera recovery, and depth map estimation.

Feature Matching

The first step is matching features between pairs of image frames. Unlike traditional techniques which rely on sparse feature correspondence, we employ an optical flow estimation network that predicts dense correspondences between pairs of images. Deep learning models for optical flow have been shown to generate per-pixel dense matches that are robust to different textures, occlusions, large displacements and deformations. We integrate the state-of-the-art deep optical flow network RAFT Teed and Deng (2021) to generate dense feature matches. The RAFT architecture extract per-pixel features, it employs multi-scale 4D hypercorrelation volumes, and iteratively updates the flow field through a recurrent unit. The model is pretrained on FlyingChairs Dosovitskiy et al. (2015) and FlyingThings Mayer et al. (2016), followed by fine-tuning on the benchmark dataset, e.g. MPI-Sintel Butler et al. (2012) and achieves an end-point error (EPE) of 2.855 pixels, one of the lowest in recent years.

Given a pair of consecutive RGB images I_1 and I_2, features are extracted from the input images using a convolutional network. The feature encoder and context encoder extracts per-pixel features. The encoder network is applied to both images and maps them to dense feature maps. The correlation layer computes the visual similarity between pixels. Given the image features maps, the hypercorrelation volume is formed by calculating the cosine similarity between all pairs of per-pixel feature vectors. A 4-layer pyramid is constructed by pooling the last two dimensions of the correlation volume with kernel sizes 1, 2, 4, and 8, followed by lookups on all levels of the pyramid to compute each correlation value. Lastly, the update operator imitates the process of an iterative optimization, with the iterative updates, the update operator estimates a sequence of flow estimation and produces an update direction that applies to the current estimation. The update operator takes the flow and correlation as an input and outputs the update direction until its convergence to a fixed point. The update operator works as an energy minimization and optimization function which

outputs the final optical flow estimation.

Camera Recovery

The normalized pose estimation module computes relative camera poses from the 2D optical flow correspondences. Given a set of matching points and the camera intrinsic matrix K, the essential matrix E can be recovered from the five-point algorithm Nistér (2004). Given a pair of matches $\boldsymbol{x}_i$ and $\boldsymbol{x'}_i$ and the camera intrinsic matrix K, Structure-from-Motion finds a camera rotation matrix R and a translation vector T. The point X_i is the result of the triangulation of the corresponding points given by,

$$\boldsymbol{x}_i = K[I|0]Xi \qquad\qquad \boldsymbol{x'}_i = K[R|T]Xi$$

where I is the identity matrix. To recover the essential matrix E we need at least 5 points, followed by R and T decomposition from E. After the triangulation of the matched points, x_i and x'_i using the camera poses, we get the 3D points represented as a depth map for each pair of images in the image sequence.

Depth Estimation

Although the camera recovery results in dense depth maps, the depth estimation process samples a subset of these otherwise it would not take advantage of the epipolar constraint and re-calculates the depth. This process is similar to multi-view stereo, where the standard plane-sweep algorithm samples the distribution of matching candidates and estimates the depth. We train the model in an end-to-end manner and supervise it using ground truth depth maps. Moreover, to make the depth estimation scale-invariant, the translation vectors are normalized and the matching candidates are depending on the camera poses and the scale factor. The loss function is given by,

$$\boldsymbol{L}_{depth} = \sum_{\boldsymbol{x}} l_{huber}(\hat{\boldsymbol{d}}_{\boldsymbol{x}} - \boldsymbol{d}_{\boldsymbol{x}})$$

$$\boldsymbol{L}_{flow} = \sum_{\boldsymbol{x}} (\hat{\boldsymbol{o}_{\boldsymbol{x}}} - \boldsymbol{o}_{\boldsymbol{x}})^2$$

$$\boldsymbol{L}_{total} = \boldsymbol{L}_{depth} + \boldsymbol{L}_{flow}$$

where $\hat{\boldsymbol{d}_{\boldsymbol{x}}}$ is the predicted depth and $\boldsymbol{d}_{\boldsymbol{x}}$ is the ground truth depth, $\hat{\boldsymbol{o}_{\boldsymbol{x}}}$ is the predicted optical flow and $\hat{\boldsymbol{o}_{\boldsymbol{x}}}$ is the ground truth optical flow. The total loss $\boldsymbol{L}_{total}$ can be calculated from the sum of the depth loss $\boldsymbol{L}_{depth}$ and optical flow loss $\boldsymbol{L}_{flow}$, the Huber function is given by,

$$l_{huber}(z) = \begin{cases} 0.5z^2, \text{if } \mid z \mid < 1 \\ \mid z - 0.5 \mid, \text{otherwise} \end{cases}$$

where z is the estimated depth.

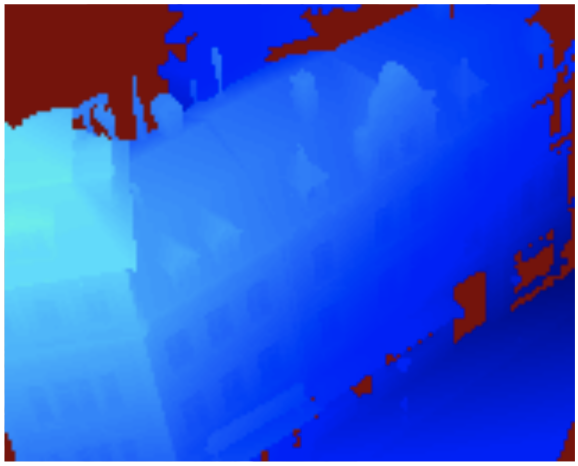
6.4.2 Multi-view 3D Reconstruction

Reconstruction of the scene geometry is the last step. Given the pair-wise depth maps, we merge the reconstructions into a global point cloud using homography warping which implicitly uses camera geometries to build the 3D volume. To generate the 3D point cloud from the per-view depth maps, we apply depth map fusion, which integrates depth maps from different views to a unified point cloud representation. In the visibility-based fusion algorithm Yao et al. (2018), depth occlusions and violations across different viewpoints are minimized and the average overall reprojected depth is taken as the final depth estimation of the pixel. Finally, the fused depth maps are directly reprojected to generate the 3D point cloud.

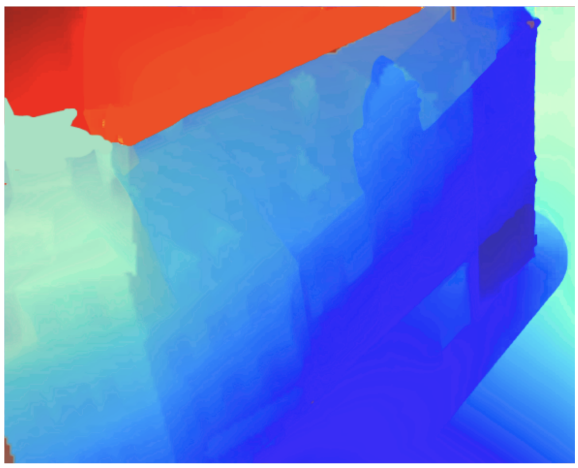
6.5 Experiments

We implemented our framework in Python3.8 with Pytorch 1.6. All experiments were conducted on a workstation with an Intel i7 processor and a NVIDIA GTX 3080Ti graphics card.

(a)



(b)



(c)

Figure 6.2: The reference image (a), the ground truth depth map (b) and the output depth map (c).

6.5.1 Dataset

The dataset used in this chapter is the benchmark dataset DTU presented in Jensen et al. (2014). The DTU dataset is a large-scale multi-view 3D reconstruction dataset collected in a strictly controlled laboratory environment. The real-world annotations provided by this dataset are the point cloud data collected by the structured light scanner, which allows for the quantitative evaluation of the results of the 3D reconstruction. The dataset consists of 124 different scenes rotated and scanned four times at 90-degree intervals, hence giving a complete view of the models. Each scene was captured with 49 or 64 images with a resolution of 1600×1200.

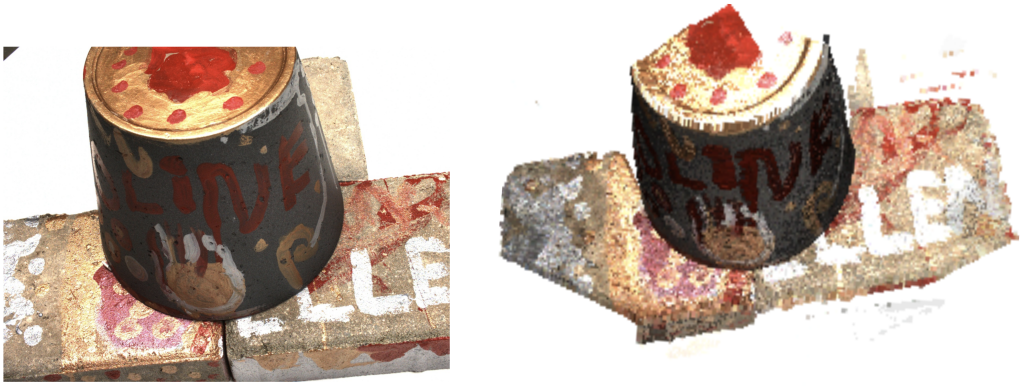
6.5.2 Quantitative Analysis

In this section, we present a quantitative evaluation and comparison between the proposed deep learning-based multi-view Structure-from-Motion framework and state-of-the-art deep multi-view 3D reconstruction methods.

Figure 6.2 shows an example of depth estimation and the qualitative comparison with the ground truth. The average error for the test dataset is 7.43, calculated from pixels with end-point-error greater than 3 pixels. Figure 6.3 shows the resulting 3D pointcloud and a qualitative comparison with ground truth. Our method achieves an overall mean distance of 0.411. We compare with MVSNet, R-MVSNet and SurfaceNet on 17 scenes from the DTU dataset. As shown in Table 6.1, our method achieves state-of-the-art performance in mean accuracy and mean completeness at faster computational times.

Method	Mean Acc.	Mean Comp.	Overall (mm)	Runtime(s)
MVSNet	0.396	0.527	0.462	15.12
R-MVSNet	**0.385**	0.459	0.422	23.19
SurfaceNet	0.450	1.04	0.745	-
Ours	0.391	**0.429**	**0.411**	**2.19**

Table 6.1: Mean Acc. is the mean accuracy of the distance metric (mm) and Mean Comp. is the mean completeness of the distance metric (mm). Runtime is the time of depth estimation for a pair of images.



(a) (b)

(c)

Figure 6.3: The input image, ground truth, and output 3D reconstruction, respectively.

6.6 Conclusion

In this chapter, we addressed the vision task of 3D reconstruction from images. Deep learning techniques have been shown to outperform procedural techniques. We incorporated a deep learning-based optical flow technique that uses hypercorrelation volumes to achieve accurate dense matching between images to a state-of-the-art two-view deep learning technique. We further extended this technique to the multi-view case. The evaluation on a complex benchmark dataset and further comparison with other state-of-the-art techniques show that the proposed improvement is superior in terms of accuracy and performs faster. In the future, we plan on exploring alternative architectures for the reconstruction that would further improve the final accuracy.

Chapter 7

Application: Tracking and Identification of Ice Hockey Players

(a) Example of broadcasting camera.

(b) Example of input frame from an ice hockey game video

(c) The optical flow estimation of HybridFlow Q. Chen and Poullis (2022b) with (b) to the next frame.

Figure 7.1

As shown in Figure 7.1, similar to Wide Area Motion Imagery (WAMI) input, another application of HybridFlow Q. Chen and Poullis (2022b) is in sports streaming. This chapter is based on the paper "Tracking and Identification of Ice Hockey Players" currently under review by ACM Multimedia Systems, 2023. We present a real-world application of motion flow to the detection, reidentification, and tracking of ice hockey players.

7.1 Abstract

Ice hockey has the fastest registered speeds in non-motorized sports, making it inherently challenging to track the players. We present a complete framework for player identification and tracking, which fine-tunes state-of-the-art deep neural networks on data captured from ice hockey games. A region proposal technique detects persons in a sequence of images. Ambiguities due to the similar appearance among players of the same team are resolved using a text detector model, which performs character recognition on regions containing text detected by a scene text recognition model. After the identification of the players, tracking is performed using a visual multi-object tracking model. We report on experiments on data captured from real ice hockey games.

Key Words – computer vision in sports, player identification, jersey numbers, ice hockey

7.2 Introduction

Automated sports video analysis is increasingly gaining significant interest from the computer vision community. Knowing the player movements and their game statistics makes it more engaging for the spectators. More importantly, accurate analytics give insight into the decision-making of the game plan by facilitating coaching decisions.

Other than motorized sports, ice hockey is the fastest-paced sport where the typical speed of professional players can reach up to 25 mp/h, and the puck up to 100 mp/h. Detecting, localizing, recognizing, and tracking players moving at this speed from videos becomes a highly challenging task.

This work presents a player tracking and identification system based on deep neural networks for fast-paced sports and showcases its application to ice hockey. We first train Faster R-CNN object

detector S. Ren, He, Girshick, and Sun (2015) to detect the persons in each video frame. Since the players of the same team will have the same appearance, we detect the region of the jersey number on the back of the player uniforms using a scene text recognition model J. Baek et al. (2019) and fine-tune the CRAFT text detector Y. Baek, Lee, Han, Yun, and Lee (2019) to detect the jersey number of each player. Tracking multiple players is challenging due to the similar appearance of the players within the same team, the occlusions and complicated motion of the players that make the problem even more complex. Tracking is performed using the Neural Solver Mot Brasó and Leal-Taixé (2020), a visual multi-object tracking framework. Neural Solver Mot can track multiple objects in video sequences based on rudimentary data association and state estimation techniques.

To summarize, our contributions are (i) a complete framework for player tracking and identification for the fastest-paced sport of ice hockey that achieves 80.2 score for Multiple Object Tracking Accuracy (MOTA) on real data, and (ii) a practical method of transfer learning and fine-tuning a text detection model Y. Baek et al. (2019) on player jersey numbers that obtains 83.76% accuracy.

7.3 Related Work

In this section, we provide information about the datasets and discuss related work grouped according to the tasks: player detection, player tracking, and jersey number recognition.

7.3.1 Dataset

State-of-the-art techniques for player tracking such as Vats, Walters, Fani, Clausi, and Zelek (2021) and Chan, Levine, and Javan (2021) uses broadcast National Hockey League (NHL) videos. However, there is no public benchmark dataset that additionally provides team and player identification. In this chapter, we use the McGill Hockey Player Tracking Dataset (MHPTD) Yingnan Zhao (2020) that consists of NHL broadcasting videos captured by the main camera. The videos contain only *tracklet* annotations, which define the tracking information for each player. We extend it with annotations of jersey number labels for player identification.

7.3.2 Player detection

The foundation of the majority of player detection algorithms has been the Viola-Jones Object Detection Framework Viola and Jones (2001) and Histograms of Oriented Gradients for Human Detection (HOG) Dalal and Triggs (2005) and relied on the segmentation and recognition of players. Most of the shortcomings of these early attempts at human detection Okuma, Taleghani, Freitas, Little, and Lowe (2004); Šaric, Dujmic, Papic, and Rožic (2008) that relied on hand-crafted features were eradicated with modern deep learning-based approaches. The advent of deep neural networks began with AlexNet Krizhevsky et al. (2012), which won the Imagenet Large Scale Visual Recognition Challenge (ILSVRC) Russakovsky et al. (2012). Due to the continuous improvement and advances in hardware and convolutional neural network methodologies, many new robust techniques have been proposed to handle the challenges in sports videos. Nowadays, object detection is largely based on deep neural networks such as YOLO Redmon, Divvala, Girshick, and Farhadi (2016), or part-based approaches Senocak, Oh, Kim, and So Kweon (2018) and result in higher performance than the traditional methods in terms of missed, false, duplicate detections, and unreliable detection boundary.

Chan et al. proposed a residual network (ResNet) He, Zhang, Ren, and Sun (2016) as the CNN base with recurrent long short-term memory (LSTM) Hochreiter and Schmidhuber (1997) for identifying the players. Vats et al. Vats et al. (2021) presented a temporal 1D CNN without any other dedicated networks for processing temporal information. Perhaps the most successful and popular solution to object detection was presented in a line of works starting with Region-based Convolutional Neural Networks (R-CNN) Girshick, Donahue, Darrell, and Malik (2014), Fast R-CNN Girshick (2015), and Faster R-CNN S. Ren et al. (2015); which are considered to be state-of-the-art deep learning visual object detection algorithms. In our work, we employ Faster R-CNN S. Ren et al. (2015) scheme because of its fast convergence when generating detection proposals.

7.3.3 Player Tracking

SORT Bewley, Ge, Ott, Ramos, and Upcroft (2016), and Deep SORT Wojke, Bewley, and Paulus (2017) are widely used tracking-by-detection multi-object tracking frameworks, which track

multiple objects in video sequences based on rudimentary data association and state estimation techniques. The trackers have different metrics by which they compare detections:

- Features. SIFT D. G. Lowe (2004), SURF Bay et al. (2006) and ORB Rublee et al. (2011) are the most popular descriptors for feature extraction and matching in object tracking methods. ORB is faster in terms of feature extraction and tolerance to the image rotation and noise; thus, it is widely used in tracking, mapping, and relocalization.

- Kalman filter. Kalman filter Kalman (1960) is used for tracking moving objects and estimates an object's velocity and acceleration by measuring its locations. Kalman filter is primarily used for associating detections with trajectories Bewley et al. (2016); Wojke et al. (2017); Zhang, Wang, Wang, Zeng, and Liu (2021). Chen et al. X. Chen, Wang, and Xuan (2018) track multiple moving objects with occlusion using unscented Kalman filtering techniques.

- Person re-identification (ReID). ReID is the task of identifying people across different images. Ice hockey player tracking methods such as Cai, Freitas, and Little (2006) apply hand-crafted features for the detection and re-identification. Ahmed et al. Ahmed, Jones, and Marks (2015) propose a method of learning features and corresponding similarity metrics for person re-identification. The network outputs either a similarity score between the images or classification of the images as the same in the case that the images depict the same person.

Neural Solver Mot Brasó and Leal-Taixé (2020) is a recent work that jointly learns features over the global graph of the entire set of detections and predicts final solutions. Our tracking framework employs the Neural Solver Mot Brasó and Leal-Taixé (2020) and applies the ReID metrics instead of face recognition or number detection because, in sports games, faces or jersey numbers are not always visible to the primary camera.

7.3.4 Number Recognition

Similar to hand-writing recognition Graves and Schmidhuber (2008); Pratt, Ochoa, Yadav, Sheta, and Eldefrawy (2019), jersey number recognition algorithms can be categorized into the following two groups: (a) Optical Character Recognition (OCR) based methods and (b) Convolution Neural Networks (CNN) based methods.

OCR-based methods such as Lu et al. (2013); Messelodi and Modena (2012) employ hand-crafted features to localize the text or number regions on the player uniform and then pass the segmented regions to the OCR module for recognition of the text or number.

On the other hand, CNN-based models Gerke, Muller, and Schafer (2015); Lyu, Liao, Yao, Wu, and Bai (2018); Shi, Bai, and Yao (2016) notably improved the performance of number recognition compared to OCR-based methods. However, the scope of these methods is limited to the training set. Detection of jersey numbers usually follows a traditional localization step and recognition step. When character cells are detected, recognition proceeds as a two-pass process: recognizing each word and resolving fuzzy spaces. Erroneous recognition occurs in classes (i.e. jersey numbers) that share at least one digit. Digit-wise approaches proposed by Li et al. Gerke et al. (2015); G. Li, Xu, Liu, Li, and Wang (2018) improve on this problem by fusing with the spatial transformer network (STN). Jersey numbers recognition is challenging due to player poses and view-point variations, CRAFT Y. Baek et al. (2019) is a scene text detection method showing promising results in challenging scenes with arbitrarily-oriented, curved, or deformed texts.

7.4 System Overview

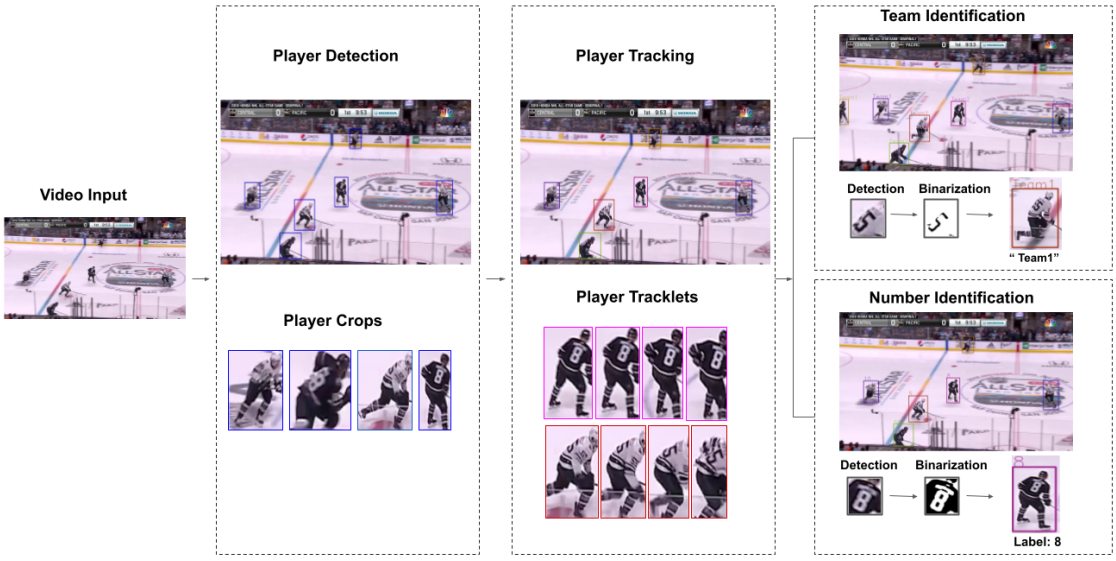


Figure 7.2: Overview of player tracking and identification system.

Our proposed solution to player identification is a pipeline involving four steps. First, we detect

the players using the region proposals returned by Faster R-CNN and track them throughout the sequence of images using similarity metrics for person re-identification. The result is a set of tracklets describing the motion path of each player in the image sequence. Next, for each player tracklet, we perform team identification using the dominant colour from the detected patch and the player identification using the jersey number recognized. Figure 7.2 shows the pipeline of our system. We explain each component in more detail in the following sections.

7.4.1 Player detection

The first step of the pipeline is player detection. Given a video of an ice hockey game, it is converted into a sequence of images. Next, we perform person detection as an initial step since the players are instances of the person class. All subsequent processing is performed only on the person-detected regions.

YOLO Redmon et al. (2016) and Faster R-CNN S. Ren et al. (2015) are two of the most widely used object detection frameworks providing reliable results. We employ Faster R-CNN Inception-V2-COCO Model for person detection because of its superior performance. The model was trained on the Common Objects in Context (COCO) Lin et al. (2014) dataset on 91 categories of objects. Given an image, object detection models such as Faster R-CNN first generate potential bounding boxes using region-based detectors comprising convolutional feature maps. Then, they apply a Region Proposal Network (RPN) classifier, which simultaneously regresses region bounds and objectness scores at each location on the proposed regions/bounding boxes. RPNs are designed to predict region proposals with various scales and aspect ratios efficiently. After predicting the region, post-processing is used to refine the bounding boxes, eliminate duplicate detections, and re-score the bounding boxes based on other objects in the scene. The loss of Faster-RCNN is given by,

$$L(\{\mathcal{P}_i\},\{b_i\}) = \frac{1}{S_c}\sum_i L_c(\mathcal{P}_i, Gp_i) + w \times \frac{1}{S_r}\sum_i \mathcal{P}_i L_r(B_i, Gb_i) \tag{16}$$

where i is the index of an anchor in a batch and $\mathcal{P}_i$ is the probability of anchor i being an object. b_i is a vector representing the coordinates of the predicted bounding box. S_c and S_r are the normalization mini-batch size of classification and regression, respectively. L_c and L_r are the

classification and regression loss, respectively. The ground-truth label Gp_i is 1 if the anchor is positive and is 0 if the anchor is negative. B_i is a vector representing the coordinates bounding box, and Gb_i is that of the ground-truth box. The two terms are weighted by a balancing parameter w.

7.4.2 Player Tracking

With the players detected, the second step is to track individual players. We apply Neural Solver Mot Brasó and Leal-Taixé (2020) fine-tuned on the ice hockey dataset as the tracker to create the player tracklets. As expected, the accuracy of the detection has a substantial effect on the accuracy of the tracker. Since achieving perfect detection is exceptionally difficult, we account for detection failures using person re-identification (ReID) metrics. Using external ReID datasets is a common practice among Multiple Object Tracking (MOT) methods. The network is pre-trained for the task of ReIdentification (ReID) on three publicly available datasets, namely Market1501 Zheng et al. (2015), CUHK03 W. Li, Zhao, Xiao, and Wang (2014) and DukeMTMC Ristani, Solera, Zou, Cucchiara, and Tomasi (2016).

Algorithm 3 gives the pseudocode for player tracking. The input is a set of player detections $P = p_1, \ldots, p_n$, where n is the total number of objects for all frames of a video. Each detection is represented by $p_i = (a_i, c_i, t_i)$, where a_i denotes the raw pixels of the bounding box, c_i contains its 2D image coordinates and t_i its timestamp. A tracklet is then defined as a set of time-ordered object detections $T_i = p_{i_1}, \ldots, p_{i_{n_i}}$, where n_i is the number of detections that form trajectory i. The goal of MOT is to find the set of tracklets $T' = T_1, \ldots, T_m$, that best explains the observations O. The problem can be modelled as an undirected graph $G = (V, E)$, where $V = 1, \ldots, n, E \subset V \times V$, and each node $i \in V$ represents a unique detection $p_i \in O$. The set of edges E is constructed so that every pair of detections, i.e., nodes, in different frames is connected, hence allowing to recover tracklets with missed detections.

7.4.3 Player Identification

Players are identified by their jersey numbers. Recognizing jersey numbers is challenging because the jerseys are deformable objects and can appear somewhat distorted in the image. Moreover, there is considerable variation in the players' posture and camera view angles, which significantly

Algorithm 3: Algorithm for Player Tracking

Input : Player Detections $P = \{p_1, ..., p_n\}$
Multiple-Object Tracking (MOT) Graph $G = (V, E)$,
Fractional solution $\hat{F}$
Output: set of $tracklets$ T' = $\{T_1, \ldots, T_m\}$ $e = 0$ for all e in $G = (V, E)$ #Initialization
for $p_i \in P$ **do**
    node v represents p_i, $v \in V$
    if p_i *has the same trajectory* T & *is temporally consecutive* **then**
        $e = 1$ in $G = (V, E)$
    else
        $e = 0$ in $G = (V, E)$
for $\{(e_1, e_2), ..., (e_{n_{i-1}}, e_{n_i})\} \in E$ **do**
    if $\hat{F}_(e_{n_{i-1}}, e_{n_i}) \geq \tau_\theta$ **then**
        T_i =1, $T_i \in \{T_1, ..., T_m\}$
                                                #Evaluation
    else
        T_i =0

affects the appearance of jersey numbers in terms of projected area and perceived font.

Number/text detection

After extracting the tracklets for each player, we detect the jersey number. For each image in a player's tracklet, we apply a scene text detection method Y. Baek et al. (2019) to localize the jersey number region. The model architecture has a backbone network VGG-16 Simonyan and Zisserman (2014) and is supervised to localize character regions and link the regions in a bottom-up manner. Using the CRAFT Y. Baek et al. (2019) pre-trained model, we detect texts of various horizontal, curved and arbitrary-oriented shapes. The model outputs 2-channel score maps: the region score with the location of every character and an affinity score for linking characters to instances. The loss function L is defined as follows:

$$L = \sum_i S_r(i) - S'_r(i)_2^2 + \sum_i S_a(i) - S'_a(i)_2^2 \tag{17}$$

where $S'_r(i)$ and $S'_a(i)$ indicate region score and affinity map of the ground truth respectively, and $S_r(i)$ and $S_a(i)$ indicate the predicted region score and affinity score, respectively. We further

improve the robustness by extending it with a post-processing step that filters out text instances that are unlikely to be a jersey number based on the aspect ratio of the detected region.

Number identification

All number-detected regions are further processed for (i) team identification and (ii) number identification to identify each player's tracklets.

Team identification.

In some circumstances, the recognized jersey number of players might be the same. The goal of the team identification step is to binarize the input patches and separate the patches where the dominant colour is white (dark foreground colour on a bright background) from the patches where a dominant colour is black (bright foreground colour on a dark background); the two groups corresponding to the two teams. Additionally, if a team roster is available, we eliminate false detections and recognitions by removing detected jersey numbers that do not exist.

Number identification.

Number identification is essentially a text recognition task. We use the pre-trained model for TPS-ResNet-BiLSTM-Atten J. Baek et al. (2019) text recognition. The model can recognize the number as a whole and therefore overcomes the difficulties arising with multiple-digit jersey numbers captured from non-frontal views with distortions. The model is a four-stage scene text recognition framework which contains the transformation and employs the thin-plate spline (TPS), visual feature extraction, sequence modelling of Bidirectional LSTM (BiLSTM) and predictions of the character sequence. There are two types of implementations in the method of Baek et al. Y. Baek et al. (2019): Connectionist Temporal Classification (CTC) and Attention mechanism (Attn).

In Connectionist Temporal Classification (CTC), the conditional probability is computed by summing the probabilities that are mapped onto the label sequence, as in equation 18:

$$\mathcal{P}(S_l|S_i) = \sum_{\pi:M(\pi)=S_l} \mathcal{P}(\pi|S_i) \tag{18}$$

where S_l is the label sequence, S_i is input sequence and $\mathcal{P}(\pi|H)$ is the probability of observing either a character or a blank at a point in time, and M is the mapping of π onto S_l. Using Attention

mechanism (Attn), the output O_t at time step t is predicted using an LSTM attention decoder as follows,

$$O_t = softmax(W_0 h_t + p_0) \qquad h_t = LSTM(O_{t-1}, c_t, h_{t-1}) \tag{19}$$

where W_0, p_0 are the trainable parameters, c_t is a context vector, and h_t, h_{t-1} represent the decoder LSTM hidden states at time steps t and $t-1$, respectively.

7.5 Results

7.5.1 Dataset

The most relevant state-of-the-art methods to our tracking system are Vats et al. (2021) and Chan et al. (2021), however direct comparison is not possible because the authors do not make their datasets available. Therefore, we report on the MHPTD dataset Yingnan Zhao (2020), a publicly available dataset that consists of 25 NHL gameplay video clips of resolution 1280 $\times$ 720 pixels. Each clip contains one shot of the gameplay from the overhead camera position and comprises a series of frames that run for an uninterrupted period of time without a cut scene or camera switch. The clips provided have mixed frame rates, in two popular NHL broadcast video frame rates available on the market, 60 and 30 frames per second. We further extend the ground truth tracking information provided in MHPTD with manually labelled tracking IDs with the jersey number and team label to facilitate the evaluation of the player tracking and identification.

7.5.2 Player Tracking

Player detection is performed using a Faster-RCNN network S. Ren et al. (2015) pre-trained on the COCO dataset Lin et al. (2014). We compared two different object detectors for player detection, the detector presented in Faster-RCNN S. Ren et al. (2015) and the one in YOLO_v3 Redmon et al. (2016). A comparison of the player detection results is shown in Figure 7.3a and 7.3b. As shown, there can be erroneous detections due to misclassification, occlusions and the audience of the game,

Method	MOTA↑	IDF1↑	MT↑	FP↓	FN↓
SORT Bewley et al. (2016)	55.1	76.3	404	615	1296
Deep SORT Wojke et al. (2017)	56.3	77.1	435	487	968
MOT Neural Solver Brasó and Leal-Taixé (2020)	64.5	80.2	656	422	917

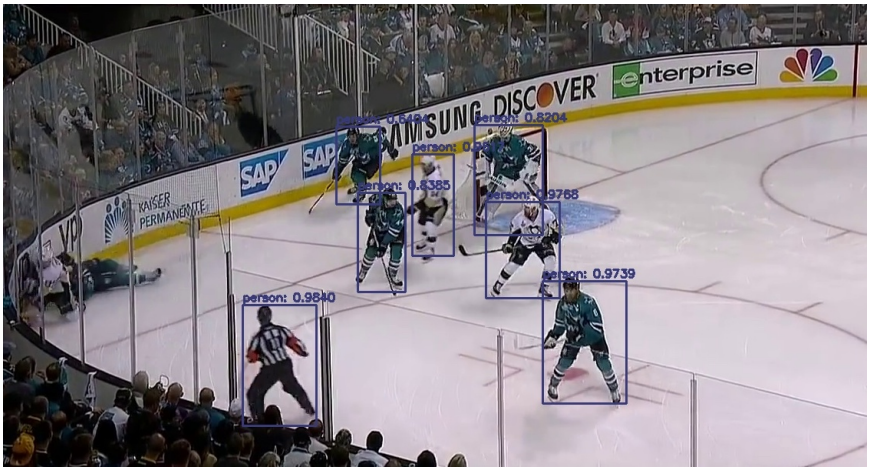
Table 7.1: Comparison of different approaches for multiple object tracking in a video clip.

most of them can be filtered with the length of $tracklets$ and patch size. An example of $tracklet$, a sequence of images of a tracked player, is shown in Figure 7.3c with YOLO_v3 in the top row, and Faster-RCNN in the bottom row. The Faster-RCNN model detects a more complete region of the players. The object detector of Faster-RCNN obtains an average precision (AP) of 66.8 on the test videos, the YOLO_v3 obtains an average precision (AP) of 53.32.

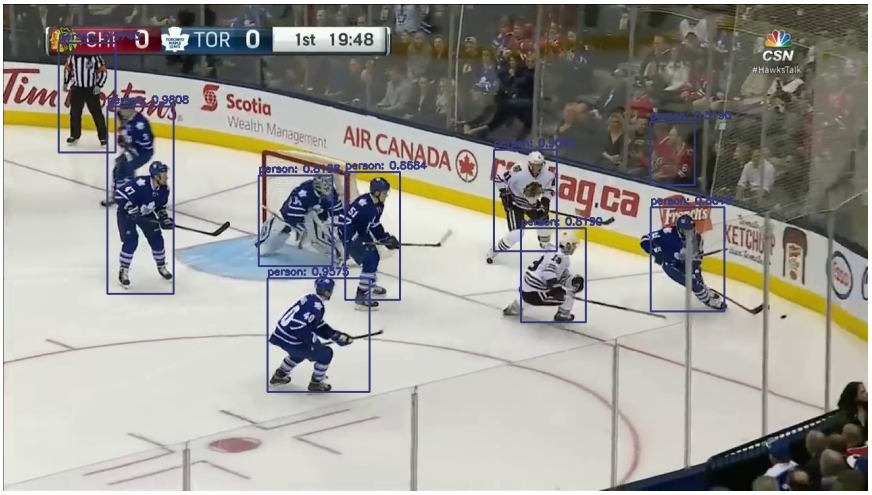
$$MOTA = 1 - \frac{\sum_i (FN_i + FP_i + AE_i)}{\sum_i GT_i} \tag{20}$$

We experimented with three state-of-the-art tracking algorithms on the hockey player dataset. The Multiple Object Tracking Accuracy (MOTA) Kasturi et al. (2009) and IDF1 Score (IDF1) Ristani et al. (2016) are the most important evaluation metrics in multiple object tracking. MOTA is calculated with the formula in Equation 20, where i is the frame index, GT is the ground truth, FP the false positives, FN the false negatives, and association errors (AE) count for the fraction. IDF1 is the fraction of correctly identified detections over the average number of true and computed detections. The trajectory coverage is represented by Mostly Tracked (MT) trajectories.

As shown in Table 7.1, the best tracking performance is achieved using the MOT Neural Solver tracking model Brasó and Leal-Taixé (2020) with the person re-identification (reID) re-trained on the hockey dataset. The reported average was 56.3 for MOTA and 60.67 for IDF1 from the original MOT Neural resolver Brasó and Leal-Taixé (2020). Based on our experiments, the MOT Neural resolver obtains the highest average MOTA score of 64.5 and IDF1 score of 80.2 on the test videos.



(a)



(b)

(c)

Figure 7.3: (a),(b) Examples of player detection results in two frames from video clips. (c) A visual comparison of output player tracklets using YOLO_v3 model (top-row) and Faster-RCNN model (bottom-row).

7.5.3 Player identification

Text detection

We use the pretrained weights of the CRAFT detector and adapt it to the ice hockey dataset by fine-tuning on 500 images in our dataset. For the fine-tuning, the model is trained for 30 epochs at a learning rate of $3.2e-5$. During training, we apply image augmentations, namely, affine transformation, Gaussian blur and colour channels manipulation to both the original player image and the corresponding number of bounding boxes. The other subsets are used for testing and validation. An example of text detection is shown in Figure 7.4a. The fine-tuning of the pre-trained model to the ice hockey dataset results in better region detection of the jersey numbers. Some failed detections are shown in Figure 7.4b, typically occurring when there are complex backgrounds such as banner advertisements which may contain text, player collisions and occlusions, low contrast, and contours resulting from stripes and other logos on the players' jerseys.

Team Identification

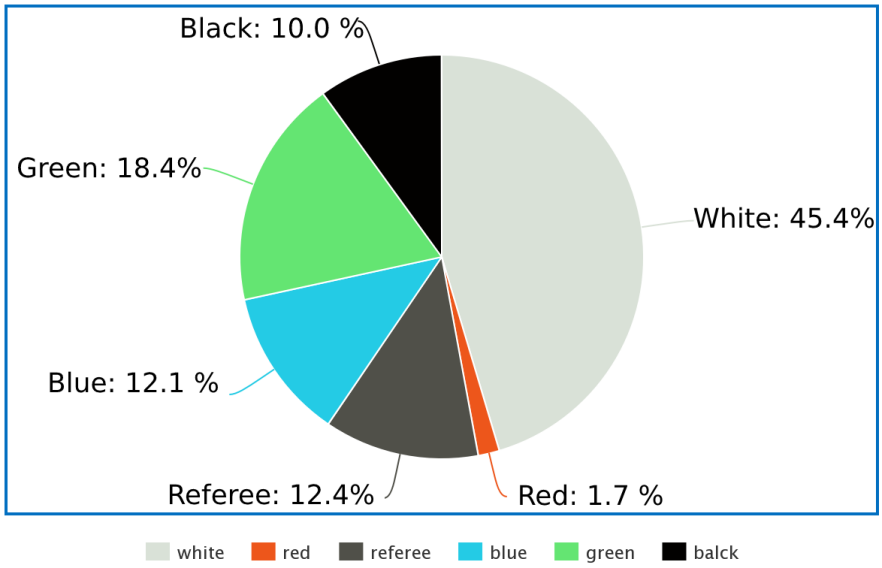
Given the detected text region for each player, we binarize the image and convert it to black and white. Next, we detect the dominant patch colour and determine whether it is white text on a dark background or black text on a bright background. Thus, we split the players into two groups corresponding to the home team and the guest team.

According to the MHPTD dataset Yingnan Zhao (2020), the player jersey number distribution in the Figure 7.4c, shows that almost half of the players wear white-coloured jerseys. Similar proportions of players are wearing blue, green and black sweaters. Our method achieves an accuracy of 78.36% in the team classification. Most of the errors are attributed to the referees being considered a player, and colours of lower contrast leading to false detections.

Jersey Number Identification

For jersey number recognition, we employ the pretrained model TPS-ResNet-BiLSTM-Atten for text recognition J. Baek et al. (2019). The attention mechanism handles challenging cases such as jersey numbers sharing at least one digit, variations of player pose, and changes in the camera

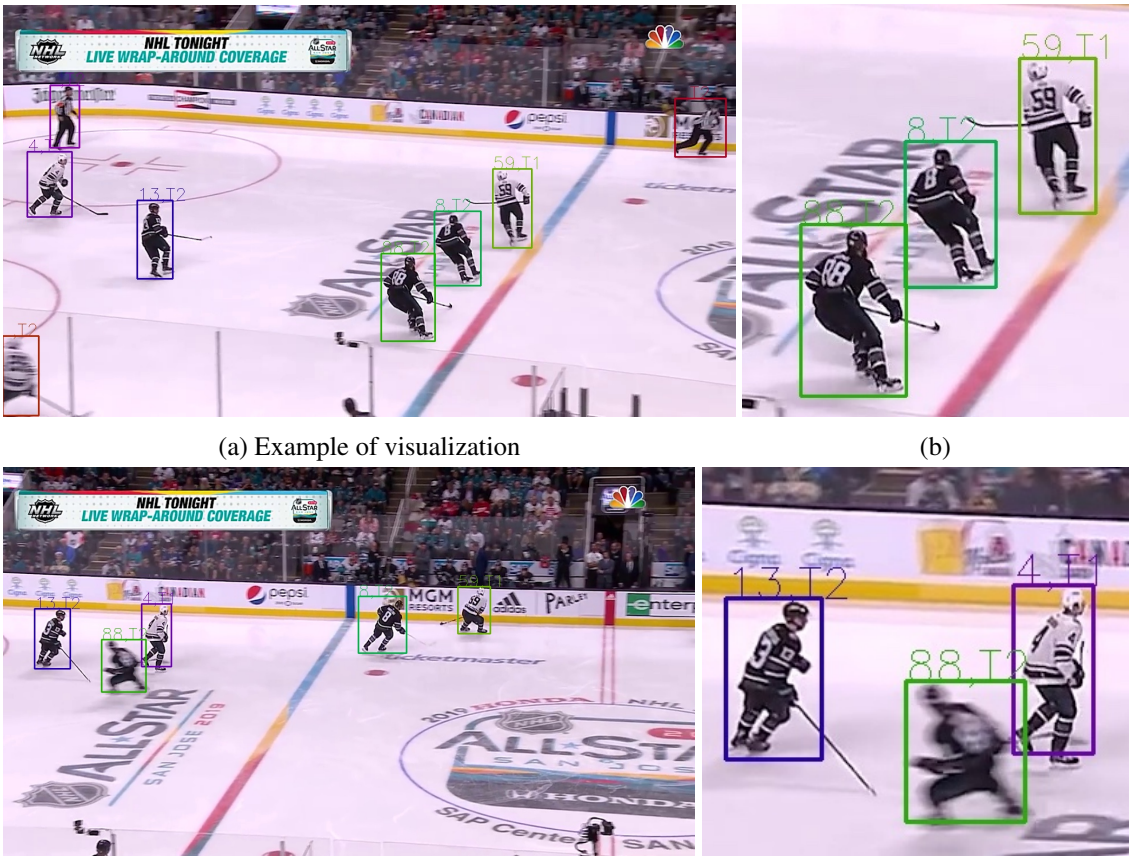
(a) (b)



(c)

Figure 7.4: (a) Comparison of Text Detection without (top row) or with (bottom row) fine-tuning. (b) Some failed detections, typically occurring when there are complex backgrounds such as banner advertisements which may contain text, player collisions and occlusions, low contrast, and contours resulting from stripes and other logos on the players' jerseys. (c) Analysis of player jersey color distribution in the dataset MHPTD Yingnan Zhao (2020).

viewpoint. The result from tracking contains 462 player tracklets from 15,194 player images, the jersey number bounding box annotation and a per player class. For each player's tracklet, we assign the detected jersey number label with the highest votes. The accuracy of jersey number identification is 83.76%. The team identification and player jersey number identification are visualized in the input video, as shown in the example in Figure 7.5. If a player is tracked, a random coloured bounding box is drawn, and the jersey number label and the identified team are annotated above the box. Since player tracking is not using the jersey number information, some non-recognizable poses in the same tracklet will be handled by assigning the jersey number label from the other frames of the same tracklet.



(a) Example of visualization (b)

(c) Example of motion blur (d)

Figure 7.5: Visualization of the output of the tracking system. If a player is tracked, a random coloured bounding box is drawn, and the jersey number label and the identified team are annotated above the box. (b) and (d) is the close-up view of (a) and (c) respectively.

7.6 Conclusion

We presented a complete framework for player tracking and identification in ice hockey that exploits the high performance of deep learning neural networks. The framework consists of three main components, namely, player detection, player tracking and player identification. We extended the tracking information publicly available dataset Yingnan Zhao (2020) with jersey number and team information and report on experiments. Using our system the average precision (AP) for player detection is 64.5, the Multiple Object Tracking Accuracy (MOTA) for player tracking is 80.2, and the accuracies for team identification and player number identification are 78.36% and 83.76%, respectively. Our framework can simultaneously track multiple players in fast-paced sports such as ice hockey and has comparable performance with state-of-the-art player tracking and identification systems.

Chapter 8

Conclusion

The work in this thesis considers a complete pipeline for an image-based 3D reconstruction system. The presented contributions build upon fundamental image-based 3D reconstruction techniques and are applied to real-world applications. The aim of this thesis is to improve state-of-the-art systems in terms of algorithmic efficiency, generalization, robustness, and accuracy and present the entire 3D reconstruction pipeline. Towards this goal, we presented improved algorithms, new approaches, and their applications for the different stages of a 3D reconstruction system. The majority of the contributions presented adhere to the common idea of fully automatic and robust integration of classic theory and improvements from novel methods, and derive a deep understanding of the underlying problems. In this thesis, all presented contributions are demonstrated through experimental evaluations. Experimental evaluation was a key part of both coming up with new ideas and proving that the proposed methods worked. Experiments on several datasets, including the benchmarks, demonstrated the reliability of the proposed approaches. As part of the thesis, the developed methods and algorithms have been combined into a single-step 3D dense reconstruction system. The evaluation in this thesis and other recent 3D reconstruction benchmarks show that both the individual parts and the system as a whole improve the state of the art in terms of robustness, efficiency, and accuracy.

In Chapter 1, we introduced the background of the 3D reconstruction topic in computer vision and discussed the motivations, typical pipelines, and challenges. In Chapter 2, we introduced the

related basic principles in computer vision—multi-view geometry, deep learning, and optimization—that are pre-requisites to understanding the thesis. Next, in Chapter 3, we explored and reviewed the existing work that is related to our 3D reconstruction system. Chapter 4 presents the paper "Motion Estimation for Large Displacements and Deformations" Q. Chen and Poullis (2022b), which is the first step in the 3D reconstruction pipeline to find dense correspondences between pairs of images. Traditional image-based modeling techniques follow a two-step process: first sparse reconstruction, then dense reconstruction. In Chapter 5 of this thesis, we present the paper "Single-shot Dense Reconstruction with Epic-flow" Q. Chen and Poullis (2018). The proposed single-shot process employs dense flow fields to recover correspondences, which reduces the computational complexity but improves the scalability. Nowadays, the prevalence of deep learning techniques may set the standard for precision and efficacy. Chapter 6 is based on the work in the paper "End-to-End Multi-View Structure-from-Motion" Q. Chen and Poullis (2022a) and presents the proposed deep learning approach to 3D reconstruction. In Chapter 7, we present a real-world application of motion flow to the detection, reidentification, and tracking of ice hockey players.

8.1 Future Work

In this thesis, we have made a step-change to state-of-the-art image-based 3D modeling. However, many remaining problems and constraints stand in the way of accomplishing the ambitious objective of a fully general-purpose image-based 3D modeling system that is capable of producing highly accurate and photorealistic reconstructions from any sensor and any scenario. This chapter discusses some of the underlying challenges that still exist as well as potential future solutions to address these issues.

8.1.1 Efficiency and Scalability

Further work is needed to keep up with the ever-increasing amount and scale of images. Even though this thesis and other recent work suggest a number of ways to speed things up, more optimizations and efficient algorithms are still needed. This challenge can be alleviated by hardware advancements and algorithmic improvements. For the system to work better, more research needs

to be done on multi-threading, parallel programming, and other algorithmic optimizations. Deep learning methods also take a long time to train the model, but the time it takes to test the model is much shorter. Therefore, future work using deep learning techniques must focus on reducing the training time and requiring less training data.

8.1.2 Procedural vs. Learning

Traditionally, all of the components of an image-based 3D modeling system have been hand-crafted using our understanding of geometry, optimization, and the physics of image formation. Over the last several decades, more and more of these hand-crafted techniques have been replaced by automatic learning algorithms, supported by remarkable advancements in machine learning, particularly deep learning. In this thesis, we showed and tested an end-to-end Structure-from-Motion (SfM) network as a replacement for traditional methods to make the image-based 3D modeling pipeline more efficient, reliable, and accurate. The proposed deep-learning approach follows the same pipeline as our proposed single-shot dense reconstruction pipeline, which performs optical flow estimation for dense correspondences, then reconstructs the scene by applying SfM. In the future, it will be interesting to see which other parts of the pipeline can be replaced by deep learning components and, in principle, whether it is possible to build a fully learned reconstruction pipeline. Furthermore, image-based 3D modeling is an open question as to whether a fully learned system will be superior, and how much capacity and memory are needed to create an effective model in a fully automatic system through deep learning. In our future work, we will be focused on automatically learning algorithms and tightly integrating them with existing knowledge of multi-view geometry reconstruction and optimization.

References

Achanta, R., Shaji, A., Smith, K., Lucchi, A., Fua, P., & Süsstrunk, S. (2012). Slic superpixels compared to state-of-the-art superpixel methods. *IEEE transactions on pattern analysis and machine intelligence*, *34*(11), 2274–2282.

Agarwal, S., Mierle, K., & Team, T. C. S. (2022, 3). *Ceres Solver.* Retrieved from https://github.com/ceres-solver/ceres-solver

Agarwal, S., Snavely, N., Seitz, S. M., & Szeliski, R. (2010). Bundle adjustment in the large. In *European conference on computer vision* (pp. 29–42).

Ahmed, E., Jones, M., & Marks, T. K. (2015). An improved deep learning architecture for person re-identification. In *Proceedings of the ieee conference on computer vision and pattern recognition* (pp. 3908–3916).

Anandan, P. (1989). A computational framework and an algorithm for the measurement of visual motion. *International Journal of Computer Vision*, *2*(3), 283–310.

Arandjelovic, R., & Zisserman, A. (2012, June). Three things everyone should know to improve object retrieval. In *Computer vision and pattern recognition, conference on* (pp. 2911–2918). Providence, Rhode Island. Retrieved from http://dx.doi.org/10.1109/CVPR.2012.6248018

Asari, V. K. (2013). *Wide area surveillance: Real-time motion detection systems* (Vol. vol.6). Springer.

Baek, J., Kim, G., Lee, J., Park, S., Han, D., Yun, S., ... Lee, H. (2019). What is wrong with scene text recognition model comparisons? dataset and model analysis. In *Proceedings of the ieee/cvf international conference on computer vision* (pp. 4715–4723).

Baek, Y., Lee, B., Han, D., Yun, S., & Lee, H. (2019). Character region awareness for text detection. In *Proceedings of the ieee/cvf conference on computer vision and pattern recognition* (pp. 9365–9374).

Bailer, C., Varanasi, K., & Stricker, D. (2017). Cnn-based patch matching for optical flow with thresholded hinge embedding loss. In *Proceedings of the ieee conference on computer vision and pattern recognition* (pp. 3250–3259).

Baker, S., Scharstein, D., Lewis, J., Roth, S., Black, M. J., & Szeliski, R. (2011). A database and evaluation methodology for optical flow. *International journal of computer vision*, *92*(1), 1–31.

Balntas, V., Riba, E., Ponsa, D., & Mikolajczyk, K. (2016). Learning local feature descriptors with triplets and shallow convolutional neural networks. In *Bmvc* (Vol. 1, p. 3).

Bar-Haim, A., & Wolf, L. (2020). Scopeflow: Dynamic scene scoping for optical flow. In *Proceedings of the ieee/cvf cvpr* (pp. 7998–8007).

Barnes, C., Goldman, D. B., Shechtman, E., & Finkelstein, A. (2011). The patchmatch randomized matching algorithm for image manipulation. *Communications of the ACM*, *54*(11), 103–110.

Bay, H., Tuytelaars, T., & Van Gool, L. (2006). Surf: Speeded up robust features. In *European conference on computer vision* (pp. 404–417).

Bewley, A., Ge, Z., Ott, L., Ramos, F., & Upcroft, B. (2016). Simple online and realtime tracking. In *2016 ieee international conference on image processing (icip)* (pp. 3464–3468).

Blasch, E., & Seetharaman. (2014). Summary of methods in wide-area motion imagery (wami). In *Geospatial infofusion and video analytics iv; and motion imagery for isr and situational awareness ii* (Vol. 9089, p. 90890C).

Brasó, G., & Leal-Taixé, L. (2020). Learning a neural solver for multiple object tracking. In *Proceedings of the ieee/cvf conference on computer vision and pattern recognition* (pp. 6247–6257).

Brox, T., Bruhn, A., Papenberg, N., & Weickert, J. (2004). High accuracy optical flow estimation based on a theory for warping. In *European conference on computer vision* (pp. 25–36).

Bursuc, A., Tolias, G., & Jégou, H. (2015). Kernel local descriptors with implicit rotation matching.

In *Proceedings of the 5th acm on international conference on multimedia retrieval* (pp. 595–598).

Butler, D. J., Wulff, J., Stanley, G. B., & Black, M. J. (2012). A naturalistic open source movie for optical flow evaluation. In *European conference on computer vision* (pp. 611–625).

Cai, Y., Freitas, N. d., & Little, J. J. (2006). Robust visual tracking for multiple targets. In *European conference on computer vision* (pp. 107–118).

Canny, J. (1986). A computational approach to edge detection. *IEEE Transactions on pattern analysis and machine intelligence*(6), 679–698.

Chan, A., Levine, M. D., & Javan, M. (2021). Player identification in hockey broadcast videos. *Expert Systems with Applications*, *165*, 113891.

Chen, L.-C., Papandreou, G., Kokkinos, I., Murphy, K., & Yuille, A. L. (2017). Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs. *IEEE transactions on pattern analysis and machine intelligence*, *40*(4), 834–848.

Chen, Q., & Poullis, C. (2018). Single-shot dense reconstruction with epic-flow. In *2018 - 3dtv-conference: The true vision - capture, transmission and display of 3d video (3dtv-con)* (p. 1-4). doi: 10.1109/3DTV.2018.8478620

Chen, Q., & Poullis, C. (2022a). End-to-end multi-view structure-from-motion with hypercorrelation volumes. In *2022 - ieee international conference on signal processing, sensors, and intelligent systems (spsis).*

Chen, Q., & Poullis, C. (2022b). Motion estimation for large displacements and deformations. *Nature Scientific Reports*. doi: SciRep12,19721(2022).https://doi.org/10.1038/s41598-022-21987-7

Chen, R., Han, S., Xu, J., & Su, H. (2019). Point-based multi-view stereo network. In *Proceedings of the ieee/cvf international conference on computer vision* (pp. 1538–1547).

Chen, X., Wang, X., & Xuan, J. (2018). Tracking multiple moving objects using unscented kalman filtering techniques. *arXiv preprint arXiv:1802.01235*.

Cho, M., Lee, J., & Lee, K. M. (2010). Reweighted random walks for graph matching. In *European conference on computer vision* (pp. 492–505).

Collins, R. T. (1996). A space-sweep approach to true multi-image matching. In *Proceedings cvpr*

ieee computer society conference on computer vision and pattern recognition (pp. 358–363).

Dalal, N., & Triggs, B. (2005). Histograms of oriented gradients for human detection. In *2005 ieee computer society conference on computer vision and pattern recognition (cvpr'05)* (Vol. 1, pp. 886–893).

Deng, J., Dong, W., Socher, R., Li, L.-J., Li, K., & Fei-Fei, L. (2009). Imagenet: A large-scale hierarchical image database. In *2009 ieee conference on computer vision and pattern recognition* (pp. 248–255).

Dokeroglu, T., Sevinc, E., & Cosar, A. (2019). Artificial bee colony optimization for the quadratic assignment problem. *Applied soft computing*, *76*, 595–606.

Dong, J., & Soatto, S. (2015). Domain-size pooling in local descriptors: Dsp-sift. In *Proceedings of the ieee conference on computer vision and pattern recognition* (pp. 5097–5106).

Dosovitskiy, A., & Fischer. (2015). Flownet: Learning optical flow with convolutional networks. In *Proceedings of the ieee international conference on computer vision* (pp. 2758–2766).

Dosovitskiy, A., Fischer, P., Ilg, E., Hausser, P., Hazirbas, C., Golkov, V., ... Brox, T. (2015). Flownet: Learning optical flow with convolutional networks. In *Proceedings of the ieee international conference on computer vision* (pp. 2758–2766).

Ecker, A., & Ullman, S. (2009). A hierarchical non-parametric method for capturing non-rigid deformations. *Image and Vision Computing*, *27*(1-2), 87–98.

Edelberg, J., & Daniel. (2012). Autonomous cross-correlation of optical mti for live inspection and tracking. In *Geospatial infofusion ii* (Vol. 8396, p. 839609).

Felzenszwalb, P. F., & Huttenlocher, D. P. (2004). Efficient graph-based image segmentation. *International Journal of Computer Vision*, *59*(2), 167-181.

Furukawa, Y., Curless, B., Seitz, S. M., & Szeliski, R. (2010). Towards internet-scale multi-view stereo. In *2010 ieee computer society conference on computer vision and pattern recognition* (pp. 1434–1441).

Furukawa, Y., & Ponce, J. (2007). Accurate, dense, and robust multi-view stereopsis (pmvs). In *Ieee computer society conference on computer vision and pattern recognition* (Vol. 2).

Gadot, D., & Wolf, L. (2016). Patchbatch: A batch augmented loss for optical flow. In *Proceedings of the ieee conference on computer vision and pattern recognition* (pp. 4236–4245).

Gerke, S., Muller, K., & Schafer, R. (2015). Soccer jersey number recognition using convolutional neural networks. In *Proceedings of the ieee international conference on computer vision workshops* (pp. 17–24).

Gherardi, R., Farenzena, M., & Fusiello, A. (2010). Improving the efficiency of hierarchical structure-and-motion. In *2010 ieee computer society conference on computer vision and pattern recognition* (pp. 1594–1600).

Gibson, J. J. (1950). The perception of the visual world. *Houghton Mifflin*.

Girshick, R. (2015). Fast r-cnn. In *Proceedings of the ieee international conference on computer vision* (pp. 1440–1448).

Girshick, R., Donahue, J., Darrell, T., & Malik, J. (2014). Rich feature hierarchies for accurate object detection and semantic segmentation. In *Proceedings of the ieee conference on computer vision and pattern recognition* (pp. 580–587).

Graves, A., & Schmidhuber, J. (2008). Offline handwriting recognition with multidimensional recurrent neural networks. *Advances in neural information processing systems*, *21*.

Groenert, M., & Bryski, D. (2009). Airborne infrared persistent imaging requirements. In *Adaptive coded aperture imaging, non-imaging, and unconventional imaging sensor systems* (Vol. 7468, p. 746802).

Hackbusch, W. (1994). *Iterative solution of large sparse systems of equations* (Vol. 95). Springer.

Harris, C., Stephens, M., et al. (1988). A combined corner and edge detector. In *Alvey vision conference* (Vol. 15, pp. 10–5244).

Hartmann, W., Galliani, S., Havlena, M., Van Gool, L., & Schindler, K. (2017). Learned multi-patch similarity. In *Proceedings of the ieee international conference on computer vision* (pp. 1586–1594).

He, K., Zhang, X., Ren, S., & Sun, J. (2015). Deep residual learning for image recognition. *CoRR*, *abs/1512.03385*. Retrieved from http://arxiv.org/abs/1512.03385

He, K., Zhang, X., Ren, S., & Sun, J. (2016). Deep residual learning for image recognition. In *Proceedings of the ieee conference on computer vision and pattern recognition* (pp. 770–778).

Hochreiter, S., & Schmidhuber, J. (1997). Long short-term memory. *Neural computation*, *9*(8),

1735–1780.

Hoffer, E., & Ailon, N. (2015). Deep metric learning using triplet network. In *International workshop on similarity-based pattern recognition* (pp. 84–92).

Horaud, R., Hansard, M., Evangelidis, G., & Ménier, C. (2016). An overview of depth cameras and range scanners based on time-of-flight technologies. *Machine vision and applications*, *27*(7), 1005–1020.

Horn, B. K., & Schunck, B. G. (1981). Determining optical flow. *Artificial intelligence*, *17*(1-3), 185–203.

Hu, Y., Li, Y., & Song, R. (2017). Robust interpolation of correspondences for large displacement optical flow. In *Proceedings of the ieee conference on computer vision and pattern recognition* (pp. 481–489).

Hu, Y., Song, R., & Li, Y. (2016). Efficient coarse-to-fine patchmatch for large displacement optical flow. In *Proceedings of the ieee conference on computer vision and pattern recognition* (pp. 5704–5712).

Hur, J., & Roth, S. (2019). Iterative residual refinement for joint optical flow and occlusion estimation. In *Proceedings of the ieee cvpr* (pp. 5754–5763).

Ilg, E., Mayer, N., Saikia, T., Keuper, M., Dosovitskiy, A., & Brox, T. (2017). Flownet 2.0: Evolution of optical flow estimation with deep networks. In *Proceedings of the ieee conference on computer vision and pattern recognition* (pp. 2462–2470).

Ilg, E., Mayer, N., Saikia, T., Keuper, M., Dosovitskiy, A., & Brox, T. (2017). Flownet 2.0: Evolution of optical flow estimation with deep networks. In *2017 ieee conference on computer vision and pattern recognition (cvpr)* (p. 1647-1655). doi: 10.1109/CVPR.2017.179

Jensen, R., Dahl, A., Vogiatzis, G., Tola, E., & Aanæs, H. (2014). Large scale multi-view stereopsis evaluation. In *2014 ieee conference on computer vision and pattern recognition* (pp. 406–413).

Ji, M., Gall, J., Zheng, H., Liu, Y., & Fang, L. (2017). Surfacenet: An end-to-end 3d neural network for multiview stereopsis. In *Proceedings of the ieee international conference on computer vision* (pp. 2307–2315).

Jiang, S., Campbell, D., Lu, Y., li, H., & Hartley, R. (2021, 04). Learning to estimate hidden

motions with global motion aggregation. In *The international conference on computer vision (iccv)*.

Kalman, R. E. (1960). A new approach to linear filtering and prediction problems.

Kant, S. (2012). Activity-based exploitation of Full Motion Video (FMV). In D. Self (Ed.), *Full motion video (fmv) workflows and technologies for intelligence, surveillance, and reconnaissance (isr) and situational awareness* (Vol. 8386, p. 83860D). SPIE. Retrieved from https://doi.org/10.1117/12.920280 doi: 10.1117/12.920280

Kasturi, R., Goldgof, D., Soundararajan, P., Manohar, V., Garofolo, J., Bowers, R., ... Zhang, J. (2009). Framework for performance evaluation of face, text, and vehicle detection and tracking in video: Data, metrics, and protocol. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *31*(2), 319-336. doi: 10.1109/TPAMI.2008.57

Kazhdan, M., Bolitho, M., & Hoppe, H. (2006). Poisson surface reconstruction. In *Proceedings of the fourth eurographics symposium on geometry processing* (Vol. 7).

Ke, Y., & Sukthankar, R. (2004). Pca-sift: a more distinctive representation for local image descriptors. In *Proceedings of the 2004 ieee computer society conference on computer vision and pattern recognition, 2004. cvpr 2004.* (Vol. 2, p. II-II). doi: 10.1109/CVPR.2004.1315206

Kendall, A., Martirosyan, H., Dasgupta, S., Henry, P., Kennedy, R., Bachrach, A., & Bry, A. (2017). End-to-end learning of geometry and context for deep stereo regression. In *Proceedings of the ieee international conference on computer vision* (pp. 66–75).

Keysers, D., Deselaers, T., Gollan, C., & Ney, H. (2007). Deformation models for image recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *29*(8), 1422–1435.

Krizhevsky, A., Sutskever, I., & Hinton, G. E. (2012). Imagenet classification with deep convolutional neural networks. In F. Pereira, C. Burges, L. Bottou, & K. Weinberger (Eds.), *Advances in neural information processing systems 25* (pp. 1097–1105). Curran Associates, Inc. Retrieved from http://papers.nips.cc/paper/4824-imagenet-classification-with-deep-convolutional-neural-networks.pdf

Krizhevsky, A., Sutskever, I., & Hinton, G. E. (2017). Imagenet classification with deep convolutional neural networks. *Communications of the ACM*, *60*(6), 84–90.

Kushal, A., & Agarwal, S. (2012). Visibility based preconditioning for bundle adjustment. In *2012*

ieee conference on computer vision and pattern recognition (pp. 1442–1449).

Lee, J. Y., DeGol, J., Zou, C., & Hoiem, D. (2021). Patchmatch-rl: Deep mvs with pixelwise depth, normal, and visibility. In *Proceedings of the ieee/cvf international conference on computer vision* (pp. 6158–6167).

Lempitsky, V., Rother, C., Roth, S., & Blake, A. (2009). Fusion moves for markov random field optimization. *IEEE transactions on pattern analysis and machine intelligence*, *32*(8), 1392–1405.

Lepetit, V., & Fua, P. (2006). Keypoint recognition using randomized trees. *IEEE transactions on pattern analysis and machine intelligence*, *28*(9), 1465–1479.

Li, G., Xu, S., Liu, X., Li, L., & Wang, C. (2018). Jersey number recognition with semi-supervised spatial transformer network. In *Proceedings of the ieee conference on computer vision and pattern recognition workshops* (pp. 1783–1790).

Li, W., Zhao, R., Xiao, T., & Wang, X. (2014). Deepreid: Deep filter pairing neural network for person re-identification. In *Proceedings of the ieee conference on computer vision and pattern recognition* (pp. 152–159).

Li, Y., Hu, Y., Song, R., Rao, P., & Wang, Y. (2017). Coarse-to-fine patchmatch for dense correspondence. *IEEE Transactions on Circuits and Systems for Video Technology*, *28*(9), 2233–2245.

Lin, T.-Y., Maire, M., Belongie, S., Hays, J., Perona, P., Ramanan, D., ... Zitnick, C. L. (2014). Microsoft coco: Common objects in context. In *European conference on computer vision* (pp. 740–755).

Liu, C., Yuen, J., & Torralba, A. (2010). Sift flow: Dense correspondence across scenes and its applications. *IEEE transactions on pattern analysis and machine intelligence*, *33*(5), 978–994.

Liu, P., Lyu, M., King, I., & Xu, J. (2019). Selflow: Self-supervised learning of optical flow. In *Proceedings of the ieee cvpr* (pp. 4571–4580).

Locher, A., Perdoch, M., & Van Gool, L. (2016). Progressive prioritized multi-view stereo. In *Proceedings of the ieee conference on computer vision and pattern recognition* (pp. 3244–3252).

Lowe, D. G. (2004). Distinctive image features from scale-invariant keypoints. *International*

journal of computer vision, *60*(2), 91–110.

Lowe, D. G., et al. (1999). Object recognition from local scale-invariant features. In *iccv* (Vol. 99, pp. 1150–1157).

Lowe, G. (2004). Sift-the scale invariant feature transform. *International Journal of Computer Vision 60(2)*, *2*(91-110), 2.

Lu, C.-W., Lin, C.-Y., Hsu, C.-Y., Weng, M.-F., Kang, L.-W., & Liao, H.-Y. M. (2013). Identification and tracking of players in sport videos. In *Proceedings of the fifth international conference on internet multimedia computing and service* (pp. 113–116).

Lucas, B. D., Kanade, T., et al. (1981). *An iterative image registration technique with an application to stereo vision* (Vol. 81). Vancouver.

Luo, C., Yang, Z., Wang, P., Wang, Y., Xu, W., Nevatia, R., & Yuille, A. (2019). Every pixel counts++: Joint learning of geometry and motion with 3d holistic understanding. *IEEE TPAMI*, *42*(10), 2624–2641.

Lyu, P., Liao, M., Yao, C., Wu, W., & Bai, X. (2018). Mask textspotter: An end-to-end trainable neural network for spotting text with arbitrary shapes. In *Proceedings of the european conference on computer vision (eccv)* (pp. 67–83).

Martinec, D., & Pajdla, T. (2007). Robust rotation and translation estimation in multiview reconstruction. In *2007 ieee conference on computer vision and pattern recognition* (pp. 1–8).

Maurer, D., Marniok, N., Goldluecke, B., & Bruhn, A. (2018). Structure-from-motion-aware patchmatch for adaptive optical flow estimation. In *Proceedings of the european conference on computer vision (eccv)* (pp. 565–581).

Mayer, N., Ilg, E., Hausser, P., Fischer, P., Cremers, D., Dosovitskiy, A., & Brox, T. (2016). A large dataset to train convolutional networks for disparity, optical flow, and scene flow estimation. In *Proceedings of the ieee conference on computer vision and pattern recognition* (pp. 4040–4048).

Meister, S., Hur, J., & Roth, S. (2018, February). UnFlow: Unsupervised learning of optical flow with a bidirectional census loss. In *Aaai.* New Orleans, Louisiana.

Menze, M., & Geiger, A. (2015). Object scene flow for autonomous vehicles. In *Conference on computer vision and pattern recognition (cvpr).*

Menze, M., Heipke, C., & Geiger, A. (2015). Joint 3d estimation of vehicles and scene flow. In *Isprs workshop on image sequence analysis (isa).*

Messelodi, S., & Modena, C. (2012, 01). Scene text recognition and tracking to identify athletes in sport videos. *Multimedia Tools and Applications*, *63*, 1-25. doi: 10.1007/s11042-011-0878-y

Muja, M., & Lowe, D. G. (2009). Fast approximate nearest neighbors with automatic algorithm configuration. *VISAPP (1)*, *2*(331-340), 2.

Murray, D. W., & Buxton, B. F. (1987). Scene segmentation from visual motion using global optimization. *IEEE transactions on pattern analysis and machine intelligence*(2), 220–228.

Nistér, D. (2004). An efficient solution to the five-point relative pose problem. *IEEE transactions on pattern analysis and machine intelligence*, *26*(6), 756–770.

Nister, D., & Stewenius, H. (2006). Scalable recognition with a vocabulary tree. In *2006 ieee computer society conference on computer vision and pattern recognition* (Vol. 2, pp. 2161–2168).

Okuma, K., Taleghani, A., Freitas, N. d., Little, J. J., & Lowe, D. G. (2004). A boosted particle filter: Multitarget detection and tracking. In *European conference on computer vision* (pp. 28–39).

Pelapur, R., Candemir, S., Bunyak, F., Poostchi, M., Seetharaman, G., & Palaniappan, K. (2012). Persistent target tracking using likelihood fusion in wide-area and full motion video sequences. In *2012 15th international conference on information fusion* (pp. 2420–2427).

Philbin, J., Isard, M., Sivic, J., & Zisserman, A. (2010). Descriptor learning for efficient retrieval. In *European conference on computer vision* (pp. 677–691).

Piergiovanni, A., & Ryoo, M. S. (2019). Representation flow for action recognition. In *Proceedings of the ieee cvpr* (pp. 9945–9953).

Porter, R., Fraser, A. M., & Hush, D. (2010). Wide-area motion imagery. *IEEE Signal Processing Magazine*, *27*(5), 56–65.

Porzi, L., Hofinger, M., Ruiz, I., Serrat, J., Bulo, S. R., & Kontschieder, P. (2020). Learning multi-object tracking and segmentation from automatic annotations. In *Proceedings of the ieee/cvf cvpr* (pp. 6846–6855).

Pratt, S., Ochoa, A., Yadav, M., Sheta, A., & Eldefrawy, M. (2019). Handwritten digits recognition

using convolution neural networks. *Journal of Computing Sciences in Colleges*, *34*(5), 40–46.

Ranjan, A., & Black, M. J. (2017). Optical flow estimation using a spatial pyramid network. In *2017 ieee cvpr* (p. 2720-2729). doi: 10.1109/CVPR.2017.291

Redmon, J., Divvala, S., Girshick, R., & Farhadi, A. (2016). You only look once: Unified, real-time object detection. In *Proceedings of the ieee conference on computer vision and pattern recognition* (pp. 779–788).

Ren, S., He, K., Girshick, R., & Sun, J. (2015). Faster r-cnn: Towards real-time object detection with region proposal networks. *Advances in neural information processing systems*, *28*.

Ren, Z., Yan, J., Ni, B., Liu, B., Yang, X., & Zha, H. (2017). Unsupervised deep learning for optical flow estimation. In *Thirty-first aaai conference on artificial intelligence.*

Revaud, J., Weinzaepfel, P., Harchaoui, Z., & Schmid, C. (2015). Epicflow: Edge-preserving interpolation of correspondences for optical flow. In *Proceedings of the ieee cvpr* (pp. 1164–1172).

Revaud, J., Weinzaepfel, P., Harchaoui, Z., & Schmid, C. (2016). Deepmatching: Hierarchical deformable dense matching. *International Journal of Computer Vision*, *120*(3), 300–323.

Ristani, E., Solera, F., Zou, R., Cucchiara, R., & Tomasi, C. (2016). Performance measures and a data set for multi-target, multi-camera tracking. In *European conference on computer vision* (pp. 17–35).

Rodríguez, A. L., López-de Teruel, P. E., & Ruiz, A. (2011). Reduced epipolar cost for accelerated incremental sfm. In *Cvpr 2011* (pp. 3097–3104).

Roth, S., Lempitsky, V., & Rother, C. (2009). Discrete-continuous optimization for optical flow estimation. In *Statistical and geometrical approaches to visual motion analysis* (pp. 1–22). Springer.

Rublee, E., Rabaud, V., Konolige, K., & Bradski, G. (2011). Orb: An efficient alternative to sift or surf. In *2011 international conference on computer vision* (pp. 2564–2571).

Russakovsky, O., Deng, J., Su, H., Krause, J., Satheesh, S., Ma, S., ... Fei-Fei, L. (2012). ImageNet Large Scale Visual Recognition Challenge. *International Journal of Computer Vision (IJCV)*.

Šaric, M., Dujmic, H., Papic, V., & Rožic, N. (2008). Player number localization and recognition in soccer video using hsv color space and internal contours. *International Journal of Electrical*

and Computer Engineering, *2*(7), 1408–1412.

Schonberger, J. L., & Frahm, J.-M. (2016). Structure-from-motion revisited. In *Proceedings of the ieee conference on computer vision and pattern recognition* (pp. 4104–4113).

Schönberger, J. L., Zheng, E., Frahm, J.-M., & Pollefeys, M. (2016). Pixelwise view selection for unstructured multi-view stereo. In *European conference on computer vision* (pp. 501–518).

Seitz, S. M., Curless, B., Diebel, J., Scharstein, D., & Szeliski, R. (2006). A comparison and evaluation of multi-view stereo reconstruction algorithms. In *2006 ieee computer society conference on computer vision and pattern recognition (cvpr'06)* (Vol. 1, pp. 519–528).

Senocak, A., Oh, T.-H., Kim, J., & So Kweon, I. (2018). Part-based player identification using deep convolutional representation and multi-scale pooling. In *Proceedings of the ieee conference on computer vision and pattern recognition workshops* (pp. 1732–1739).

Shi, B., Bai, X., & Yao, C. (2016). An end-to-end trainable neural network for image-based sequence recognition and its application to scene text recognition. *IEEE transactions on pattern analysis and machine intelligence*, *39*(11), 2298–2304.

Simonyan, K., Vedaldi, A., & Zisserman, A. (2014). Learning local feature descriptors using convex optimisation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *36*(8), 1573–1585.

Simonyan, K., & Zisserman, A. (2014). Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*.

Simo-Serra, E., Trulls, E., Ferraz, L., Kokkinos, I., Fua, P., & Moreno-Noguer, F. (2015). Discriminative learning of deep convolutional feature point descriptors. In *Proceedings of the ieee international conference on computer vision* (pp. 118–126).

Snavely, N., Seitz, S. M., & Szeliski, R. (2006). Photo tourism: Exploring photo collections in 3d. In *Siggraph conference proceedings* (pp. 835–846). New York, NY, USA: ACM Press.

Snavely, N., Seitz, S. M., & Szeliski, R. (2008). Modeling the world from internet photo collections. *International journal of computer vision*, *80*(2), 189–210.

Sun, D., Roth, S., Lewis, J. P., & Black, M. J. (2008). Learning optical flow. In *European conference on computer vision* (pp. 83–97).

Sun, D., Yang, X., Liu, M., & Kautz, J. (2018). Pwc-net: Cnns for optical flow using pyramid, warping, and cost volume. In *2018 ieee/cvf conference on computer vision and pattern recognition* (p. 8934-8943). doi: 10.1109/CVPR.2018.00931

Tang, C., & Tan, P. (2018). Ba-net: Dense bundle adjustment network. *arXiv preprint arXiv:1806.04807*.

Teed, Z., & Deng, J. (2018). Deepv2d: Video to depth with differentiable structure from motion. *arXiv preprint arXiv:1812.04605*.

Teed, Z., & Deng, J. (2021, 8). Raft: Recurrent all-pairs field transforms for optical flow (extended abstract). In Z.-H. Zhou (Ed.), *Proceedings of the thirtieth international joint conference on artificial intelligence, IJCAI-21* (pp. 4839–4843). International Joint Conferences on Artificial Intelligence Organization. (Sister Conferences Best Papers)

Tian, Y., Yu, X., Fan, B., Wu, F., Heijnen, H., & Balntas, V. (2019). Sosnet: Second order similarity regularization for local descriptor learning. In *Proceedings of the ieee conference on computer vision and pattern recognition* (pp. 11016–11025).

Tola, E., Lepetit, V., & Fua, P. (2009). Daisy: An efficient dense descriptor applied to wide-baseline stereo. *IEEE transactions on pattern analysis and machine intelligence*, *32*(5), 815–830.

Tuytelaars, T., Mikolajczyk, K., et al. (2008). Local invariant feature detectors: a survey. *Foundations and trends® in computer graphics and vision*, *3*(3), 177–280.

Ummenhofer, B., Zhou, H., Uhrig, J., Mayer, N., Ilg, E., Dosovitskiy, A., & Brox, T. (2017). Demon: Depth and motion network for learning monocular stereo. In *Proceedings of the ieee conference on computer vision and pattern recognition* (pp. 5038–5047).

Vats, K., Walters, P., Fani, M., Clausi, D. A., & Zelek, J. (2021). Player tracking and identification in ice hockey. *arXiv preprint arXiv:2110.03090*.

Viola, P., & Jones, M. (2001). Rapid object detection using a boosted cascade of simple features. In *Proceedings of the 2001 ieee computer society conference on computer vision and pattern recognition. cvpr 2001* (Vol. 1, pp. I–I).

Viswanathan, D. G. (2009). Features from accelerated segment test (fast). In *Proceedings of the 10th workshop on image analysis for multimedia interactive services, london, uk* (pp. 6–8).

Wang, J., Song, Y., Leung, T., Rosenberg, C., Wang, J., Philbin, J., ... Wu, Y. (2014). Learning

fine-grained image similarity with deep ranking. In *Proceedings of the ieee conference on computer vision and pattern recognition* (pp. 1386–1393).

Wang, J., Zhong, Y., Dai, Y., Birchfield, S., Zhang, K., Smolyanskiy, N., & Li, H. (2021). Deep two-view structure-from-motion revisited. In *Proceedings of the ieee/cvf conference on computer vision and pattern recognition* (pp. 8953–8962).

Wang, Y., Wang, P., Yang, Z., Luo, C., Yang, Y., & Xu, W. (2019). Unos: Unified unsupervised optical-flow and stereo-depth estimation by watching videos. In *Proceedings of the ieee conference on computer vision and pattern recognition* (pp. 8071–8081).

Wedel, A., Pock, T., Braun, J., Franke, U., & Cremers, D. (2008). Duality tv-l1 flow with fundamental matrix prior. In *2008 23rd international conference image and vision computing new zealand* (pp. 1–6).

Wedel, A., Pock, T., Zach, C., Bischof, H., & Cremers, D. (2009). An improved algorithm for tv-l 1 optical flow. In *Statistical and geometrical approaches to visual motion analysis* (pp. 23–45). Springer.

Wei, X., Zhang, Y., Li, Z., Fu, Y., & Xue, X. (2020). Deepsfm: Structure from motion via deep bundle adjustment. In *European conference on computer vision* (pp. 230–247).

Wills, J., Agarwal, S., & Belongie, S. (2006). A feature-based approach for dense segmentation and estimation of large disparity motion. *International Journal of Computer Vision*, *68*(2), 125–143.

Wojke, N., Bewley, A., & Paulus, D. (2017). Simple online and realtime tracking with a deep association metric. In *2017 ieee international conference on image processing (icip)* (pp. 3645–3649).

Wu, C. (2013). Towards linear-time incremental structure from motion. In *2013 international conference on 3d vision-3dv 2013* (pp. 127–134).

Wu, C., Agarwal, S., Curless, B., & Seitz, S. M. (2011). Multicore bundle adjustment. In *Cvpr 2011* (pp. 3057–3064).

Wu, R., Chen, Y., Blasch, E., Liu, B., Chen, G., & Shen, D. (2014). A container-based elastic cloud architecture for real-time full-motion video (fmv) target tracking. In *2014 ieee applied imagery pattern recognition workshop (aipr)* (pp. 1–8).

Yang, J., Mao, W., Alvarez, J. M., & Liu, M. (2020). Cost volume pyramid based depth inference for multi-view stereo. In *Proceedings of the ieee/cvf conference on computer vision and pattern recognition* (pp. 4877–4886).

Yao, Y., Luo, Z., Li, S., Fang, T., & Quan, L. (2018). Mvsnet: Depth inference for unstructured multi-view stereo. In *Proceedings of the european conference on computer vision (eccv)* (pp. 767–783).

Yao, Y., Luo, Z., Li, S., Shen, T., Fang, T., & Quan, L. (2019). Recurrent mvsnet for high-resolution multi-view stereo depth inference. *Computer Vision and Pattern Recognition (CVPR)*.

Yi, K. M., Trulls, E., Lepetit, V., & Fua, P. (2016). Lift: Learned invariant feature transform. In *European conference on computer vision* (pp. 467–483).

Yin, Z., & Shi, J. (2018). Geonet: Unsupervised learning of dense depth, optical flow and camera pose. In *Proceedings of the ieee conference on computer vision and pattern recognition* (pp. 1983–1992).

Yingnan Zhao, K. C., Zihui Li. (2020). A method for tracking hockey players by exploiting multiple detections and omni-scale appearance features. *Project Report*.

Yu, J., & Ramamoorthi, R. (2020). Learning video stabilization using optical flow. In *Proceedings of the ieee/cvf cvpr* (pp. 8159–8167).

Zbontar, J., LeCun, Y., et al. (2016). Stereo matching by training a convolutional neural network to compare image patches. *J. Mach. Learn. Res.*, *17*(1), 2287–2318.

Zhang, Y., Wang, C., Wang, X., Zeng, W., & Liu, W. (2021). Fairmot: On the fairness of detection and re-identification in multiple object tracking. *International Journal of Computer Vision*, *129*(11), 3069–3087.

Zheng, L., Shen, L., Tian, L., Wang, S., Wang, J., & Tian, Q. (2015). Scalable person re-identification: A benchmark. In *Proceedings of the ieee international conference on computer vision* (pp. 1116–1124).

Zhou, F., & De la Torre, F. (2012). Factorized graph matching. In *2012 ieee conference on computer vision and pattern recognition* (pp. 127–134).

Zhou, F., & De la Torre, F. (2013). Deformable graph matching. In *2013 ieee conference on computer vision and pattern recognition* (pp. 2922–2929).

Zhou, T., Brown, M., Snavely, N., & Lowe, D. G. (2017). Unsupervised learning of depth and ego-motion from video. In *Proceedings of the ieee conference on computer vision and pattern recognition* (pp. 1851–1858).