Optimization techniques for image registration applied to remote sensing

Bruno Conejo

Université Paris-Est
Laboratoire d'Informatique Gaspard-Monge
UMR 8049

A thesis submitted for the degree of
Doctor of Philosophy

Optimization techniques for image registration applied to remote sensing

Bruno Conejo

supervised by
Prof. Pascal Monasse

February 3, 2018

Abstract

This thesis studies the computer vision problem of image registration in the context of geological remote sensing surveys. More precisely we dispose in this work of two images picturing the same geographical scene but acquired from two different view points and possibly at a different time. The task of registration is to associate to each pixel of the first image its counterpart in the second image.

While this problem is relatively easy for human-beings, it remains an open problem to solve it with a computer. Numerous approaches to address this task have been proposed. The most promising techniques formulate the task as a numerical optimization problem. Unfortunately, the number of unknowns along with the nature of the objective function make the optimization problem extremely difficult to solve. This thesis investigates two approaches along with a coarsening scheme to solve the underlying numerical problem.

Each approach makes use of a different assumption to simplify the original problem. The convex approach is computationally faster while the non-convex approach delivers more precise solutions. On top of both approaches, we investigate coarsening frameworks to speed-up the computations.

In our context, we study the First order Primal-Dual techniques for convex optimization. After a progressive introduction of underlying mathematics, we study the dual optimal solution space of the TV-regularized problems. We prove a new theorem that greatly facilitates the demonstration of previously established theorems. We also provide a new algorithm to optimize the famous ROF-model.

As a second approach, we survey the graph-cuts techniques. We also investigate different mincut-maxflow solvers since they are an essential building block of the graph-cuts techniques. We propose a new implementation of the celebrated Fast-PD solver that drastically outperform the initial implementation provided by original authors.

We also study coarsening methods for both optimization approaches. We experiment with image and energy pyramid coarsening scheme for graph-cut techniques. In this context we propose a novel framework that drastically speeds-up the inference run-time while maintaining remarkable accuracy.

Finally, we experiment with different remote sensing problems to demonstrate the versatility and efficiency of our approaches. Especially, we investigate the computation of depth maps from stereo-images acquired from aerial and space surveys. Using LiDAR acquisitions we also propose an algorithm to automatically infer the damages due to earthquakes and soil liquefaction. Finally, we also monitor the ground deformation induced by earthquake using realistic simulated model.

Résumé

Dans le contexte de la vision par ordinateur cette thèse étudie le problème d'appariement d'images dans le cadre de la télédétection pour la géologie. Plus précisément, nous disposons dans ce travail de deux images de la même scène géographique, mais acquises à partir de deux points de vue différents et éventuellement à un autre moment. La tâche d'appariement est d'associer à chaque pixel de la première image un pixel de la seconde image.

Bien que ce problème soit relativement facile pour les êtres humains, il reste difficile à résoudre par un ordinateur. De nombreuses approches pour traiter cette tâche ont été proposées. Les techniques les plus prometteuses formulent la tâche comme un problème d'optimisation numérique. Malheureusement, le nombre d'inconnues ainsi que la nature de la fonction à optimiser rendent ce problème extrêmement difficile à résoudre. Cette thèse étudie deux approches avec un schéma multi-échelle pour résoudre le problème numérique sous-jacent.

Chaque approche utilise une hypothèse différente pour simplifier le problème initial. L'approche convexe est plus rapide, tandis que l'approche non convexe offre des solutions plus précises. En plus des deux approches, nous étudions les schéma multi-échelle pour accélérer les calculs.

Dans notre contexte, nous étudions les techniques Primal-Dual de première ordre pour l'optimisation convexe. Après une introduction progressive des mathématiques sous-jacentes, nous étudions l'espace dual de solution optimale des problèmes régularisés par a priori *tv*. Nous prouvons un nouveau théorème qui facilite grandement la démonstration d'autres théorèmes précédemment établis. Nous proposons également un nouvel algorithme pour optimiser le célèbre modèle ROF.

Pour la seconde seconde approche nous examinons les techniques de *graph-cut*. Nous étudions également différents algorithmes de *mincut-maxflow* car ils constituent un élément essentiel des techniques de *graph-cut*. Nous proposons une nouvelle implémentation du célèbre algorithme *Fast-PD* qui améliore drastiquement les performances.

Nous étudions également les méthodes multi-échelles pour les deux approches d'optimisation. Nous expérimentons les schémas de pyramide d'image et d'énergie pour les techniques *graph-cut*. Dans ce contexte, nous proposons une nouvelle approche qui accélère considérablement le temps d'exécution de l'inférence tout en conservant remarquablement la précision des solutions.

Enfin, nous étudions différents problèmes de télédétection pour démontrer la polyvalence et l'efficacité de nos approches. En particulier, nous étudions le calcul des cartes de profondeur à partir d'images stéréo aériennes et spatiales. En utilisant les acquisitions de LiDAR, nous proposons également un algorithme pour déduire automatiquement les dommages causés par les séismes et la liquéfaction

des sols. Enfin, nous étudions également la déformation du sol induite par tremblement de terre en utilisant une simulation sismique réaliste.

Acknowledgment

To the most important person in my life, Louise Naud, I would to say thank you for bearing with me during this journey. I would like to thank my parents for their unconditional support and affection.

I would like to acknowledge, Pascal Monasse, Jean-Philippe Avouac, Sebastien Leprince, Francois Ayoub and Nikos Komodakis for their guidance and numerous pieces of advice. A special thank you to Heather Steele, Lisa Christiansen, Brigitte Mondou and Sylvie Cash.

I would like to thank Arwen Bradley, Hugues Talbot, Francois Malgouyres and Andrés Almansa for their time and deep review of my work. Finally, I would also like to thank people from the GPS division at Caltech and from the image lab of LIGM.

Contents

Chapitre 1

Introduction partielle (en français)

Nous débutons ce chapitre en introduisant le contexte de nos travaux 2.1. Nous expliquons comment la télédétection est utilisée dans les études géologiques. En particulier, nous expliquons le lien avec la vision par ordinateur et plus particulièrement l'appariement d'images. Les travaux précédents sont présentés dans la section 2.2. Nous introduisons dans la section 2.3 le model mathématique utilisé dans ce document. Finalement, dans la section 2.4 nous présentons un résumé du contenu technique de cette thèse ainsi que nos contributions scientifiques.

1.1 Contexte

1.1.1 Vue du ciel

Le siècle dernier a connu un nombre croissant de techniques et de dispositifs pour observer la Terre. Un développement majeur est l'utilisation de satellites et d'avions équipés de capteur d'imagerie pour observer la Terre vue du ciel. Avec le développement des technologie spatiales, les satellites ont largement contribué à l'étude de la Terre et d'autres planètes telles que Mars. Nous disposons maintenant de nombreuses images haute résolution de multiple planètes et de leurs satellites naturels.

On peut classer les capteurs d'observation en deux familles principales. Les capteurs actifs tels que le LiDAR et le Radar enregistrent le reflet du signal qu'ils ont émis. Au contraire, les capteurs passifs tels que les caméras panchromatiques, couleurs ou hyper-spectrales enregistrent directement le signal émis par la scène observée.

Observing sensors Dans ce travail, nous étudions des acquisitions à partir de caméras panchromatiques et de capteurs LiDAR.

Photogrammetry La photogrammétrie est l'ensemble des techniques qui utilisent la photographie pour mesurer les distances entre les objets. De 1850 à 1900, le goniographe était l'outil de référence pour dessiner des cartes comme illustré par la figure 1.1. La photogrammétrie analogique utilisée de 1900 à 1960 s'appuie sur le concept de vision stéréo-métrique. Cependant, un opérateur exécutait toujours la tâche essentielle d'appariement comme illustré dans la figure 1.2. À partir de 1960, la disponibilité des ordinateurs a progressivement diminué la nécessité d'une implication humaine. À partir de 2000, la photogrammétrie moderne repose entièrement sur des données numérisées et requiert très peu d'intervention humaine. Pour plus de détails, nous proposons au lecteur curieux les travaux de [73], [66], [115] et [80] pour une analyse historique, et [76] pour les fondements mathématiques.

FIGURE 1.1 – Operateur manipulant un goniographe. *Propriété NOAA.*

FIGURE 1.2 – Operateur dessinant une carte en utilisant les techniques de photogrammétrie analogique. *Propriété WSP group.*

LiDAR La NASA a investi pendant les années 1970 dans le développement de techniques modernes de télédétection à base de laser. Cette étude initiale visait principalement à mesurer les propriétés de l'atmosphère et de l'océan, la canopée forestière ou les couches de glace. Dans les années 1990, l'accessibilité aux dispositifs de positionnement tels que GPS, Global Positioning System et les centrales inertielles, permet d'utiliser le LiDAR à des fins de cartographie. Enfin, dans les années 2000, les logiciels de traitement et la diminution du coût des infrastructures informatiques a fait du LiDAR un outil de cartographie précis et efficace, comme illustré dans les figures 1.3 and 1.4. Pour plus de details, nous conseillons la lecture de [144], [5], [11] et [31] pour une revue des technologies LiDAR et ses applications, mais également [6] pour une introduction des fondations mathématiques.

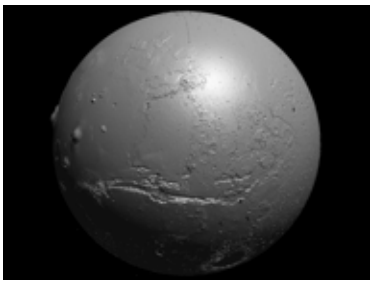


FIGURE 1.3 – Acquisition LiDAR de Mars, *Propriété NASA*.

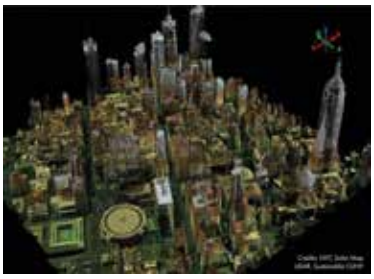


FIGURE 1.4 – Nuage de points 3d de New-York City, *Propriété NYC*.

Plateformes d'observation Les techniques d'observation de la Terre sont un sujet de recherche et de développement critique pour les forces militaires. Des données historiques [52] indiquent l'utilisation de photos aériennes par British Royal Air Force à des fins de reconnaissance pendant la Première Guerre mondiale. Ces approches se sont généralisées lors de la seconde guerre mondiale [104] et [123]. En conséquence, la photographie aérienne et la photogrammétrie ont bénéficié d'énormes progrès.

Le 7 mars 1947, une caméra montée sur une fusée allemande V-2 modifiée a capturé la première image de la Terre à partir de l'espace. Cependant, étant donné que la fusée ne pouvait atteindre qu'une altitude légèrement supérieure à 150 kilomètres, elle n'a pas pu mettre en orbite sa charge utile. Néanmoins, un panorama jusque-là inédit 1.5 a pu être créé en raccordant plusieurs images. Nous conseillons l'ouvrage [39] pour une perspective historique de l'imagerie spatiale.

Débutant en 1960, le programme TIROS [154], *Television InfraRed Observation Satellite*, dirigé par la NASA, a prouvé l'efficacité des satellites d'observation pour étudier la Terre. L'objectif principal de ce programme était de développer un système d'information météorologique par satellite. Lancé le 1er avril 1960, TIROS-1 embarquait deux caméras de télévision 1.6 qui ont capturé des milliers d'images lors des 78 jours de mission 1.7. Une revue technique du satellite TIROS-1 est disponible dans [159].

Le succès du programme TIROS a été suivi de nombreuses autres missions importantes. Par exemple, le populaire programme Landsat [107] qui a débuté au début des années 1970 est toujours en activité. Il offre à ce jour l'enregistrement global continu le plus long de la surface terrestre [106]. Des programmes commerciaux ou publics plus récents tels que Worldview, Pleiades ou DubaiSat offrent une qualité et une résolution d'imagerie sans précédent. De plus, l'agilité de petits satellites tels que la constellation RapidEye ou les satellites SkySat permet une réponse rapide à des demandes d'acquisitions.

Nous avons également observé au cours de cette dernière décennie une démo-

FIGURE 1.5 – Premiere image de la Terre vu de l'espace, *Propriété NASA*.

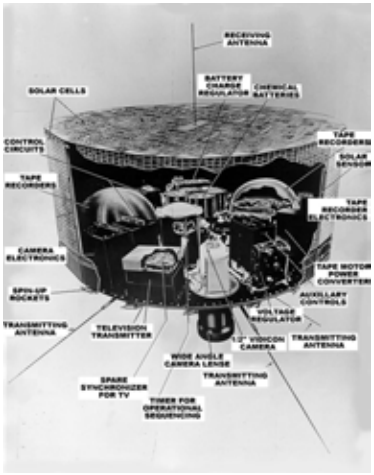


FIGURE 1.6 – Equipments du satellite TIROS-1, *Propriété NOAA*.

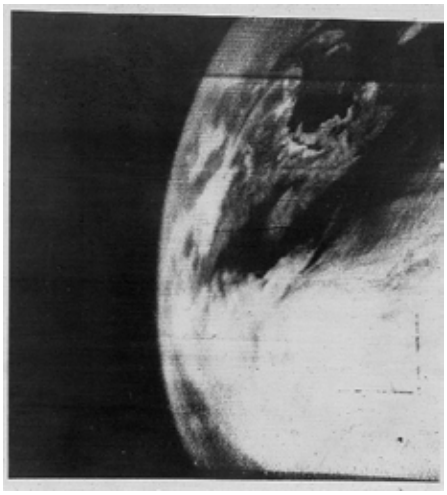


FIGURE 1.7 – Image de la Terre aquise par le satellite TIROS-1, *Propriété NASA*.

cratisation progressive des drones, créant une troisième option pour l'imagerie vue du ciel. Cependant, à ce jour, les drones restent plus adaptés à une imagerie locale et très précise. Par conséquent, ils semblent moins adaptés à notre tâche où de vastes zones doivent être cartographiées. En conséquence, nous nous concentrons uniquement sur l'imagerie aérienne et satellitaire.

Comparaison entre l'imagerie aerienne et satellitaire L'imagerie aérienne et satellitaire présente différent avantages et inconvéniants comme expliqué dans [61] and [164] :

Couverture : les images aériennes sont acquises grâce à des avions survolant une zone d'intérêt. Cela signifie que les endroits reclus peuvent être potentiellement difficiles à observer avec l'imagerie aérienne. En revanche, l'imagerie satellitaire offre généralement une couverture plus globale. De plus, la télédétection satellitaire permet d'observer des paysages de plus grande taille que l'enquête aérienne.

Réponse à un demande d'observation : Les orbites satellitaires définissent quand une zone géographique peut être observée. En fonction des positions et du cycle de l'orbite de la constellation satellitaire, la réponse à une requête d'observation peut prendre des heures, des jours ou même des semaines. Avec une planification appropriée, un avion peut se trouver à un endroit au moment désiré. En outre, si différents endroits doivent être imagés en même temps il est possible utiliser plus d'avions.

Météorologie : Un manteau nuageux important ou une forte pluie limite l'applicabilité des deux types d'acquisition. Les observations satellitaires sont plus sensible à la météorologie du fait que la lumière traverse l'intégralité de l'atmosphère terrestre. En revanche, les acquisitions aériennes sont moins susceptibles d'être affectées par des nuages haute altitude.

Données historiques : L'imagerie satellitaire bénéficie d'importantes archives historiques qui permettent parfois aux scientifiques de surveiller l'évolution temporelle d'un emplacement souhaité. Malheureusement de telles bases de données n'existent pas pour l'imagerie aérienne.

Precision d'acquisition : L'imagerie aérienne fournit généralement des images avec une résolution jusqu'à 6,50 cm. En revanche, l'imagerie satellitaire procure des images avec une résolution de 50 cm pour les applications non-militaire.

Le choix entre l'utilisation de l'imagerie satellitaire ou aérienne repose très largement sur l'application finale.

Comparaison entre la photogrammétrie et les acquisitions LiDAR Les techniques de cartographie par photogrammétrie et acquisition LiDAR présentent différents avantages et inconvénients comme détaillé dans [7] et [9] :

Observer à travers la cannopée : Les acquisitions LiDAR ont la possibilité de pénétrer les forêts denses [10]. Cela permet au LiDAR de cartographier avec une précision élevée la topographie du relief terrestre. Les techniques de photogrammétrie doivent utiliser des algorithmes pour estimer et soustraire la hauteur de la canopée [158].

Précision : La cartographie générée par acquisition LiDAR est généralement plus précise et plus dense. Cela est dû au fait que le LiDAR mesure directement les distances alors que la photogrammétrie mesure indirectement les distances.

Photographie : Les techniques de photogrammétrie peuvent produire en plus de la cartographie une image de la scène. Nous notons que certains capteurs LiDAR sont associés avec la caméra pour produire également une image de la scène.

Couverture : Bien que des bases de données d'acquisition LiDAR existent, elles demeurent limitées par rapport aux bases de données de photogrammétrie.

Comme pour la comparaison entre la l'imagerie aérienne et satellitaire, l'application finale définit quel type d'acquisition est le plus approprié.

1.1.2 Applications à la géologie

Les géologues ont rapidement adopté l'utilisation de la photogrammétrie et des acquisitions de LiDAR comme illustré dans [43], [69] et [167]. Nous examinons dans ces sections certaines de ces applications.

Cartographie de la topographie L'obtention d'une cartographie précise du paysage est extrêmement importante pour les géologues. Le modèle d'élévation numérique, DEM, peut être produit soit par des techniques de photogrammétrie, soit par traitement d'acquisitions de LiDAR [120]. Les DEM sont généralement des données d'entrée requises pour beaucoup de tâches différentes. Par exemple, les géologues peuvent surveiller l'évolution du paysage en profitant des bases de données d'images satellites. Le DEM aide également les géologues à préparer leur sondage sur le sol où seules des mesures éparses et locales peuvent être réalisées.

Catastrophes naturelles Les catastrophes naturelles se produisent de manière imprévisible et peuvent entraîner des dégâts et des pertes de vie. Ils perturbent généralement la surface terrestre et les environnements urbains comme expliqué dans [86]. Une mesure précise de cette perturbation ne permet pas seulement d'améliorer notre compréhension scientifique, mais également de mieux organiser les secours. En utilisant une série temporelle de DEMs acquises avant et après l'événement catastrophique, il est possible d'obtenir de précieuses informations.

Tremblement de terre Les caractéristiques du paysage et la déformation de la surface le long des failles actives fournissent des informations sur la tectonique [59]. Il existe différents types de failles comme illustré dans la figure 1.8. Les séismes sont généralement mesurés à l'aide de sysmomètres. Cependant, les stations GPS peuvent fournir une mesure précise mais locale de la déformation du sol [84]. Les progrès de la télédétection permettent également d'estimer cette

déformation à plus grande échelle mais au détriment de la précision [112]. L'utilisation d'un DEM pré et post-événement permet aux scientifiques de créer des cartes de la déformation. Ces cartes peuvent ensuite être combinées avec des mesures GPS ou des relevés au sol. L'ensemble de ces mesures fournissent des données importantes pour modéliser la physique du système de plaques.

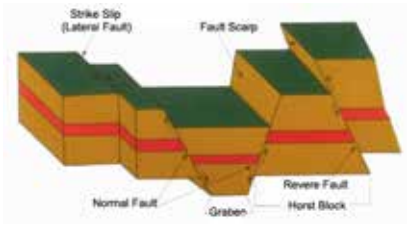


FIGURE 1.8 – Differents types de failles, *Propriété USGS*.

FIGURE 1.9 – Diagramme de glissement de terrain, *Propriété USGS*.

Liquéfaction du sol La liquéfaction du sol est la déformation du relief induite par un stress extérieur tel qu'un tremblement de terre [139]. Pendant la liquéfaction, le sol perd de sa rigidité conduisant à des dommages massifs. Par exemple, des bâtiments se sont inclinés lors du séisme de Niigata de 1964. La liquéfaction du sol est principalement un phénomène local. Par conséquent, des DEM de haute résolution sont nécessaires pour capturer cette déformation du relief.

Glissement de terrain Les glissements de terrain surviennent dans une masse de terre ou de roche d'une montagne ou d'une falaise, comme l'illustre la figure 1.9. Les géologues surveillent les glissements de terrain à trois niveaux [103] :

- En identifiant quelles pentes risquent d'être instables [77]. Cela fournit les informations nécessaires pour la prévention et le renforcement strucuturel de la pente.
- En surveillant les pentes instables pour déclencher des avertissements quelques minutes ou secondes avant un glissement de terrain [89]. C'est une préoccupation majeure pour l'exploitation minière à ciel ouvert.
- En mesurant la déformation de la pente due à un glissement de terrain [121]. Cela permet d'étalonner et d'améliorer les modèles numériques de glissement de terrain.

Dans ce contexte, autant les acquistions LiDAR que la photogrammétrie haute résolution sont pertinents pour extraire les informations.

Phénomène naturel Bien que les désastres naturels provoquent une déformation soudaine du paysage, de nombreux phénomènes naturels modifient progressivement la topographie des planètes.

Déplacement des dunes Une dune est une colline de sable construite par le vent ou un flux d'eau. Les dunes se déplacent, évoluent, fusionnent ou se divisent en raison des forces éoliennes et de la forme du bassin rocheux [157]. Les dunes existent non-seulement sur Terre, mais aussi sur Venus, Mars et Titan. À l'aide de séries temporelles de DEM de champs de dunes obtenus par photogrammétrie, des chercheurs ont récemment pu démontrer que Mars est une planète géologiquement active [155]. Des études suivantes ont même été en mesure d'estimer les cycles climatiques à partir du mouvement des champs de dunes [4].

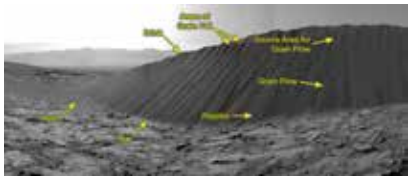


FIGURE 1.10 – Dune de sable Namib sur Mars (Curiosity rover ; 17 December 2015), *Propriété JPL.*

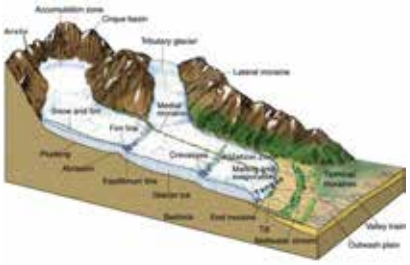


FIGURE 1.11 – Diagramme représentant le déplacement d'un glacier, *Propriété USGS.*

Mouvement des glaciers Les glaciers sont d'une importance cruciale car ils fournissent une grande quantité d'eau potable dans certaines régions. La compréhension de l'évolution de la masse des glaciers et des paquets de neige est cruciale pour atténuer les impacts sur l'approvisionnement en eau, le niveau de la mer et les dangers liés aux inondations et aux avalanches. En outre, une prédiction précise de cette évolution nécessite une compréhension de la nature et de la réponse des glaciers à diverses forces. Les changements de précipitations, de température, d'ensoleillement et des contaminants superficiels contribuent à l'avancée et à la retraite des glaciers. Par nature, les glaciers sont difficiles à accéder. Par conséquent, la télédétection offre une technique d'observation pratique. En utilisant des séries chronologiques de DEM, il est possible de suivre le mouvement des glaciers [151].

1.1.3 Un problème d'appariement d'images.

Du point de vue de la modélisation, nous pouvons formuler l'onservation des désastres et phénomènes naturels comme un problème d'appariement d'images

dense. Un problème d'appariement d'image dense est la tâche de trouver pour chaque pixel d'une image donnée son homologue dans une autre image. Selon l'application finale, le problème d'appariement aura un degré de liberté différent. Si les homologues doivent être sur une certaine ligne, nous obtenons un problème d'appariement mono-dimmensionnel. Si les homologues sont dans un plan donné, nous obtenons un problème d'appariement bi-dimmensionnel. Si aucune hypothèse n'est possible, nous avons un problème d'appariement tri-dimmensionnel.

Hypothèse sur les DEM

Comme nos observations proviennent d'un avion ou d'un satellite, nous proposons de transformer avec très peu de perte d'information le DEM acquis en une carte d'élévation d'image [105]. La position des pixels définit les coordonnées géographiques locales tandis que leurs intensités encodent l'élévation au-dessus d'une référence. Nous détaillons cela dans le chapitre 6.2.3 de cette thèse.

Appariement 1D

Pour la photogrammétrie aérienne, nous recevons deux images acquises par une caméra de type *frame*. Après avoir prise en compte la géométrie d'acquisition, le calcul de la carte d'élévation repose principalement sur la résolution d'un problème d'appariement dense 1D appelé appariement stéréo. Nous présentons dans la section 6.3 le concept d'appariement stéréo.

Appariement 2D

Pour la photogrammétrie satellitaire, nous recevons deux images acquises par un capteur de type *push-broom*. Cela complexifie la géométrie de l'acquisition. Par conséquent, le calcul de la carte d'élévation repose sur la résolution d'un problème d'appariement dense 2D. Nous en discutons dans la section 6.3.1.

Appariement 3D

Etant donné une série chronologique de cartes d'élévations du même paysage, nous pouvons effectuer un appariement 3D pour estimer la déformation induite par un phénomène d'intérêt. Dans ce contexte, les cartes d'élévations peuvent être produites par photogrammétrie ou par maillage d'une acquisition LiDAR. Nous en discutons dans la section 6.4.

Chapter 2

Introduction (English language)

We begin this first chapter by introducing in section 2.1 the context of our work. In particular, we explain how remote sensing techniques are used for geological studies and how this relates to the computer vision task of image registration. We review in section 2.2 some of the past work. The section 2.3 describes the mathematical modeling we use through this document. Finally, in section 2.4 we present a summary of the technical content of this thesis and our contributions.

2.1 Context

2.1.1 Monitoring from above

The last century has seen an increasing number of techniques and devices to monitor Earth. One major development is the use of satellites and airplanes equipped with imaging sensors to monitor the Earth from above. With the development of space technology satellites have extensively surveyed Earth and other planets such as Mars with an increasing accuracy. We now dispose of many high resolution pictures of many planets and their natural satellites.

One can classify monitoring sensors in two main families. The active sensors such as LiDAR and Radar that record the reflection of the signal they emitted. On the contrary, the passive sensors such as panchromatic, color or hyper-spectral cameras directly record the signal emitted by the observed scene.

Observing sensors In this work we consider acquisitions from panchromatic cameras and LiDAR sensors.

Photogrammetry Photogrammetry refers to the set of techniques that use photography to measure the distances between objects. From 1850 to 1900,

people relied on techniques such as those of the plane tables to draw maps. This is illustrated by figure 2.1. The analog photogrammetry which spanned the period from 1900 to 1960 relied on the concept of stereo-metric vision. However, an operator was still performing the essential registration task as depicted in figure 2.2. From 1960 the availability of computers progressively has decreased the need for human involvement. Starting from 2000's the modern photogrammetry completely relies on fully digitalized data and requires very little human intervention. For more detail, we refer the curious reader to [73], [66], [115] and [80] for an historical review, and to [76] for an introduction to the mathematical foundations.

Figure 2.1 – Operator manipulating a plane table. *Courtesy NOAA.*

Figure 2.2 – Operator drawing maps during the analog photogrammetry era. *Courtesy WSP group.*

LiDAR The NASA invested in the 1970s in the development of modern laser-based remote sensing acquisition techniques. This initial study was mainly aimed at measuring the properties of the atmosphere and the ocean water, the forest canopy or the ice sheets. In the 1990s the accessibility to positioning devices such as GPS, Global Positioning Systeme, and IMU, Inertial Measurement Unit, allowed to use LiDAR for mapping purposes. Finally, in the 2000s the availability of processing software coupled with the decreased cost of computing infrastructures made the LiDAR a precise and efficient mapping tool as illustrated in figures 2.3 and 2.4. For more detail, we refer the curious reader to [144], [5], [11] and [31] for a review of the technology and its application, and to [6] for an introduction to the mathematical foundations.

Observing platforms Earth observation techniques have been a major research and development study for military forces. Records indicate that aerial photos used by British R.A.F. for reconnaissance helped to review military tactics during World War I [52]. Such practice was generalized during World

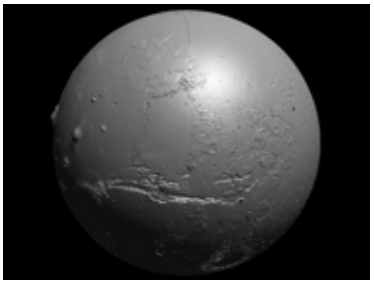


Figure 2.3 – Rendered LiDAR acquisitions of Mars.

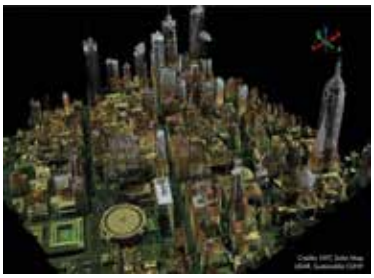


Figure 2.4 – 3d points cloud of New-York City acquired by a LiDAR device augmented with a color camera.

War II [104] and [123]. As a result aerial photography and photogrammetry made tremendous strides.

On March 7, 1947, a camera mounted on a modified German V-2 rocket captured the first picture of Earth from space. However, since the rocket could only reach an altitude slightly above 100 miles, it was unable to place into orbit its payload. Nevertheless, stitching several pictures together as illustrated in 2.5 created unseen before panoramas. We advise the reader to [39] for an historical perspective of imaging from space .

Figure 2.5 – The first picture of Earth from space, *Courtesy NASA*.

Starting in 1960, the TIROS Program [154], *Television InfraRed Observation Satellite*, directed by NASA proved the efficiency of observation satellites to study Earth. The main focus of this program was to develop a meteorological

satellite information system. Launched on April 1, 1960, TIROS-1 embarked two television cameras 2.6 that captured thousands of pictures during its 78 days mission 2.7. A technical review of the TIROS-1 satellite is given in [159].

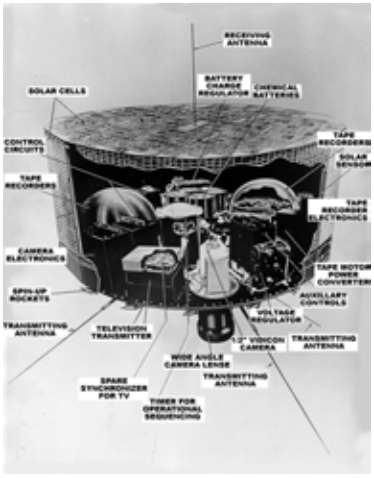


Figure 2.6 – Equipments of the TIROS-1 satellite, *Courtesy NOAA*.

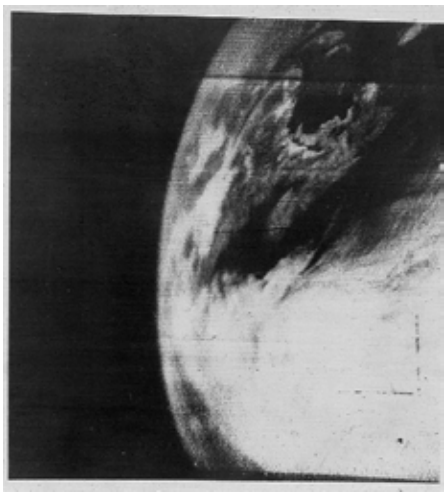


Figure 2.7 – Picture of Earth from the TIROS-1 satellite, *Courtesy NASA*.

The success of the TIROS program was followed by many other important missions. For instance, the popular Landsat program [107] that started in the early 1970s is still operating. It offers to this day the longest continuous global record of the Earth surface [106]. More recent commercial or public programs such as Worldview, Pleiades or DubaiSat offer unprecedented imaging quality and resolution. Moreover, the agility of small satellites such as the RapidEye constellation or the SkySat satellites allow for quick response to on demand acquisitions.

We also observed over the last decade a progressive democratization of drones, creating a third option for imaging from above. However, to this day drones remain more suited to very precise and local imaging. Hence, they appear less suited to our task where large areas need to be mapped. As a result we only focus on aerial and satellite imaging.

Aerial vs Satellite imaging Aerial and satellite imaging present different strong points and weaknesses as explained in [61] and [164]:

Coverage : aerial images are mainly gathered through the use of airplanes flying over the landscape of interest. This means that remote places can be difficult to survey with aerial imaging while satellite imaging generally offer a global coverage. Moreover, remote sensing satellite allows imaging of way larger landscapes than aerial survey.

Timing : the satellite orbits defines a time frame when a defined location can be imaged. Depending on the positions and the orbit cycle of the satellite constellation, the response to survey at a given location can take hours, days or even weeks. With proper planning an airplane can be at a certain location at the desired time. Moreover, if different locations need to be imaged at the same time one can always use more planes.

Weather : severe cloud coverage or rain nullify both types of acquisition. Since the light reaching the satellite is affected by the entire atmosphere, space survey tends to be more sensitive to the weather. For instance, aerial acquisitions are less likely to be affected by high altitude clouds.

Historical data : satellite imaging benefits from massive historical archives that sometimes allows observers to monitor the evolution of a desired location through time. Such large databases do not exist for aerial survey.

Image resolution : an aerial survey generally provides images with a resolution down to 6.50cm. Satellite survey captures images with resolution down to 50cm for the public viewing.

As it should always be, the final task is the one driving the choice between aerial and satellite imaging.

Photogrammetry vs Lidar Photogrammetry and Lidar mapping techniques present different benefits and drawbacks as detailed in [7] and [9]:

Canopy penetration : LiDAR has the ability to penetrate even dense forest canopies [10]. This allows the LiDAR to map with high accuracy the bare earth topography. The photogrammetry techniques have to rely on algorithms to remove the estimated canopy height [158].

Precision : As a rule of thumb, the mapping generated from LiDAR is generally more precise and dense. This is because the LiDAR directly measures distances while photogrammetry uses a proxy measurement (image registration) to estimate those distances.

Photography : The photogrammetry techniques can produce along with the mapping a picture of the scene. We note that some LiDAR are augmented with camera to also produce a picture of the scene.

Coverage : While LiDAR databases exist they do not compare in extent with photogrammetry databases.

As for the previous discussion comparing the aerial and satellite imaging, the final task will drive which type of acquisition is the most relevant.

2.1.2 Application to geology

Geologists and Earth scientists were among the early users of photogrammetry and LiDAR acquisitions as illustrated in [43], [69] and [167]. We review in this sections some of their applications.

Mapping topography Obtaining precise mapping of the landscape is extremely important to geologists. Digital Elevation Model, DEM, can be produced using either by photogrammetry techniques or by processing LiDAR acquisitions [120]. DEMs are generally required inputs for a lot of different tasks. For instance, geologists can monitor the evolution of the landscape by taking advantage of the satellite images databases. DEM also helps geologists to prepare their ground survey where only sparse and local measurements can be made.

Natural hazards Natural hazards occur unpredictably and can cause widespread damage and loss of life. They usually disrupt the Earth surface or built environment as explain in depth in [86]. Accurate measurement of this disruption non only helps to improve our scientific understanding but allows to better organize the emergency response. Using a time series of DEMs spanning before and after the catastrophic event one can automatically derive precious information.

Earthquake Landscape features and surface deformation along active faults provide insights into faulting and tectonics [59]. There exists different type of faults as illustrated in figure 2.8. Earthquakes are generally measured using seismometers. However, GPS stations can provide accurate but local measurement of the ground deformation [84]. The progress of remote sensing imagery also allows to estimate this deformation at a larger scale at the expense of precision [112]. The use of a pre and post-event DEMs allows researcher to create maps of the deformation. Those maps can then be used to augment GPS point measurements or ground surveys. All those measurements provide important data to estimate the physical modeling of fault systems.

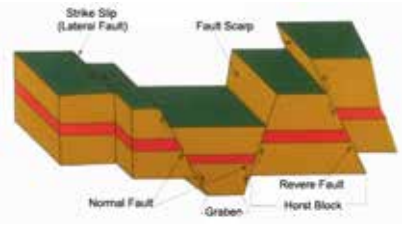


Figure 2.8 – Different types of faults.

Figure 2.9 – Landslide diagram.

Soil liquefaction Soil liquefaction is the deformation of the landscape induced by an external stress such as an earthquake [139]. During liquefaction,

The soil loses strength and stiffness creating massive damages. For instance buildings were tilted during the 1964 Niigata earthquake. Soil liquefaction is mainly a local phenomenon. Hence, high resolution DEMs are necessary to capture landscape deformation induced by soil liquefaction

Landslides Landslides are sliding down of a mass of earth or rock from a mountain or cliff as illustrated by figure 2.9. Geologists monitor landslides at three levels [103]:

- Identifying which slopes are likely to be unstable [77]. This provides the necessary information for prevention and potential structural reinforcement of the slope.
- Monitoring at high-frequency unstable slopes to trigger early landslide warnings [89]. This is a main concern for open-pit mining.
- Measuring the slope deformation due to a landslide [121]. This help to calibrate and improve numerical landslide models.

In this context, one can rely on LiDAR processing or high resolution photogrammetry to extract the pertinent information.

Natural processes While natural hazards trigger sudden deformation of the landscape, many slow natural processes progressively alter planets topography.

Motion of dunes A dune is a hill of loose sand built by wind or the flow of water. Dunes move, evolve, merge or divide due to eolian forces and the shape of the bed rock [157]. Dunes non-only exist on Earth but also on Venus, Mars and Titan. Using time series of DEMs of dune fields obtained with photogrammetry, researchers have recently been able to demonstrate that Mars is geologically active [155]. Furthermore, follow-up studies were even able to estimate climate cycles from the motion of dune fields [4].

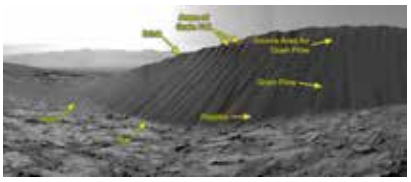


Figure 2.10 – Namib sand dune (downwind side) on Mars (Curiosity rover; 17 December 2015).

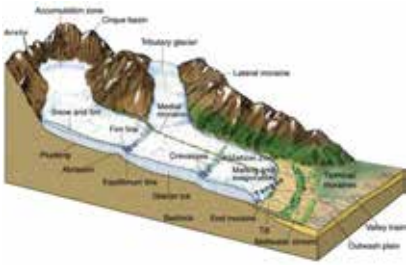


Figure 2.11 – Glacier diagram.

Motion of glaciers and ice Glaciers are of critical importance since they provide large quantity of drinking water in some areas. Understanding the changing mass balances of mountain glaciers and snow packs is crucial for mitigating impacts to water supplies, sea level, and hazards from outburst floods and avalanches. Moreover, accurate prediction of future mass balance changes requires an understanding of the nature and rate of glacier response to various forces. Changes in precipitation, temperature, solar input and surface contaminants contribute to glaciers advance and retreat. By nature, glaciers are difficult to access. Hence, remote sensing provides a handy observation technique. Using time series of DEMs, one can track the motion of glaciers or the change in ice coverage [151].

2.1.3 An image registration problem

From a modeling standpoint, we can formulate the monitoring of natural hazards and processes as a dense image registration problem. A dense image registration problem is the task of finding for each pixel of a given image its counterpart in an other image. Depending on the task the registration problem will have different degrees of freedom. If we know that the counterparts have to be on a certain line we get a 1D registration problem. If the know that the counterparts are on a given plane, we end-up with a 2D registration problem. If no assumption can be made, then we have a 3D registration problem.

Assumption for DEM

Since our observations originate from a plane or a satellite, we propose to transform with very little loss of information the acquired DEM to an image elevation map [105]. The position of the pixels refers to local geographic coordinates while their intensities encode the elevation above a reference. We detail this in the chapter 6.2.3 of this thesis.

1D Registration

For aerial photogrammetry we are given two images acquired by a frame camera. After accounting for the geometry of acquisition, the computation of the elevation map mainly relies on solving a 1D dense registration problem called stereo-matching. We will in the last chapter 6.3 present the stereo-matching concept.

2D Registration

For satellite photogrammetry we are given two images acquired by a push-broom sensor. This complexifies the geometry of acquisition. Hence, the computation of the elevation map relies on solving a 2D dense registration problem apparent to optical-flow. We will discuss this in the last chapter 6.3.1.

3D Registration

Given a time series of elevation maps of the same landscape, we can perform a 3D registration to estimate the deformation induced by a phenomenon of interest. In this context the elevation maps could have been produced by photogrammetry or by griding a LiDAR. We will discuss this in the last chapter 6.4.

2.2 Reviewing previous work

Dense image registration has been extensively studied since the inception of computer vision. We can find dense image registration problems in many fields such as medical and biology imagery, planetary sciences, industrial verification or video surveillance. We point the curious reader toward [23], [178], [45] and [138] for an extensive review on image registration.

2.2.1 Priors

Independently on their approach, all registration methods try to enforce a matching prior and a regularization prior.

Matching priors

The goal of the matching prior is to measure the similarity between parts of two images. There exist many different approaches as illustrated in [67], [29], [163], [149], [150] and [176]. The simplest technique is to directly compare the value of pixels, eventually on a patch centered around the points of interest. More advanced technique computes descriptors that encode non only the information contained by the pixel of interest but also its neighborhood. These descriptors or features are then compared to estimate a similarity score. Unfortunately, relying only on matching priors is insufficient to register images. For instance, noisy regions or geometric deformations deteriorate the quality of the similarity estimation. Moreover, texture-less areas or repetitive patterns create ambiguities. Finally, in some context some pixels have no corresponding counterparts due to occlusion.

Regularization priors

The role of the regularization prior is to enforce some *a-priori* information on how the pixels should register. In most tasks, one can assume that the geometric transformation that registers the images should follow some structure. The regularization priors simply define the properties that geometric transformations should follow. This helps to correct some of the errors or uncertainty of the matching prior. The most popular regularization favors geometric transformation that have a smooth gradient. The choice of the regularization priors mainly depends on the task and the ability to enforce it to a desired accuracy. One

can refer to [30] and [156] for examples of different regularization priors in the context of image registration.

2.2.2 Framework

We distinguish two main frameworks to enforce both the matching and regularization priors. While both frameworks have the same intent, their theoretical foundations differ.

Heuristic framework

The heuristic based framework relies on alternate application of heuristics to enforce the priors. Generally, the matching prior is enforced first giving a noisy and possibly sparse estimation of the registration. Then, the regularization prior is applied to obtain a dense and denoized registration. For instance, the median filter has been extensively used as a regularization prior. Eventually, one can proceed with multiple rounds of alternating between the priors. The decoupling of the prior's enforcement in successive steps leads to simple algorithm. However, in this setting it is unclear what is globally enforced. Indeed, alternating between priors is different than directly enforcing both priors at the same time.

Optimization framework

The optimization based framework formulates the registration task as an energy minimization problem. This framework makes use of a global energy obtained by modeling both priors with their respective energies. The main challenge remains to find a registration that minimizes this energy. We remind that one can link an energy to a probability through the use of Gibb's free energy function. Hence, the energy minimization problem is in fact equivalent to finding the most probable registration given the inputs. This connection gives strong theoretical foundation to this framework. Moreover, in some cases, the optimization problem can generalize a heuristic. For instance, it is well known that the median filter is equivalent to solving a certain ℓ_1-norm problem. This framework gives more precise results than its heuristic base counterpart (see an example in the context of 2D image registration [8]). However, this comes with an added computational complexity. In this work, we will investigate different approaches for the optimization framework.

2.3 An approximate modeling

2.3.1 Mathematical modeling

We now assume that we are given two images: a reference image and a target image. The goal of the dense image registration task is to find for each pixel of the reference image its counterpart in the target image. Without loss of generality

we can model this problem as finding the apparent pixel motion between the reference image and the target image.

Notations We introduce the following notations:

- Ω is the finite set $\{1, \ldots, n\}$ with $n \in \mathbb{N}^*$ that represents the pixels of the reference image.
- $\mathcal{X}_i$ with $i \in \Omega$ is a convex subset of $\mathbb{R}^d$ with $d \in \mathbb{N}^*$ that defines the admissible range of motions for a given pixel.
- $\mathcal{X}$ is a convex subset formed by $\mathcal{X}_0 \times \ldots \times \mathcal{X}_n$: that represent the admissible registrations.
- x is an element of $\mathcal{X}$ and represents a potential registration.
- x_i is the i^{th} element of x and represents the motion of a given pixel.
- $M_i : x_i \in \mathcal{X}_i \rightarrow \mathbb{R}$ encodes the matching prior as an energy.
- L is a continuous linear operator mapping space $\mathcal{X}$ to a vector space $\mathcal{Y}$ that encodes the dependency between the pixel motions. This is the first part of the regularization prior.
- $R : y \in \mathcal{Y} \rightarrow \mathbb{R}$ encodes as an energy the second part of the regularization prior.

Optimization model We now need to optimize the following model:

$$\arg\min_{x \in \mathcal{X}} \sum_{i \in \Omega} M_i(x_i) + R(Lx) \tag{2.1}$$

This mathematical problem is extremely difficult to solve. Indeed, the objective function exhibits the following properties:

- Non convexity: there is no convexity assumption on the functions $(M_i)_i$ and R, hence the objective function can be non-convex.
- Non smoothness: there is no smoothness assumption on the functions $(M_i)_i$ and R, hence the objective function can be non-smooth.
- Continuous variables: we assume that each variable x_i lives in a continuous convex space.
- High order terms: the operator L can create dependencies between variables in sub-sets of x.
- Large number of variables: since we work in the image registration context, the size of x is in the order of millions of elements.

For all these reasons, the problem (2.1) is generally NP-hard and only approximate solution may be sought. Notice however that in practice an approximate solution is good enough because there is no guarantee that the mathematical model is entirely faithful to the reality.

2.3.2 Approximations

Directly attempting to solve (2.1) is extremely challenging. We need to make further assumptions on the objective function to decrease its optimization complexity. By using various simplification strategies we obtain different classes of optimization problems that become tractable. However, this simplification makes the modeling of the registration task less accurate.

Discarding non convexity

We propose to use convex functions to approximate the matching and regularization functions $(M_i)_i$ and R around the current solution. The quality of the approximation generally quickly deteriorates with an increased distance to the current solution. However, this approximation gives us a convex function that represents locally the original objective function. In this settings we can use various optimization schemes.

Majorization minimization The majorization minimization scheme relies on using a convex surrogate function [82]. The surrogate function needs to coincide with the objective function at the current solution and should majorize the objective function everywhere else. Since we have some freedom to define the surrogate function, we have interest to choose one that is easy to minimize. The most appealing surrogate functions are those that can be minimized with a close-form formula. The minimization of the surrogate function gives a new solution. We iterate between these two steps until no further progress can be made.

The majorization minimization scheme is easy to apply and implement. However, it suffers of two main drawbacks. If the function is not smooth, then this scheme can terminate in a sub-optimal solution. Moreover, the convergence rate generally slows drastically when it approaches a minimum solution.

Splitting techniques The splitting techniques rely on auxiliary variables that are introduced to decouple the matching and regularization parts of the energy (see [33] and [32] for a detailed introduction). In our context, this means to add a set of variables y in place of the term Lx. Then, an additional constraint is added to the objective function to enforce that $y = Lx$. We can use penalty terms, Lagrangian multipliers or ADDM, Alternating Direction Multiplier Method, schemes to enforce this new constraint. This creates different splitting techniques. In most cases the splitting scheme alternates between 3 optimizing steps: optimizing the matching variables x, optimizing the regularization variables y, and enforcing the constraint $y = Lx$.

The splitting techniques scheme is generally more difficult to apply and implement than the majorization minimization scheme. However, this scheme has the ability to minimize non smooth functions. Unfortunately, enforcing the constraint slows down the convergence of the splitting techniques.

Primal-dual techniques The primal dual scheme of [27] relies on the concept of duality to add auxiliary variables that dynamically approximate the regularization function R by a set of linear terms. In our context, this means to add a set of variables y that model $R(Lx)$ by the scalar product $< y, Lx >$. We notice that the auxiliary variables make the optimization over each x_i independent. Hence, the optimization over x is easy as long as the functions $(M_i)_i$ are reasonably complex. This creates a very appealing scheme where we iterate between minimizing for each x_i and update the variable y to get a better approximation.

The primal dual scheme can handle non-smooth objective functions and has superior convergence properties than the splitting techniques. For these reasons we elect to choose the primal-dual scheme.

Discretization of the solution space and first order regularization

Another approach to simplify the problem (2.1) is to discretize the solution space X and to limit the linear operator L to a first order operator like a gradient. These two assumptions make the problem (2.1) belong to the class of first order pairwise MRF, Markov Random Field if the regularization function R does not rely on input images, or CRF, Conditional Random Field if it does. In this context, we can use different optimization schemes. Unfortunately, even with this simplification the problem remains generally NP-Hard. Hence, we are only guaranteed to obtain an approximate solution.

Message passing The message passing method builds on dynamic programming schemes (see [129] and [171] for a detailed introduction). It relies on propagating information through the variables to update their respective probabilities of being assigned to a given discrete value. There exist many approaches to propagate the information. However in the context of image based problems, the belief propagation with the checkerboard update rule of [49] and the tree-reweighted message passing scheme of [93] obtain the best results. Moreover, if the problem takes the form of a chain we can use the famous Viterbi algorithm [56] to compute an optimal solution.

The message passing unfortunately does not always obtain good approximations as demonstrated in [161] and [90]. Moreover, the procedure of these algorithms is quadratic with respect to the average number of discrete possible values for variables $(x_i)_i$. We note that for some regularization functions R, this complexity can be reduced to a linear one as explained in [48].

Dual decomposition The dual decomposition scheme relies on duplicating some variables of the original problem to obtain a set of sub-problems that can be optimally solved. The duplicated variables are constrained through the use of auxiliary variables to converge to the same discrete value. This scheme alternates between solving each sub-problem and updating the auxiliary variables to enforce the constraints. Two main variations exist. The original dual decomposition algorithm of [99] used Lagrangian multipliers to enforce the constraints while

the alternative direction dual decomposition algorithm of [119] makes use of the ADMM scheme. In our context the most natural and efficient decomposition is to create a chain per line and column.

The dual decomposition scheme obtains excellent approximations as demonstrated in [98]. However, it is slow to converge since it has the same complexity as the underlying algorithm used to solve the sub problems. Moreover, many iterations are needed to enforce the constraint on the auxiliary variables.

Graph-Cuts The Graph-Cuts approach of [18] also known as the making move approach, relies on iteratively updating a current solution by solving a binary problem. During an iteration, each variable can choose between maintaining its current solution or choosing a proposed one. By cycling the proposed solution through the list of admissible discrete solutions, the making move approach progressively obtains a better solution. Interestingly, the associated binary problem is in fact a maxflow-mincut problem which can be solved very efficiently. Different approaches exist and result in different making move algorithms such as [19], [20] or [152].

The making move proposes a good balance between speed and the quality of approximations. Moreover, the complexity of algorithm such as the alpha expansion of [17] and Fast-PD [102] is linear with respect to the average number of discrete possible values for variable $(x_i)_i$. For these reasons we elect to investigate the alpha expansion and Fast-PD .

2.4 Thesis overview

2.4.1 Document organization

Besides the current introduction and conclusion, this manuscript articulates around three technical chapters 3, 4, 5 and one applications chapter 6.

First order Primal-Dual techniques for convex optimization In this first technical chapter 3 we start with the basis of convex optimization. We follow with a didactic review of first order primal dual scheme for convex optimization. Then, we study the dual optimal space of TV regularized problems. Finally, we perform experiments to illustrate the techniques presented in this chapter.

Maxflow and Graph cuts techniques The second technical chapter 4 introduces the basis of discrete optimization. We then review the maxflow-mincut problem in the context of graph cuts techniques. Then, we provide an extensive discussion around the implementation of the Fast-PD algorithm. Finally, we justify the superiority of our implementation in numerous experiments.

Coarsening schemes for optimization techniques The last technical chapter 5 compares various coarsening schemes for the first order Primal-Dual techniques and graph cuts schemes. We also introduce a new pyramidal scheme for graph cuts that drastically speeds-up the computation without compromising on the quality of the obtained solutions.

Applications The last chapter of this manuscript 6 revolves around the applications of techniques presented in chapters 3, 4 and 5. We perform experiments with tasks such as stereo-matching, monitoring Earth crust deformation and damage detection due to an earthquake.

2.4.2 Contributions

We present here a summary of our contributions that we further detail in each technical chapter.

First order Primal-Dual techniques for convex optimization We derive theorems that explain how dual optimal solution spaces relate to one another for TV regularized problems. This understanding helps to derive a new proof to a variety of theorems.

Maxflow and Graph cuts techniques We completely re-implemented the Fast-PD algorithm to obtain a drastic reduction of the memory footprint while providing faster run-time. This allows us to use Fast-PD on large scale problems on a modern laptop computer.

Coarsening scheme for optimization techniques We propose a coarsening scheme that speeds up the optimization of Graph-Cuts techniques. We extend this coarsening scheme with a novel framework that drastically speeds-up the inference run-time while maintaining remarkable accuracy.

2.4.3 List of publications

- Inference by learning: Speeding-up graphical model optimization via a coarse-to-fine cascade of pruning classifiers, B. Conejo, N. Komodakis, S. Leprince, J.P. Avouac in *Advances in Neural Information Processing Systems, 2014*
- Fast global stereo matching via energy pyramid minimization, B. Conejo, S. Leprince, F. Ayoub, J.P. Avouac in *ISPRS Annals of the Photogrammetry, Remote Sensing, 2014*
- A 2D and 3D registration framework for remote-sensing data, B. Conejo, S. Leprince, F. Ayoub, J. Avouac in *AGU Fall Meeting 2013*

- Emerging techniques to quantify 3D ground deformation using high resolution optical imagery and multi-temporal LiDAR, S. Leprince, F. Ayoub, B. Conejo, J. Avouac in *AGU Fall Meeting 2012*

Chapter 3

First order Primal-Dual techniques for convex optimization

3.1 Introduction and contributions

3.1.1 Introduction

This chapter introduces an extremely powerful and versatile framework: the First Order Primal Dual convex optimization technique. Sometimes referred as the Primal Dual Hybrid Gradient method or the Primal Dual Proximal Point method, this technique is well suited for a large class of convex optimization problems with smooth and non-smooth objective function. This framework has been successfully applied to problems such as image denoising (ROF, TV-L1, ...), inpainting, debluring, image segmentation, dense registration, mincut/maxflow, and linear programming.

The underlying mathematical mechanism can appear quite frightening at first glance, with terms such as "Moreau envelope", "Proximity operator" or "Flenchel transform". However, the framework is in fact fairly easy to understand. A large quantity of published materials or technical reports extensively cover the subject. As a personal preference, I would recommend the seminal paper of Chambolle and Pock [27] for its clarity and exhaustive list of experiments related to computer vision. Another great source of information is the technical report of Parikh and Boyd [141] that covers in more depth the proximal operators.

Finally, implementing a first order primal dual algorithm is relatively straightforward on both CPU and GPU based architectures. For instance, the core of the Mincut/Maxflow algorithm is only 10 lines of Matlab code and barely more in C++. Furthermore, many components can be reused from one algorithm

to the next, which makes this technique well suited for quick prototyping or modular framework.

3.1.2 Chapter organization

In section 3.2 we introduce the basics of convex optimization and we describe the problems of interest. We progressively explain and detail in section 3.3 the different components of the first order primal dual algorithms. The section 3.4 presents frequently used proximal operators and Fenchel transformations. In section 3.5 we introduce various TV-regularized problems and we study the relationship between the optimal dual solution spaces of the TV-regularized problems. The section 3.6 illustrates the application of the first order primal dual techniques with different examples.

3.1.3 Contributions

In this chapter we demonstrate some TV regularized problems share a particular connexion through their optimal dual solution space. We prove that there exists a hierarchy of optimal dual solution spaces connecting the ROF model to a linear TV model. Furthermore, we establish that the intersection of the optimal dual solution space of a set of ROF models defines the optimal dual space of some linear TV model. To the best of our knowledge these are new results.

Building on these theorems, we state and prove a generalization of the Friedmann's theorems for Fused Lasso approximation. We also propose a new proof for quickly finding the solution of a ROF model with various global regularization terms. Finally, we introduce a new primal-dual formulation to solve the ROF model that experimentally outperform the traditional primal-dual scheme.

3.2 Problem formulation

3.2.1 Basics of convex optimization in a tiny nutshell

As a preamble, we remind some mathematical definitions useful in the context of this chapter. For an extended introduction on convex optimization, we refer the curious reader to *Fundamentals of Convex Analysis* by Hiriart-Urruty and Lemaréchal [78], *Convex Optimization* by Boyd and Vandenberghe [15] or *Optimization. Applications in image processing.* by Nikolova [133] from which we borrow the following definitions and theorems.

Existence of minimums

Before even thinking of searching for the minimums of a function, we need to establish the conditions of their existence. To this end, we need the following definitions:

Definition 1. A function f on a normed real vector space V is proper if $f : V \to]-\infty, +\infty]$ and if it is not identically equal to $+\infty$.

Definition 2. A function f on a normed real vector space V is coercive if $\lim_{\|u\| \to +\infty} f(u) = +\infty$.

Definition 3. A function f on a real topological space $\mathcal{X}$, say $f : V \to]-\infty, \infty]$ is lower semi-continuous (l.s.c.) if $\forall \lambda \in \mathbb{R}$ the set $\{u \in V : f(u) \leqslant \lambda\}$ is closed in V.

Now, we can state the following theorem:

Theorem 1. *Let $U \subset \mathbb{R}^n$ be non-empty and closed, and $f : \mathbb{R}^n \to \mathbb{R}$ be l.s.c. and proper. In the case that U is not bounded, we also suppose that f is coercive. Then, $\exists \hat{u} \in U$ such that $f(\hat{u}) = \inf_{u \in U} f(u)$.*

Gradient and subgradient

For the following definitions we assume that V is normed real vector space.

Definition 4. A function $f : U \subset V \to \mathbb{R}$ is said differentiable at $v \in U$ if the following limit exists:

$$\lim_{h \to 0} \frac{f(v+h) - f(v)}{h} \tag{3.1}$$

Definition 5. A function $f : U \subset V \to \mathbb{R}$ is smooth if it is differentiable $\forall u \in U$.

Definition 6. A function $f : U \subset V \to \mathbb{R}$ is not smooth if $\exists u \in U$ where f is not differentiable.

When f is not smooth we can extend the notion of derivative with subderivative (also refereed to as subgradient):

Definition 7. The subderivative of function $f : U \subset V \to \mathbb{R}$ at $u \in U$ is the set of $p \in \mathbb{R}$ verifying $\forall v \in U$:

$$f(u) - f(v) \geqslant \langle p, u - v \rangle \tag{3.2}$$

We note $\partial f(u)$ the subderivative of f at $u \in U$.

Convexity

Let us properly introduce the notion of convexity for spaces and functions.

Definition 8. Let V be any real vector space. $U \subset V$ is convex if $\forall (u, v) \in U \times U$ and $\forall \theta \in]0, 1[$, we have:

$$\theta u + (1 - \theta) v \in U. \tag{3.3}$$

Definition 9. A proper function $f : U \subset V \to \mathbb{R}$ is convex if $\forall (u, v) \in U \times U$ and $\forall \theta \in (0, 1)$, we have:

$$f(\theta u + (1 - \theta)v) \leqslant f(u) + (1 - \theta)f(v) \tag{3.4}$$

f is strictly convex when the inequality is strict whenever $u \neq v$.

The space of strong convex functions is a subset of convex functions.

Definition 10. A proper function $f : U \subset V \to \mathcal{R}$ is strongly convex with convexity parameter $\alpha \in \mathbb{R}^+$ if $\forall (u, v) \in U \times U$ and $\forall p \in \partial f(v)$ we have:

$$f(u) \geqslant f(v) + < p, u - v > + \frac{\alpha}{2}\|u - v\|_2^2 \tag{3.5}$$

The uniform convexity generalizes the concept of strongly convex function:

Definition 11. A proper function $f : U \subset V \to \mathcal{R}$ is uniformly convex with modulus ϕ if $\forall (u, v) \in U \times U$ and $\forall t \in [0, 1]$ we have:

$$f(tx + (1 - t)y) \leqslant tf(x) + (1 - t)f(y) - t(1 - t)\phi(\|x - y\|) \tag{3.6}$$

where ϕ is a function that is increasing and vanishes only at 0.

Characterization of minimums in convex optimization

Supposing that the condition stated in theorem 1 are met, we can define the properties that characterize minimums for convex functions defined on a convex set. We now state a central theorem in convex optimization:

Theorem 2. *For $U \subset V$ a convex space and $f : U \subset V \to \mathcal{R}$ a proper l.s.c. convex function.*

1. *If f has a local minimum at $\hat{u} \in U$, then it is a global minimum w.r.t U.*
2. *The set of minimizers of f w.r.t U is convex and closed.*
3. *If f is strictly convex, then f admits at most one minimum.*
4. *If f is also coercive or U bounded then he set of minimizers of f w.r.t U is non empty.*

We note that the theorem 2 makes use of all hypotheses of theorem 1 and adds the key hypothesis of convexity.

3.2.2 Problem of interest

As stated in the introduction chapter our goal is to optimize functions of the following prototype:

$$\arg\min_{x \in \mathcal{X}} \sum_{i \in \Omega} M_i(x_i) + R(Lx) \tag{3.7}$$

with:

- Ω the finite set $\{1, \ldots, n\}$ with $n \in \mathbb{N}^*$.
- $\mathcal{X}$ is a convex subset of $\mathbb{R}^n$.
- x is a vector of $\mathcal{X}$.
- x_i is the i^{th} element of x.

In this chapter, we assume the following hypotheses:

- All functions $M_i : x_i \in \mathbb{R}^d \to \mathbb{R}$ are (not necessarily smooth) proper l.s.c. convex functions over $\mathbb{R}^d$ with $d \in \mathbb{N}^+$.
- L is a continuous linear operator mapping space $\mathcal{X}$ to $\mathcal{Y}$.
- $R : y \in \mathcal{Y} \to \mathbb{R}$ is a (not necessarily smooth) proper l.s.c. convex function over $\mathcal{Y}$.

As a sum of proper l.s.c. convex functions, the function $\sum_{i\in\Omega} M_i(x_i) + R(Lx)$ is a proper l.s.c. convex function over $\mathcal{X}$. Hence, it admits at least one minimizer w.r.t. $\mathcal{X}$ 2, which is unique in the case of strict convexity. We call (3.7) the primal formulation of our problem.

3.2.3 From a primal to a primal dual form

Solving (3.7) presents two main challenges. First, if any of the $\{M_i\}$ and R functions is not smooth, the problem (3.7) is not a smooth optimization problem. Consequently, we need a technique that is not solely based on pure gradient descent. Secondly, the linear operator L and the function R generally couple the variables of x, i.e., elements of x interact one with another. This, again, makes the optimization harder.

However, by exploiting two brilliant yet simple ideas, primal dual optimization techniques overcome the stated challenges. For the lack of smoothness, we optimize a slightly different problem than (3.7) by operating on the Moreau envelope of $\sum_{i\in\Omega} M_i(x_i) + R(Lx)$. It turns out that the Moreau envelope of any convex function is always smooth and shares the exact same set of minimizers. For the coupling of variables, we proceed by computing the Fenchel Conjugate of R. This removes the coupling at the cost of adding a new set of variables named dual variables.

These two ideas, using the Moreau envelope and applying the Fenchel transformation, are instrumental in establishing a fast iterative optimization algorithm. They transform (3.7) into a saddle point smooth problem where each variable x_i can be optimized independently.

3.3 First order Primal Dual techniques

3.3.1 Smoothing: Moreau envelope and the proximal operator

Definition Given a proper lsc convex function $F : \mathcal{X} \subset V \to \mathbb{R}$, the Moreau envelope [126, 127] constructs a smoothed version of function F where the degree of smoothing is controlled by a parameter $\tau \in \mathbb{R}^+$. The Moreau envelope is defined as the unique solution of the optimization problem:

$$\mathcal{M}_{F,\tau}(\tilde{x}) = \min_{x \in \mathcal{X}} F(x) + \frac{1}{2\tau}\|x - \tilde{x}\|_2^2. \tag{3.8}$$

We note that this problem is smooth and it admits a unique minimizer due to the added smoothing ℓ_2 norm term. Moreover, the domain of function $\mathcal{M}_{F,\tau}$ is V independently of the initial domain $\mathcal{X}$ of function F.

At this stage, it is important to remind that a strictly convex optimization problem can still be very hard to solve. As a rule of thumb, if F is a difficult function to optimize, its Moreau envelope might still be challenging to compute. However, for a lot of interesting functions such as the ℓ_1 or Huber norm, one can derive close form formulas for the Moreau envelope. Figure 3.1 illustrates the Moreau envelope of a non smooth function, and Figure 3.2 the inner optimization problem solved during the Moreau envelope computation.

Associated to the Moreau envelope is the proximal operator that returns its unique minimizer:

$$\text{prox}_{\tau F}(\tilde{x}) = \arg\min_{x \in \mathcal{X}} F(x) + \frac{1}{2\tau}\|x - \tilde{x}\|_2^2. \tag{3.9}$$

We also make use of the proximal function:

$$P_{\tau F}(\tilde{x}, x) = F(x) + \frac{1}{2\tau}\|x - \tilde{x}\|_2^2. \tag{3.10}$$

Some properties of the Moreau envelope From figure 3.1 we see that optimizing F or its Moreau envelope leads to the same minimizer. This is a critical property of the Moreau envelope. One can easily prove the following properties [110] for any proper lsc convex function F, for any smoothing factors $\tau \in \mathbb{R}^+$ and for any point $\tilde{x} \in \mathcal{X}$:

$$\min_{x \in \mathcal{X}} F(x) \leqslant \mathcal{M}_{f,\tau}(\tilde{x}) \tag{3.11}$$

$$\mathcal{M}_{f,\tau}(\tilde{x}) \leqslant F(\tilde{x}) \tag{3.12}$$

$$\mathcal{M}_{f,\tau}(\tilde{x}) = F(\tilde{x}) \Longleftrightarrow \tilde{x} = \arg\min_{x \in \mathcal{X}} F(x) \tag{3.13}$$

Hence, minimizing the Moreau envelope of the function F provides an enticing alternative to the direct optimization of F.

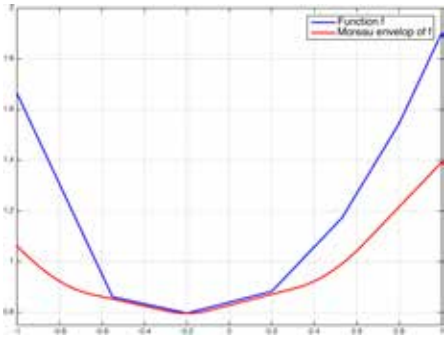


Figure 3.1 – Function F and its associated Moreau envelope $\mathcal{M}_{\tau F}$ for $\tau = 1$

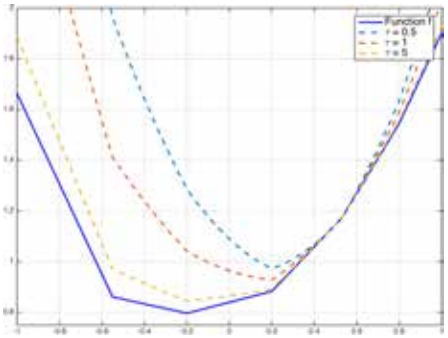


Figure 3.2 – Function F and $F(x) + 1/(2\tau)\|x - \tilde{x}\|_2^2$ with $\tilde{x} = 0.5$ and different τ.

Fixed point of the proximal operator One can demonstrate that by iteratively composing [141, 145] the proximal operator, we converge to a fixed point that is a minimizer of the function F. The reasoning follows these lines:

- Equation (3.12) states that iteratively applying the proximal operator leads to a sequence of points of decreasing value.
- Equation (3.11) provides a lower bound for the sequence. This guarantees the convergence of the sequence to a fixed point.
- Finally, (3.13) states that if a stationary point of $\mathcal{X}$ is found, it is a minimizer of F.

Hence, the proximal operator provides a very powerful method to optimize F. Computing the proximal operator associated to F remains the only difficulty.

Some properties of the proximal operator In the subsequent paragraphs, we introduce some other important properties of proximal operators. The curious reader can find proofs in [141]. We make use of the following notations:

- $G : \mathbb{R} \to \mathbb{R}$ is a mono-dimensional proper convex function.
- a a vector of $\mathbb{R}^d$.
- α and β are real scalars,
- λ is a strictly positive scalar.
- x is a multi-dimensional variable of $\mathbb{R}^d$.
- y is a mono-dimensional variable of $\mathbb{R}$.

Postcomposition If $x \in \mathbb{R}$ and $F(x) = \alpha G(x) + \beta$, then:

$$\text{prox}_{\tau F}(\tilde{x}) = \text{prox}_{\alpha\tau G}(\tilde{x}) \tag{3.14}$$

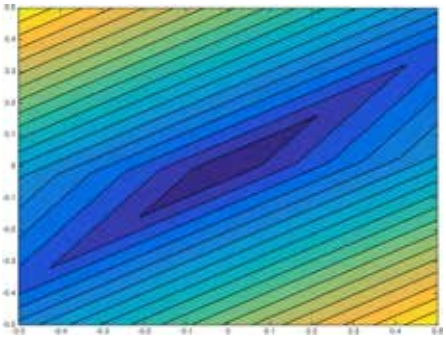


Figure 3.3 – Level set of a non smooth function. Alternating optimization methods ala Gauss Seidel can get stuck in acute corners.

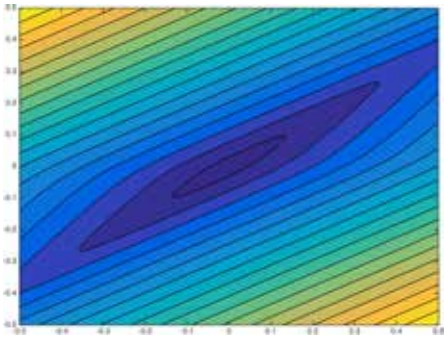


Figure 3.4 – Smoothing of previous level sets. The corners have been removed but the level sets are impacted.

Affine addition If $x \in \mathbb{R}$ and $F(x) = G(x) + \alpha x + \beta$, then:

$$\text{prox}_{\tau F}(\tilde{x}) = \text{prox}_{\tau G}(\tilde{x} - \tau\alpha) \tag{3.15}$$

This property is very handy for the primal dual algorithm.

Affine reduction If $F(x) = G(a^T x + \beta)$, then:

$$\text{prox}_{\tau F}(\tilde{x}) = \text{prox}_{\tilde{\tau} G}(a^T \tilde{x} + \beta)a \tag{3.16}$$

with $\tilde{\tau} = \tau / a^T a$.

This property is instrumental in computing the proximal operator for 2D and 3D registration problems.

Sum of functions The proximal operator is not a linear operator. Consequently, calculating the proximal operator of the sum of functions is generally computationally expensive even if the proximal operator of each function composing the sum is simple. However, in the latter case, one can take advantage of the ADMM technique to compute the proximal operator by formulating the optimization as a consensus problem [14]

3.3.2 Decoupling: Fenchel transform

The coupling issue Another difficulty of our optimization problem (3.7) is the coupling of variables (x_i) due to the linear operator L and the function R. Combined to the eventual non smoothness of R, this imposes to optimize all variables x_i jointly. The figure (3.3) illustrates the problem.

One solution is to smooth the objective function R to make corners disappear. However, this changes the objective function as seen in figure 3.4 which might

not be desirable. Using the Moreau envelope is generally computationally very intensive in general because, as seen in Section 3.3.1, the proximal operator is not an additive function. The first order primal dual technique relies on the Fenchel transformation of [51] to decouple the variables. This transformation comes with the cost of adding new variables, named dual variables, to the optimization problem (3.7). The Fenchel transform augments (3.7) to a saddle point optimization problem, i.e., a convex minimization problem for the primal and a concave maximization problem for the dual variables.

Definition The Fenchel transform [50] also named convex conjugate of a function R computes:

$$R^*(y) = \sup_x \langle x, y \rangle - R(x) \tag{3.17}$$

This transformation encodes the convex hull of the function R epigraph as a set of hyperplanes. It is worth noting that the Fenchel transform always yields a convex function. The figure 3.5 illustrate the Fenchel transform.

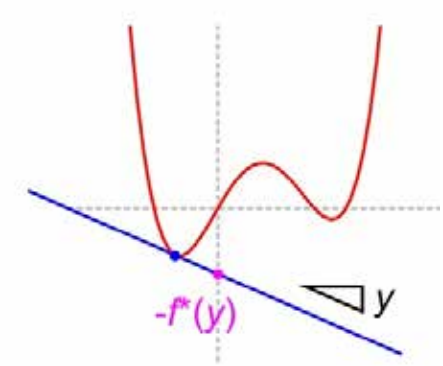


Figure 3.5 – Illustration of the Fenchel transform. The function $R(.)$ is given by the red curve. The blue line is the tangent of function $R(.)$ for the point represented by the blue dot. The intersection of tangent with the vetical axis reprensented by the pink dot gives the opposite value of the Fenchel transform.

Applying twice the Fenchel transform, i.e., the Fenchel transform of the Fenchel transform, computes the bi-convex conjugate:

$$R^{**}(x) = \sup_y \langle x, y \rangle - R^*(y) \tag{3.18}$$

If R is a convex function, one can prove that $R^{**} = R$. This simplifies (3.18) to:

$$R(x) = \sup_y \langle x, y \rangle - R^*(y) \tag{3.19}$$

If R^* is simple enough to compute, i.e., a close form exists, then the transformation gives an efficient approach to decouple the variable x in our context.

Some properties of the Fenchel transform We use the same notation as in paragraph 3.3.1, i.e.:

- $G : \mathbb{R} \to \mathbb{R}$ is a proper convex function.
- α and β are real scalars.
- x and y are mono-dimensional variables of $\mathbb{R}$.

Affine reduction If $F(x) = G(\alpha x + \beta)$, then:

$$F^*(y) = -\frac{\beta}{\alpha} y + G^*\left(\frac{y}{\alpha}\right) \tag{3.20}$$

Affine addition If $F(x) = G(x) + \alpha x + \beta$, then:

$$F^*(y) = -\beta + G^*(y - \alpha) \tag{3.21}$$

This property is very handy for the primal dual algorithm.

Postcomposition If $F(x) = \alpha G(x) + \beta$, then:

$$F^*(y) = -\beta + \alpha G^*\left(\frac{y}{\alpha}\right) \tag{3.22}$$

3.3.3 Primal Dual algorithm

To a primal dual solver

Now that we are equipped with the proximal operator and the Fenchel transform we can gradually modify our original optimization problem (3.7) to a more pleasant problem [25, 27, 177]. First, we make use of the Moreau envelope and we substitute to the functions $M_i(x_i)$ their smoothed version:

$$\arg\inf_{x \in \mathcal{X}} \sum_{i \in \Omega} M_i(x_i) + \frac{1}{2\tau} \|x_i - \tilde{x}_i\|_2^2 + R(Lx). \tag{3.23}$$

Then, we apply the Fenchel transform to R:

$$\arg\inf_{x \in \mathcal{X}} \sup_{y \in \mathcal{Z}} \sum_{i \in \Omega} M_i(x_i) + \frac{1}{2\tau} \|x_i - \tilde{x}_i\|_2^2 + \langle y, Lx \rangle - R^*(y). \tag{3.24}$$

At this stage the x_i are totally decoupled and their optimization is achieved by computing a proximal operator. However, the function R^* might not be smooth. Hence, we use the Moreau envelope again, but this time on R^*. We get:

$$\arg\inf_{x \in \mathcal{X}} \sup_{y \in \mathcal{Z}} \sum_{i \in \Omega} M_i(x_i) + \frac{1}{2\tau} \|x_i - \tilde{x}_i\|_2^2 + \langle y, Lx \rangle - R^*(y) - \frac{1}{2\sigma} \|y - \tilde{y}\|_2^2. \tag{3.25}$$

Writing x and y as time series $(x^n)_n$ and $(y^n)_n$, we obtain for each time step n:

$$\arg\inf_{x^n\in\mathcal{X}}\sup_{y^n\in\mathcal{Y}}\sum_{i\in\Omega}M_i(x_i^n)+\frac{1}{2\tau}\|x_i^n-x_i^{n-1}\|_2^2+\langle y^n,Lx^n\rangle-R^*(y^n)-\frac{1}{2\sigma}\|y^n-y^{n-1}\|_2^2. \tag{3.26}$$

For all time steps, the problem (3.24) is convex with respect to x^n and concave with respect to y^n. The two series converge to a fixed saddle point. Moreover, we can carry the optimization by alternating on x^n and y^n since the functional is now smooth.

One last trick, that we do not develop in this document is the smoothing of the sequence x^n. When optimizing with respect to y^n, we substitute x^n by $\tilde{x}^n = x^n + \theta(x^n - x^{n-1})$ with $\theta \in [0, 1]$. This trick greatly improves the convergence rate in general.

Hypothesis for convergence and convergence rate

As the first order primal dual optimization algorithm is iterative, we need to pay a particular attention to the values of τ and σ to ensure its convergence [27]. As illustrated by figure 3.1, τ and σ control the step size of primal and dual variables updates. While smaller steps slow down the convergence, too large steps make the algorithm unstable, leading to a possible divergence of the sequences (x^n) and (y^n). We notice that at each iteration x^n and y^n exchange information through an interface expressed as the linear operator L. Hence, the nature of L plays a critical role on the range of τ and σ.

In our context, the linear operator L is generally the weighted gradient (possibly oriented and non local) or the laplacian operator. Hence, the value of $\|L\|$ can easily be precomputed or closely upper-bounded.

The Algorithm 1 can be slightly modified to make use of strong convexity to achieve $\mathcal{O}(1/N^2)$ and even $\mathcal{O}(1/e^N)$ convergence rate. We refer the reader to Algorithms 2 and 3 in [27] for more details.

Algorithm description for non smooth problems

When neither the sum of $(M_i)_i$ nor R is smooth, we make use of the general first order primal dual algorithm:

It was proved in [27] that to guarantee convergence τ and σ have to satisfy the following inequality:

$$\tau\sigma\|L\| < 1 \tag{3.30}$$

with:

$$\|L\| = \max\{\|Lx\| : x \in \mathcal{X} \text{ with } \|x\| \leqslant 1\} \tag{3.31}$$

It has been demonstrated in [27] that the Algorithm 1 converges to an optimal primal-dual saddle point in $\mathcal{O}(1/N)$. This convergence rate is optimal for first order algorithm and non smooth function [131].

Algorithm 1: First order primal dual algorithm

Data: Inputs: $(M_i)_i$, R, L
Result: x
Compute the Fenchel transform: $\rightarrow R^*$.
Initialize primal and dual variable $\rightarrow x^0 = 0$, $y^0 = 0$.
Set $\tilde{x} = x^0$
while *Stopping criterion is not verified* **do**
Optimize the dual variables:

$$y^{n+1} = \text{prox}_{\sigma(\langle y, L\tilde{x}\rangle - R^*(y^n))}(y^n) \tag{3.27}$$

Optimize the primal variables:

$$x^{n+1} = \text{prox}_{\tau\left(\sum_{i\in\Omega} M_i(x_i) + \langle L^T y^n, x\rangle\right)}(x^n) \tag{3.28}$$

Smooth variable:

$$\tilde{x} = x^{n+1} + \theta\left(x^{n+1} - x^n\right) \tag{3.29}$$

Algorithm description for half-uniformly convex problems

If the sum of $(M_i)_i$ or R^* is uniformly convex, we can make use of an accelerated algorithm. For simplicity we assume that the sum of $(M_i)_i$ is γ-uniformly convex function.

It was proved in [27] that to guarantee convergence τ_0 and σ_0 have to satisfy the following inequality:

$$\tau_0\sigma_0\|L\| < 1 \tag{3.36}$$

It has been demonstrated in [27] that Algorithm 2 converges to an optimal primal-dual saddle point in $\mathcal{O}(1/N^2)$. This convergence rate is optimal for first order algorithm and uniformly convex problem with respect to either the primal or dual variables [131].

Algorithm description for uniformly convex problems

It was proved in [27] that Algorithm 3 converges if μ satisfies the following inequality:

$$\mu \leqslant 2\frac{\sqrt{\gamma\delta}}{\|L\|} \tag{3.41}$$

It has been demonstrated in [27] that Algorithm 3 converges to an optimal primal-dual saddle point in $\mathcal{O}(1/e^N)$. This convergence rate is optimal for first order algorithm and uniformly convex problem [131].

Algorithm 2: First order primal dual algorithm for half smooth problem

Data: Inputs: $(M_i)_i$, R, L, τ, σ
Result: x
Compute the Fenchel transform: $\rightarrow R^*$.
Initialize primal and dual variable $\rightarrow x^0 = 0$, $y^0 = 0$.
Set $\tilde{x} = x^0$
while *Stopping criterion is not verified* **do**

Optimize the dual variables:

$$y^{n+1} = \text{prox}_{\sigma_n(\langle y, L\tilde{x}\rangle - R^*(y^n))}(y^n) \tag{3.32}$$

Optimize the primal variables:

$$x^{n+1} = \text{prox}_{\tau_n\left(\sum_{i\in\Omega} M_i(x_i) + \langle L^T y^n, x\rangle\right)}(x^n) \tag{3.33}$$

Update the smoothing and steps size parameters:

$$\theta_n = \frac{1}{\sqrt{1 + 2\gamma\tau_n}}, \quad \tau_{n+1} = \theta_n \tau_n, \quad \sigma_{n+1} = \frac{\sigma_n}{\theta_n} \tag{3.34}$$

Smooth variable:

$$\tilde{x} = x^{n+1} + \theta_n \left(x^{n+1} - x^n\right) \tag{3.35}$$

Algorithm 3: First order primal dual algorithm for half smooth problem

Data: Inputs: $(M_i)_i$, R, L, μ
Result: x
Compute the Fenchel transform: $\rightarrow R^*$.
Initialize primal and dual variable $\rightarrow x^0 = 0$, $y^0 = 0$.
Set $\tilde{x} = x^0$
Set step sizes and smoothing value:

$$\theta_n \in \left[\frac{1}{1+\mu}, 1\right], \quad \tau = \frac{\mu}{2\gamma}, \quad \sigma = \frac{\mu}{2\delta} \tag{3.37}$$

while *Stopping criterion is not verified* **do**

Optimize the dual variables:

$$y^{n+1} = \text{prox}_{\sigma_n(\langle y, L\tilde{x}\rangle - R^*(y^n))}(y^n) \tag{3.38}$$

Optimize the primal variables:

$$x^{n+1} = \text{prox}_{\tau_n\left(\sum_{i\in\Omega} M_i(x_i) + \langle L^T y^n, x\rangle\right)}(x^n) \tag{3.39}$$

Smooth variable:

$$\tilde{x} = x^{n+1} + \theta_n\left(x^{n+1} - x^n\right) \tag{3.40}$$

3.3.4 Conditioning and Auto tuning of step sizes

Improving conditioning

The convergence speed of Algorithm 1 is tightly linked to the conditioning of the linear operator L since its norm impacts the step size parameters τ and σ. We can always make one of τ or σ larger, but this comes at the expense of reducing the other. Hence, improving the conditioning of L is critical.

Fortunately, conditioning improvement has been studied for a long time and successfully applied to primal dual problems [143, 162]. In our context, we only investigate the simple case where the operator L can be written as a diagonal matrix D with strictly positive elements and a well conditioned linear operator G:

$$L = DG \tag{3.42}$$

Here, the diagonal matrix D deteriorates the condition number of G, leading to the badly conditioned operator L. However, by applying a simple change of variable in the functional to optimize we can recover a well conditioned problem. By defining, $y = D^{-1}z$, we transform (3.26) to:

$$\begin{aligned} \arg \min_{x^n \in \mathcal{X}} \max_{z^n \in D\mathcal{Y}} & \sum_{i \in \Omega} M_i(x_i^n) + \frac{1}{2\tau}\|x_i^n - x_i^{n-1}\|_2^2 + \langle z^n, Gx^n \rangle \\ & - R^*(D^{-1}z^n) - \frac{1}{2\sigma}\|D^{-1}(z^n - z^{n-1})\|_2^2. \end{aligned} \tag{3.43}$$

Hence, we can now have larger step sizes τ and σ. Moreover, the overall problem retains the same complexity since the computation of proximal operators are not made more complicated by linear scaling.

If necessary, the same transformation can be applied to primal variables x. We refer the curious reader to [143].

Auto tuning of step sizes and stopping criterion

At this stage, defining correctly the step sizes τ and σ remains of critical importance to obtain a fast convergence. As a rule of thumb, we want the algorithm to make as much progress in the primal space as it does in the dual space. We can measure such progress by tracking the primal and dual residuals. If the primal residual is large compared to the dual residual, then we want to make more progress in the primal space at the next iteration. To this end, we need to increase τ. An analogue reasoning stands if the dual residual is larger than the primal residual. If both residuals are roughly equally large, then we are on course and no update of τ or σ is necessary.

Hence, we now make use of sequences $(\tau^n)_n$ and $(\sigma^n)_n$ to describe the evolution of τ and σ through time. Moreover, it has been demonstrated in [65] that the convergence of algorithm 1 is achieved only if the sequences $(\tau^n)_n$ and

$(\sigma^n)_n$ also converge. To this end, we introduce another sequence $(\alpha^n)_n$ that strictly decreases over time and controls how much $(\tau^n)_n$ and $(\sigma^n)_n$ are updated.

The primal residual of (3.26) is defined as:

$$p^{n+1} = \frac{1}{\tau^n}(x^n - x^{n+1}) - L^T(y^n - y^{n+1}) \tag{3.44}$$

Similarly, the dual residual of (3.26) is defined as:

$$d^{n+1} = \frac{1}{\sigma^n}(y^n - y^{n+1}) - L(x^n - x^{n+1}) \tag{3.45}$$

Algorithm 4 describes the update rules for $(\tau^n)_n$ and $(\sigma^n)_n$.

Algorithm 4: Auto tuning of step sizes

Data: Inputs: p^{n+1}, d^{n+1}, τ^n, σ^n, α^n, η
Result: τ^{n+1}, σ^{n+1}, α^{n+1}
if $\rho\|p^{n+1}\| < \|d^{n+1}\|$ **then**
    We need to increase dual variables convergence speed:

$$\tau^{n+1} = \tau^n(1-\alpha^n), \quad \sigma^{n+1} = \frac{\sigma^n}{1-\alpha^n}, \quad \alpha^{n+1} = \eta\alpha^n \tag{3.46}$$

else if $\rho\|d^{n+1}\| < \|p^{n+1}\|$ **then**
    We need to increase primal variables convergence speed:

$$\tau^{n+1} = \frac{\tau^n}{1-\alpha^n}, \quad \sigma^{n+1} = \sigma^n(1-\alpha^n), \quad \alpha^{n+1} = \eta\alpha^n \tag{3.47}$$

else

$$\tau^{n+1} = \tau^n, \quad \sigma^{n+1} = \sigma^n, \quad \alpha^{n+1} = \alpha^n \tag{3.48}$$

With:

- ρ is a parameter in $[1, +\infty)$ controlling when a correction needs to be applied.
- η is a parameter in $[0, 1)$ enforcing the convergence of the sequences $(\alpha^n)_n$, $(\tau^n)_n$ and $(\sigma^n)_n$.

3.4 Reformulating ℓ_1 based functions, Proximal operators, and Fenchel transformations

Before diving in the computations of proximal operators and Fenchel transforms, we introduce an important reformulation of ℓ_1 based functions.

3.4.1 On reformulating ℓ_1 based functions

Reformulation formula

We assume we are given an ℓ_1 based function $f : \mathbb{R} \to \mathbb{R}$ parametrized by $(w^-, w^+) \in \mathbb{R} \times \mathbb{R}$ of the following prototype:

$$f(x) = w^- \max(-x,\, 0) + w^+ \max(x,\, 0). \tag{3.49}$$

We show that for any $a \in \mathbb{R}$ we can reformulate the equation as

$$f(x) = \left(w^- - a\right) \max(-x,\, 0) + \left(w^+ + a\right) \max(x,\, 0) - ax \tag{3.50}$$

The proof follows from:

$$ax = a\left(\min(x,\, 0) + \max(x,\, 0)\right), \tag{3.51}$$
$$ax = -a \max(-x,\, 0) + a \max(x,\, 0), \tag{3.52}$$
$$0 = -a \max(-x,\, 0) + a \max(x,\, 0) - ax. \tag{3.53}$$

By adding (3.53) to (3.49) and rearranging terms, we obtain (3.50), thus, completing the proof.

Some examples of reformulations

Using (3.50) we can transform $f(.)$ to a symmetric function plus a linear term by choosing:

$$a = \frac{w^- - w^+}{2} \tag{3.54}$$
$$f(x) = \frac{w^+ + w^-}{2} \max(-x,\, 0) + \frac{w^+ + w^-}{2} \max(x,\, 0) + \frac{w^+ - w^-}{2} x \tag{3.55}$$
$$= \frac{w^+ + w^-}{2} |x| + \frac{w^+ - w^-}{2} x \tag{3.56}$$

We can also transform $f(.)$ to half linear term and a linear term by choosing either:

$$a = w^- \tag{3.57}$$
$$f(x) = \left(w^- - w^-\right) \max(-x,\, 0) + \left(w^+ + w^-\right) \max(x,\, 0) - w^- x \tag{3.58}$$
$$= \left(w^+ + w^-\right) \max(x,\, 0) - w^- x \tag{3.59}$$

or:

$$a = -w^+ \tag{3.60}$$
$$f(x) = \left(w^- + w^+\right) \max(-x,\, 0) + \left(w^+ - w^+\right) \max(x,\, 0) + w^+ x \tag{3.61}$$
$$= \left(w^+ + w^-\right) \max(-x,\, 0) + w^+ x \tag{3.62}$$

We illustrate these three reformulations in figure 3.6.

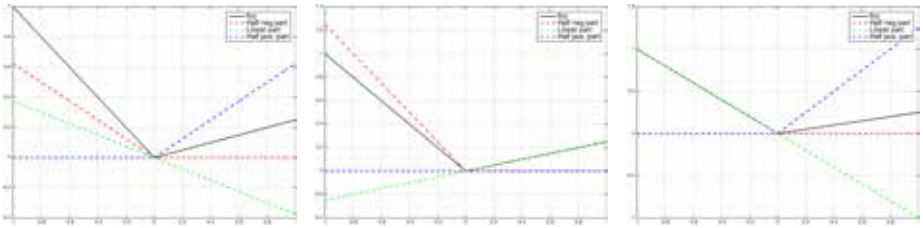


Figure 3.6 – Example of a reformulations of the same function: left is a symmetric reformulation, center is a half negative reformulation, and right is a half positive reformulation.

3.4.2 Proximal operators

We present here some useful proximal operators.

Asymmetric ℓ_1

The asymmetric ℓ_1 function illustrated in figure 3.7 has the following prototype:

$$F(x) = \begin{cases} -w^- x & \text{if } x < 0 \\ w^+ x & \text{if } x \geqslant 0 \end{cases} \tag{3.63}$$

$$= w^- \max(-x,\, 0) + w^+ \max(x,\, 0) \tag{3.64}$$

with $(w^-, w^+) \in \mathbb{R} \times \mathbb{R}$ and $w^- + w^+ \geqslant 0$ to ensure convexity.

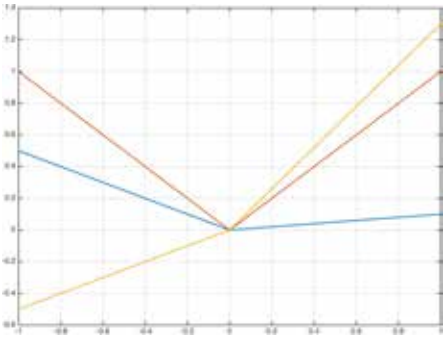


Figure 3.7 – Various instances of asymmetric ℓ_1 functions

The proximal operator associated to the asymmetric ℓ_1 function is:

$$\text{prox}_{F,\tau}(\tilde{x}) = \begin{cases} \tilde{x} + \tau w^- & \text{if} \quad \tilde{x} < -\tau w^- \\ 0 & \text{if} \quad \tilde{x} \in [-\tau w^-, \tau w^+] \\ \tilde{x} - \tau w^+ & \text{if} \quad \tilde{x} > +\tau w^+ \end{cases} \tag{3.65}$$

One can recognize in (3.65) a generalization of the *soft-thresholding* operator.

Sum of asymmetric ℓ_1

We now examine a function F, illustrated in figure 3.8, of the following prototype:

$$F(x) = \sum_{i=1}^{N} w_i^- \max(-x + b_i,\ 0) + w_i^+ \max(x - b_i,\ 0) \tag{3.66}$$

where each pair of $(w_i^-, w_i^+) \in \mathbb{R} \times \mathbb{R}$ verifies $w_i^- + w_i^+ \geqslant 0$ and each $b_i \in \mathbb{R}$.

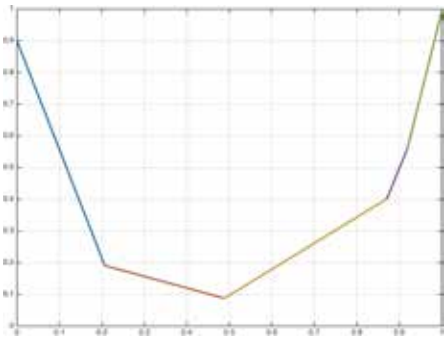


Figure 3.8 – The function F, a sum of asymmetric ℓ_1 functions. Each segments composing F is plotted in a different color.

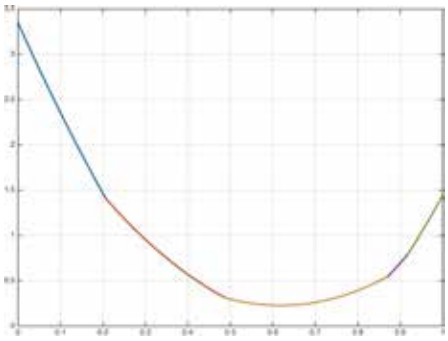


Figure 3.9 – The function P that needs to be minimized to obtain the proximal operator of function F with $\tilde{x} = 0.7$ and $\tau = 0.1$

Without any loss of generality we reformulate F as a sum of half linear terms using example from (3.59):

$$F(x) = c + ax + \sum_{i=1}^{N} w_i \max(x - b_i,\ 0) \tag{3.67}$$

with: $c = \sum_{i=1}^{N} w_i^- b_i,\ \ a = \sum_{i=1}^{N} w_i^-$ and, each $w_i = w_i^+ + w_i^-$

To compute the proximal operator we need to minimize the following function illustrated in figure 3.9:

$$P(x) = F(x) + \frac{1}{2\tau}(x - \tilde{x})^2 \tag{3.68}$$

$$= \sum_{i=1}^{N} w_i \max(x - b_i,\ 0) + \frac{1}{2\tau}\left(x - (\tilde{x} - \tau a)\right)^2 + \text{constant} \tag{3.69}$$

$$= \sum_{i=1}^{N} w_i \max(x - b_i,\ 0) + G(x) \tag{3.70}$$

with $\tau \in \mathbb{R}^+$ and $\tilde{x} \in \mathbb{R}$.

The function F is a piecewise linear function composed of $N+1$ segments as illustrated in figure 3.8. Since P is a convex function, its derivative is a monotonic increasing function. Hence, by identifying a segment that has a negative gradient at one hand and a positive derivative at the other hand we have found that an interval that contains the minimizer of P. Finally, over this interval the function P is purely quadratic. Hence, computing the minimum is then trivial. We describe this procedure in algorithm 5. One can obtain a simpler algorithm if the $\{b_i\}_{i\in\{1,\dots,N\}}$ are given sorted.

Asymmetric Huber

The asymmetric Huber illustrated in figure 3.10 naturally extends both Huber and $\ell 1$ norms, and it remains a convex function:

$$F(x) = \begin{cases} -a(x+\alpha/2) & \text{if } x < -\alpha \\ ax^2/(2\alpha) & \text{if } x \in [-\alpha, 0] \\ bx^2/(2\beta) & \text{if } x \in (0, \beta] \\ b(x-\beta/2) & \text{if } x > \beta \end{cases} \tag{3.72}$$

with α, β, a and b all positive real scalars.

The proximal operator associated to the asymmetric Huber function is:

$$\text{prox}_{F,\tau}(\tilde{x}) = \begin{cases} \tilde{x} + \tau a & \text{if} \quad \tilde{x} < -\alpha - \tau a \\ -\alpha\tilde{x}/(\alpha+\tau a) & \text{if} \quad \tilde{x} \in [-\alpha-\tau a, 0] \\ -\beta\tilde{x}/(\beta+\tau b) & \text{if} \quad \tilde{x} \in (0, \beta+\tau b] \\ \tilde{x} - \tau b & \text{if} \quad \tilde{x} > \beta + \tau b \end{cases} \tag{3.73}$$

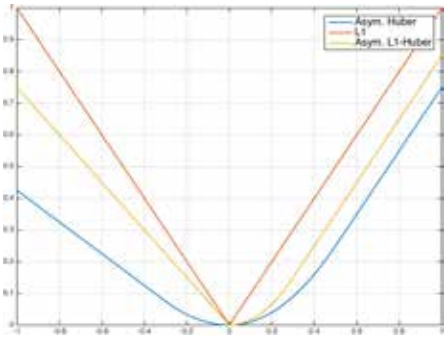


Figure 3.10 – Various instances of asymmetric Huber functions

Convex second order form

Another very interesting class of functions with simple proximal operator are the convex quadratic functions:

$$F(x) = x^T H x + a^T x + c, \tag{3.74}$$

Algorithm 5: Computation of proximal operator for piecewise linear functions

Data: Inputs: $\{b_i\}_{i\in\{1,...,N\}}$, $\{w_i\}_{i\in\{1,...,N\}}$, G, and $\tilde{x}$
Result: x^*

Initialize variables:
$x = \tilde{x}$,
$l = \min(dom(F)), \quad u = \min(dom(F))$,
$w_{cur} = 0, \quad \Sigma_w = 0.$
$B = \{b_i\}_i, \quad W = \{w_i\}_i,$

for $i \in \{1, ..., N\}$ **do**
  if $\Sigma_w + w_{cur} + \dfrac{dG}{dx}(u) > 0$ **then**
    Exit *for loop*
  else
    Get bounds of next segment:
      $l \leftarrow u$
      $j \leftarrow$ indice of minimun element of B
      $u \leftarrow B[j]$

    Update gradient:
      $\Sigma_w \leftarrow \Sigma_w + w_{cur}$
      $w_{cur} = W[j]$

    Update B and W:
      $B \leftarrow B - B[j]$
      $W \leftarrow W - W[j]$

if $i == N$ **then**
  Get bounds of last segment:
    $l \leftarrow u$
    $u \leftarrow +\infty$
    $\Sigma_w \leftarrow \Sigma_w + w_{cur}$

Find minimizer on found segment:

$$x^* = \underset{[l,u]}{\arg\min}\, \Sigma_w + G(x) \tag{3.71}$$

return x^*

with H a semi-definite positive matrix to ensure convexity.

Their proximal operator is given by:

$$\text{prox}_{\tau F}(\tilde{x}) = (I + \tau H)^{-1} (\tilde{x} - \tau a) \tag{3.75}$$

Note that evaluating the proximal operator relies on computing the inverse of $I + \tau H$. If H is diagonal, then evaluating this proximal operator is trivial. For others cases, we can rely on computing once and caching the inverse of operator $(I + \tau H)$.

Localization technique for complicated mono-dimensional variable

When F is a complicated function of a mono-dimensional variable, one can apply the localization technique [16] that iteratively reduces the search space of the minimum. This technique shares similarity with the bisection method but yields faster convergence by integrating topological information provided by the sub-gradients of the proximal function.

At each iteration, the localization technique defines a search direction by computing g, an element of the sub-derivative of the proximal function at current point x. Hence, any point situated in the opposite direction can be removed from the search space since the function to optimize is convex. Using the monotonicity of the sub-derivative of the proximal operator we can demonstrate that the minimum cannot be further to the current point x than $|\tau g|$. Hence, this gives us another bound to truncate the search space. Finally, we move x to the middle of the search space and iterate as illustrated in figure 3.11.

The range of the search space gives us an upper bound on the precision of x with respect to optimality. We can stop the algorithm iterations when a desired precision of ϵ has been reached. The algorithm 6 describes the localization procedure illustrated by figure 3.11.

This technique is a special case of the cutting plane algorithm. One can easily apply cutting plane to a multi-dimensional variable but the computational complexity becomes greatly increased.

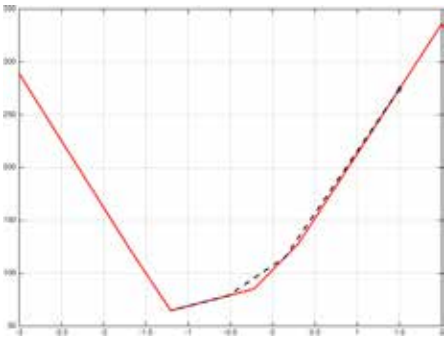


Figure 3.11 – The doted segments show the trajectory followed by x to compute the minimizer of $F(x)+1/(2\tau)\|x-\tilde{x}\|_2^2]$ plotted as a red curve.

Algorithm 6: Localization technique

Data: Inputs: F, $\tilde{x}$, τ and ϵ
Result: x
Initialize variables:

$$x = \tilde{x}, \quad l = \min(dom(F)), \quad u = \max(dom(F)) \tag{3.76}$$

while $u - l \geqslant \epsilon$ **do**
Compute an element of the sub-gradient of F at point x:

$$h \in \frac{\partial F}{\partial x} \tag{3.77}$$

Compute an element of the sub-gradient of the proximal function:

$$g = h + \frac{1}{\tau}(x - \tilde{x}) \tag{3.78}$$

if $g > 0$ **then**

$$l = \max(l, x - \tau g), \quad u = \min(u, x) \tag{3.79}$$

else if $g < 0$ **then**

$$l = \max(l, x), \quad u = \min(u, x - \tau g) \tag{3.80}$$

Update position of x:

$$x = \frac{l + u}{2} \tag{3.81}$$

3.4.3 Fenchel transform

We present here some useful Fenchel transform.

Asymmetric Huber

We remind the definition of the asymmetric Huber function:

$$F(x) = \begin{cases} -a(x+\alpha/2) & \text{if} \quad x < -\alpha \\ ax^2/(2\alpha) & \text{if} \quad x \in [-\alpha, 0] \\ bx^2/(2\beta) & \text{if} \quad x \in (0, \beta] \\ b(x-\beta/2) & \text{if} \quad x > \beta \end{cases} \tag{3.82}$$

Its Fenchel transform has a closed form solution:

$$F^*(y^*) = \begin{cases} \infty & \text{if} \quad y^* < -a \\ \alpha\|y^*\|_2^2 & \text{if} \quad y^* \in [a, 0] \\ \beta\|y^*\|_2^2 & \text{if} \quad y^* \in [0, b] \\ \infty & \text{if} \quad y^* > b \end{cases} \tag{3.83}$$

It is interesting to note that the Fenchel transform of the asymmetric Huber function is strongly convex.

Convex second order form

$$F(x) = \frac{1}{2}x^T H x + a^T x + c, \tag{3.84}$$

with H a semi-definite positive matrix.

$$F^*(y) = -\frac{1}{2}(a+y)^T H^{-1}(a+y) + c \tag{3.85}$$

3.5 TV regularized problems

3.5.1 Notations

We now restrict the operator L to be pairwise weighted operator. Hence, we make use of a directed graph $\mathcal{G}$ and a set of weights $\{w_{ij}\}_{ij}$ to make notations more explicit. The graph $\mathcal{G} = [\mathcal{V}, \mathcal{E}]$ is composed of a finite set of vertices $\mathcal{V}$ and a finite set of directed edges $\mathcal{E}$. The edge (i, j) of $\mathcal{E}$ creates a connexion from a vertex $i \in \mathcal{V}$ to a vertex $j \in \mathcal{V}$. Moreover, each edge (i, j) of $\mathcal{E}$ is associated to a weight $w_{ij} \in \mathbb{R}^+$.

With these notations the problems of interest become:

$$\min_{x \in \mathcal{X}} \sum_{i \in \mathcal{V}} M_i(x_i) + \sum_{(i,j) \in \mathcal{E}} w_{ij} R\left(x_i - x_j\right). \tag{3.86}$$

3.5.2 Some classic TV regularized problems

We now present some classic TV regularized problems in their primal and dual forms.

TV-linear

The simplest non degenerate problem uses a linear function for each vertex term and an ℓ_1-norm as regularizer:

$$\begin{aligned} &x_i \in [-1, 1] \\ &M_i(x_i) = c_i x_i, \quad c_i \in \mathbb{R}, \quad \forall i \in \mathcal{V} \\ &R(x) = |x| \end{aligned}$$

The primal forms of the TV-linear problem is:

$$\min_{(x_i \in [-1,1])_i} \sum_{i \in \mathcal{V}} c_i x_i + \sum_{(i,j) \in \mathcal{E}} w_{ij} \left| x_i - x_j \right|. \tag{3.87}$$

Its dual form is:

$$\max_{(y_{ij} \in [-w_{ij}, w_{ij}])_{ij}} - \sum_{i \in \mathcal{V}} \left| c_i + \sum_{j \in \mathcal{E}(i,.)} y_{ij} - \sum_{j \in \mathcal{E}(.,i)} y_{ji} \right| \tag{3.88}$$

TV-ℓ_2: The general ROF model

The TV-ℓ_2 problem was introduced by Rudin,Osher,and Fatemi in [148] for image denoising. Hence, it is also known as the ROF model. We present a slightly more general form here with:

$$\begin{aligned} &x_i \in \mathbb{R} \\ &M_i(x_i) = \frac{a_i}{2} x_i^2 + c_i x_i, \quad a_i \in \mathbb{R}^{+,*}, \quad c_i \in \mathbb{R}, \quad \forall i \in \mathcal{V} \\ &R(x) = |x| \end{aligned}$$

The primal forms of the TV-ℓ_2 problem is:

$$\min_{(x_i \in \mathbb{R})_i} \sum_{i \in \mathcal{V}} \frac{a_i}{2} x_i^2 + c_i x_i + \sum_{(i,j) \in \mathcal{E}} w_{ij} \left| x_i - x_j \right|. \tag{3.89}$$

Its dual form is:

$$\max_{(y_{ij} \in [-w_{ij}, w_{ij}])_{ij}} - \sum_{i \in \mathcal{V}} \frac{1}{2a_i} \left(c_i + \sum_{j \in \mathcal{E}(i,.)} y_{ij} - \sum_{j \in \mathcal{E}(.,i)} y_{ji} \right)^2 \tag{3.90}$$

TV-ℓ_1

The famous TV-ℓ_1 improves the ROF model by making the vertices terms less sensitive to outliers:

$$\begin{aligned}
&x_i \in \mathbb{R} \\
&M_i(x_i) = a_i|x + c_i|, \quad a_i \in \mathbb{R}^{+,*}, \quad c_i \in \mathbb{R}, \quad \forall i \in \mathcal{V} \\
&R(x) = |x|
\end{aligned}$$

The primal form of the TV-ℓ_1 problem is:

$$\min_{(x_i \in \mathbb{R})_i} \quad \sum_{i \in \mathcal{V}} a_i |x_i + c_i| + \sum_{(i,j) \in \mathcal{E}} w_{ij} \, |x_i - x_j| \, . \tag{3.91}$$

Its dual form is (up to a constant term):

$$\begin{aligned}
\max_{(y_{ij} \in [-w_{ij}, w_{ij}])_{ij}} \quad & - \sum_{i \in \mathcal{V}} c_i \left(\sum_{j \in \mathcal{E}(i,.)} y_{ij} - \sum_{j \in \mathcal{E}(.,i)} y_{ji} \right) \\
\text{subject to.} \quad & \left| \sum_{j \in \mathcal{E}(i,.)} y_{ij} - \sum_{j \in \mathcal{E}(.,i)} y_{ji} \right| \leqslant a_i \quad \forall i \in \mathcal{V}
\end{aligned} \tag{3.92}$$

TV-Huber

The TV-Huber model smoothes the TV-ℓ_1 vertices and edges terms:

$$\begin{aligned}
&x_i \in \mathbb{R} \\
&M_i(x_i) = a_i|x + c_i|_\alpha, \quad a_i \in \mathbb{R}^{+,*}, \quad c_i \in \mathbb{R}, \quad \alpha \in \mathbb{R}^+, \quad \forall i \in \mathcal{V} \\
&R(x) = |x|_\beta \quad \beta \in \mathbb{R}^+
\end{aligned}$$

where $|.|_\beta$ is the symmetric Huber norm defined by (3.72) with smoothing parameter β.

The primal forms of the TV-Huber problem is:

$$\min_{(x_i \in \mathbb{R})_i} \quad \sum_{i \in \mathcal{V}} a_i |x_i + c_i|_\alpha + \sum_{(i,j) \in \mathcal{E}} w_{ij} \, |x_i - x_j|_\beta \, . \tag{3.93}$$

Its dual form is (up to a constant term):

$$\begin{aligned}
\max_{(y_{ij} \in [-w_{ij}, w_{ij}])_{ij}} \quad & - \frac{\alpha}{2} \sum_{i \in \mathcal{V}} \left(\frac{c_i}{\alpha} + \sum_{j \in \mathcal{E}(i,.)} y_{ij} - \sum_{j \in \mathcal{E}(.,i)} y_{ji} \right)^2 - \frac{\beta}{2} \sum_{(i,j) \in \mathcal{E}} y_{ij}^2 \\
\text{subject to.} \quad & \left| \sum_{j \in \mathcal{E}(i,.)} y_{ij} - \sum_{j \in \mathcal{E}(.,i)} y_{ji} \right| \leqslant a_i \quad \forall i \in \mathcal{V}
\end{aligned} \tag{3.94}$$

3.5.3 Truncation theorem for convex TV regularized problems

Theorem 3. *Truncation theorem for TV regularized problems*
We suppose we have solved the following problem:

$$(z_i^*)_i = \underset{(z_i \in [\alpha,\beta])_i}{\arg\min} \quad \sum_{i \in \mathcal{V}} f_i(z_i) + \sum_{(i,j)\in\mathcal{E}} w_{ij} \max\left(z_i - z_j, 0\right). \tag{3.95}$$

where:

- $\alpha \in -\infty \cup \mathbb{R}$, $\beta \in \mathbb{R} \cup +\infty$, *and* $\alpha \leqslant \beta$,
- *The function* $f_i : \mathbb{R} \to \mathbb{R}$ *is a proper, l.s.c convex function* $\forall i \in \mathcal{V}$.

For all $[a, b] \subseteq [\alpha, \beta]$, *the following problem shares a special relationship with* (3.95)*:*

$$z_i^t = \underset{(z_i \in [a,b])_i}{\arg\min} \quad \sum_{i \in \mathcal{V}} f_i(z_i) + \sum_{(i,j)\in\mathcal{E}} w_{ij} \max\left(z_i - z_j, 0\right). \tag{3.96}$$

Indeed, a solution of (3.96) *is simply the truncation of a solution of* (3.95)*:*

$$z_i^t = \left([z_i^*]_{[a,b]}\right)_i, \quad \forall i \in \mathcal{V} \tag{3.97}$$

where $[.]_{[a,b]}$ *is the truncation operator:*

$$[x]_{[a,b]} = \begin{cases} a & if \quad x \leqslant a \\ x & if \quad x \in [a,b] \\ b & if \quad x \geqslant b \end{cases} \tag{3.98}$$

Lemma 1. *Let* $(y_{ij}^*)_{ij}$ *the optimal dual variables of the dual problem associated to* (3.95). *Then,* $(y_{ij}^*)_{ij}$ *are also optimal for the dual problem associated to* (3.96).

Proof

The demonstration relies on the proof that a primal dual fixed point of equation (3.96) is obtained from a primal-dual fixed point of equation (3.95).

We first express the primal dual problems of equation (3.95):

$$(z_i^*)_i, (y_{ij}^*)_{ij} = \arg \min_{(z_i \in [\alpha,\beta])_i} \max_{(y_{ij} \in [0, w_{ij}])_{ij}} \sum_{i \in \mathcal{V}} f_i(z_i) + \sum_{(i,j)\in\mathcal{E}} y_{ij} \left(z_i - z_j\right). \tag{3.99}$$

and equation (3.96):

$$(z_i^t)_i, (y_{ij}^t)_{ij} = \arg \min_{(z_i \in [a,b])_i} \max_{(y_{ij} \in [0, w_{ij}])_{ij}} \sum_{i \in \mathcal{V}} f_i(z_i) + \sum_{(i,j)\in\mathcal{E}} y_{ij} \left(z_i - z_j\right). \tag{3.100}$$

We initialize the variables of problem (3.100) to:

$$\begin{aligned} z_i &= [z_i^*]_{[a,b]}, \qquad \forall i \in \mathcal{V} \\ y_{ij} &= y_{ij}^* \qquad \forall (i,j) \in \mathcal{E} \end{aligned} \tag{3.101}$$

We show that the optimal dual variables of (3.99) are also optimal dual variables for (3.100). Using the dual update rule of the primal-dual algorithm 1 we have $\forall (i,j) \in \mathcal{E}$:

$$y_{ij}^n = \left[y_{ij}^* + \sigma \left([z_i^*]_{[a,b]} - [z_j^*]_{[a,b]} \right) \right]_{[0,w_{ij}]} \tag{3.102}$$

Supposing that $z_i^* - z_j^* > 0$ we have:

$$\begin{aligned} [z_i^*]_{[a,b]} - [z_j^*]_{[a,b]} &\geqslant 0 \\ y_{ij}^* &= w_{ij} \end{aligned} \tag{3.103}$$

Hence, we obtain:

$$\begin{aligned} y_{ij}^n &= w_{ij} \\ &= y_{ij}^* \end{aligned} \tag{3.104}$$

The same reasoning holds for $z_i^* - z_j^* < 0$:

$$\begin{aligned} y_{ij}^n &= 0 \\ &= y_{ij}^* \end{aligned} \tag{3.105}$$

Finally, if $z_i^* - z_j^* = 0$, we trivially obtain:

$$y_{ij}^n = y_{ij}^* \tag{3.106}$$

Hence, the dual variables $(y_{ij}^*)_{ij}$ are fixed points of the equation (3.100).

We now look at the primal variables. Using the primal update rules of algorithm 1 we have:

$$\begin{aligned} z_i^n &= \underset{x_i \in [a,b]}{\arg\min}\, f_i(x_i) + x_i \left(\sum_j y_{ij}^* - y_{ji}^* \right) + \frac{1}{2\tau} \| [z_i^*]_{[a,b]} - x_i \|_2^2 \\ &= \left[\underset{x_i \in \mathbb{R}}{\arg\min}\, f_i(x_i) + x_i \left(\sum_j y_{ij}^* - y_{ji}^* \right) + \frac{1}{2\tau} \| [z_i^*]_{[a,b]} - x_i \|_2^2 \right]_{[a,b]} \\ &= [z_i^*]_{[a,b]} \end{aligned} \tag{3.107}$$

The second equality holds since we are optimizing a mono dimensional convex function. Hence, the primal variables $([z_i^*]_{[a,b]})_i$ are fixed points of the equation (3.100).

This completes the proof.

3.5.4 A hierarchy of optimal dual spaces for TV-ℓ_2

Theorem 4. *Let $\mathcal{Y}^*_{\ell_2}$ be the space of optimal solutions for equation:*

$$\max_{y_{ij}\in[w^-_{ij},w^+_{ij}]} -\sum_{i\in\mathcal{V}} \frac{1}{2a_i}\|c_i + \sum_{j\in\mathcal{E}(i,.)} f_{ij} - \sum_{j\in\mathcal{E}(.,i)} f_{ji}\|_2^2, \tag{3.108}$$

and let $\mathcal{Y}^*_\alpha$ be the space of optimal solutions for equation:

$$\max_{y_{ij}\in[w^-_{ij},w^+_{ij}]} -\sum_{i\in\mathcal{V}} \frac{1}{2a_i}\left|c_i + \sum_{j\in\mathcal{E}(i,.)} f_{ij} - \sum_{j\in\mathcal{E}(.,i)} f_{ji}\right|_\alpha \tag{3.109}$$

Then, for any $(\alpha,\beta)\in\mathbb{R}^+\times\mathbb{R}^+$ and $\alpha\geqslant\beta$ we have:

$$\mathcal{Y}^*_{\ell_2}\subseteq\mathcal{Y}^*_\alpha\subseteq\mathcal{Y}^*_\beta\subseteq\mathcal{Y}^*_0=\mathcal{Y}^*_{\ell_1} \tag{3.110}$$

Proof

We recognize that (3.108) solves the dual of the general ROF model. The problem solved by (3.109) is simply the general ROF model with additional box constraints of the form $x_i\in[-\alpha,\alpha]$:

$$\min_{(x_i\in[-\alpha,\alpha])_i} \sum_{i\in\mathcal{V}}\frac{a_i}{2}x_i^2 + c_i x_i + \sum_{(i,j)\in\mathcal{E}} w_{ij}\left|x_i - x_j\right|.$$

$$\min_{(x_i\in[-\alpha,\alpha])_i}\ \max_{(y_{ij}\in[-w_{ij},w_{ij}])_{ij}}\ \sum_{i\in\mathcal{V}}\frac{a_i}{2}x_i^2 + x_i\left(c_i + \sum_{j\in\mathcal{E}(i,.)} y_{ij} - \sum_{j\in\mathcal{E}(.,i)} y_{ji}\right)$$

$$\max_{(y_{ij}\in[-w_{ij},w_{ij}])_{ij}} -\sum_{i\in\mathcal{V}}\frac{1}{2a_i}\left|c_i + \sum_{j\in\mathcal{E}(i,.)} y_{ij} - \sum_{j\in\mathcal{E}(.,i)} y_{ji}\right|_\alpha$$

To obtain the last equation, we simply recognize the equation of the Fenchel transform for a Huber function with smoothness α .

The proof is a simple application of the lemma of the truncation theorem. Indeed the lemma states that the optimal dual variables of the less constrained problem are also optimal for the more constrained problem. This concludes the proof.

3.5.5 Intersection of optimal dual space

Theorem 5. *Let $\mathcal{Y}^*_{\ell_2}$ be the space of optimal solutions for:*

$$\max_{y_{ij}\in[w^-_{ij},w^+_{ij}]} -\sum_{i\in\mathcal{V}} \frac{1}{2a_i}\|c_i + \sum_{j\in\mathcal{E}(i,.)} y_{ij} - \sum_{j\in\mathcal{E}(.,i)} y_{ji}\|_2^2 \tag{3.111}$$

and λ be a real valued scalar and $\mathcal{Y}^*_{\ell_1,\lambda}$ be the space of optimal solution for:

$$\max_{y_{ij}=[w_{ij}^-,w_{ij}^+]} -\sum_{i\in\mathcal{V}}\frac{1}{2a_i}\left|c_i+\lambda a_i+\sum_{j\in\mathcal{E}(i,.)}y_{ij}-\sum_{j\in\mathcal{E}(.,i)}y_{ji}\right| \tag{3.112}$$

Then

$$\mathcal{Y}^*_{\ell_2}=\bigcap_{\lambda\in\mathbb{R}}\mathcal{Y}^*_{\ell_1,\lambda} \tag{3.113}$$

First lemma

We demonstrate a first intermediate result:

Lemma 2. *Let α be a real valued scalar and $\mathcal{Y}^*_{\ell_2,\alpha}$ be the space of optimal solution for:*

$$\arg\max_{y_{ij}=[w_{ij}^-,w_{ij}^+]} -\sum_{i\in\mathcal{V}}\frac{1}{2a_i}\|c_i+\alpha a_i+\sum_{j\in\mathcal{E}(i,.)}f_{ij}-\sum_{j\in\mathcal{E}(.,i)}f_{ji}\|_2^2 \tag{3.114}$$

and β be a real valued scalar and $\mathcal{Y}^*_{\ell_2,\beta}$ be the space of optimal solution for:

$$\arg\max_{y_{ij}=[w_{ij}^-,w_{ij}^+]} -\sum_{i\in\mathcal{V}}\frac{1}{2a_i}\|c_i+\beta a_i+\sum_{j\in\mathcal{E}(i,.)}f_{ij}-\sum_{j\in\mathcal{E}(.,i)}f_{ji}\|_2^2 \tag{3.115}$$

Then,

$$\mathcal{Y}^*_{\ell_2,\alpha}=\mathcal{Y}^*_{\ell_2,\beta} \tag{3.116}$$

Proof of first Lemma

The proof of the previous lemma is straightforward:

$$\begin{aligned}
y^* &= \arg\max_{y_{ij}=[w_{ij}^-,w_{ij}^+]} -\sum_{i\in\mathcal{V}}\frac{1}{2a_i}\|c_i+\beta a_i+\sum_{j\in\mathcal{E}(i,.)}f_{ij}-\sum_{j\in\mathcal{E}(.,i)}f_{ji}\|_2^2 \\
&= \arg\max_{y_{ij}=[w_{ij}^-,w_{ij}^+]} -\sum_{i\in\mathcal{V}}\frac{1}{2a_i}\|c_i+\sum_{j\in\mathcal{E}(i,.)}f_{ij}-\sum_{j\in\mathcal{E}(.,i)}f_{ji}\|_2^2-\sum_{i\in\mathcal{V}}\beta\left(\sum_{j\in\mathcal{E}(i,.)}f_{ij}-\sum_{j\in\mathcal{E}(.,i)}f_{ji}\right) \\
&= \arg\max_{y_{ij}=[w_{ij}^-,w_{ij}^+]} -\sum_{i\in\mathcal{V}}\frac{1}{2a_i}\|c_i+\sum_{j\in\mathcal{E}(i,.)}f_{ij}-\sum_{j\in\mathcal{E}(.,i)}f_{ji}\|_2^2
\end{aligned}$$

Hence, for all couple of real scalars (α,β) we have: $\mathcal{Y}^*_{\ell_2,\alpha}=\mathcal{Y}^*_{\ell_2,0}$ and $\mathcal{Y}^*_{\ell_2,\beta}=\mathcal{Y}^*_{\ell_2,0}$. Therefore, we have $\mathcal{Y}^*_{\ell_2,\alpha}=\mathcal{Y}^*_{\ell_2,\beta}$. This concludes the proof of the lemma.

Second lemma

We demonstrate another intermediate result:

Lemma 3. *There exist a positive real scalar α such that:*

$$\bigcap_{\lambda\in\mathbb{R}} \mathcal{Y}^*_{\ell_1,\lambda} = \bigcap_{\lambda\in[-\alpha,\alpha]} \mathcal{Y}^*_{\ell_1,\lambda} \tag{3.117}$$

Proof of Lemma

Since, a is positive vector, c_i is fixed and y is bounded, we can find $\alpha \in \mathbb{R}^+$ such that for a y:

$$c_i + \alpha a_i + \sum_{j\in\mathcal{E}(i,.)} y_{ij} - \sum_{j\in\mathcal{E}(.,i)} y_{ji} \geqslant 0, \quad \forall i \in \mathcal{V} \tag{3.118}$$

and

$$c_i - \alpha a_i + \sum_{j\in\mathcal{E}(i,.)} y_{ij} - \sum_{j\in\mathcal{E}(.,i)} y_{ji} \leqslant 0, \quad \forall i \in \mathcal{V} \tag{3.119}$$

Hence, for any $\lambda \leqslant -\alpha$ or $\lambda \geqslant \alpha$ the solution space $\mathcal{Y}^*_{\ell_1,\lambda}$ is the space $\left[(y_{ij})_{ij} \ \middle| \ \forall (i,j) \in \mathcal{E}, \ \ y_{ij} \in \left[w^-_{ij}, w^+_{ij}\right]\right]$. Therefore, we have:

$$\bigcap_{\lambda\leqslant-\alpha} \mathcal{Y}^*_{\ell_1,\lambda} = \mathcal{Y}^*_{\ell_1,-\alpha} \quad \text{and} \quad \bigcap_{\lambda\geqslant\alpha} \mathcal{Y}^*_{\ell_1,\lambda} = \mathcal{Y}^*_{\ell_1,\alpha}, \tag{3.120}$$

from which we easily derive the lemma.

Proof of Theorem

We start by proving $\mathcal{Y}^*_{\ell_2} \subseteq \bigcap_{\lambda\in\mathbb{R}} \mathcal{Y}^*_{\ell_1,\lambda}$.

Thanks to theorem (4) and lemma (2) we have for all $\lambda \in \mathbb{R}$:

$$\mathcal{Y}^*_{\ell_2,\lambda} \subseteq \mathcal{Y}^*_{\ell_1,\lambda} \quad \Leftrightarrow \quad \mathcal{Y}^*_{\ell_2} \subseteq \mathcal{Y}^*_{\ell_1,\lambda} \tag{3.121}$$

Hence, we have $\mathcal{Y}^*_{\ell_2} \subseteq \bigcap_{\lambda\in\mathbb{R}} \mathcal{Y}^*_{\ell_1,\lambda}$.

Let us now prove $\bigcap_{\lambda\in\mathbb{R}} \mathcal{Y}^*_{\ell_1,\lambda} \subseteq \mathcal{Y}^*_{\ell_2}$

From the lemma (3) we get

$$\mathcal{Y}^*_{\ell_2} = \bigcap_{\lambda\in[-\alpha,\alpha]} \mathcal{Y}^*_{\ell_1,\lambda} \tag{3.122}$$

Let $y^* \in \bigcap_{\lambda\in[-\alpha,\alpha]} \mathcal{Y}^*_{\ell_1,\lambda}$. We have:

$$y^* = \underset{y_{ij}=[w^-_{ij},w^+_{ij}]}{\arg\max} -\int_{-\alpha}^{\alpha} \sum_{i\in\mathcal{V}} \frac{1}{2a_i} |c_i + \lambda a_i + \sum_{j\in\mathcal{E}(i,.)} f_{ij} - \sum_{j\in\mathcal{E}(.,i)} f_{ji}| d\lambda \tag{3.123}$$

$$= \underset{y_{ij}=[w^-_{ij},w^+_{ij}]}{\arg\max} -\sum_{i\in\mathcal{V}} \frac{1}{2a_i} \int_{-\alpha}^{\alpha} |c_i + \lambda a_i + \sum_{j\in\mathcal{E}(i,.)} f_{ij} - \sum_{j\in\mathcal{E}(.,i)} f_{ji}| d\lambda \tag{3.124}$$

We remind the anti-derivative formula for the absolute value:

$$\int_{\beta}^{\alpha} |\lambda| d\lambda = \frac{1}{2} \left(\alpha|\alpha| - \beta|\beta| \right) \tag{3.125}$$

Hence, we get:

$$\begin{aligned} y^* = \underset{y_{ij}=[w_{ij}^-, w_{ij}^+]}{\arg\max} - \sum_{i \in \mathcal{V}} \frac{1}{2a_i} & \left(c_i + \alpha a_i + \sum_{j \in \mathcal{E}(i,.)} f_{ij} - \sum_{j \in \mathcal{E}(.,i)} f_{ji} \right) \\ & \times \left| c_i + \alpha a_i + \sum_{j \in \mathcal{E}(i,.)} f_{ij} - \sum_{j \in \mathcal{E}(.,i)} f_{ji} \right| \\ & - \left(c_i - \alpha a_i + \sum_{j \in \mathcal{E}(i,.)} f_{ij} - \sum_{j \in \mathcal{E}(.,i)} f_{ji} \right) \\ & \times \left| c_i - \alpha a_i + \sum_{j \in \mathcal{E}(i,.)} f_{ij} - \sum_{j \in \mathcal{E}(.,i)} f_{ji} \right| \end{aligned}$$

But since we have chosen α large enough the last equation simplifies to:

$$\begin{aligned} y^* = \underset{y_{ij}=[w_{ij}^-, w_{ij}^+]}{\arg\max} - \sum_{i \in \mathcal{V}} \frac{1}{2a_i} & \left(c_i + \alpha a_i + \sum_{j \in \mathcal{E}(i,.)} f_{ij} - \sum_{j \in \mathcal{E}(.,i)} f_{ji} \right)^2 \\ & + \left(c_i - \alpha a_i + \sum_{j \in \mathcal{E}(i,.)} f_{ij} - \sum_{j \in \mathcal{E}(.,i)} f_{ji} \right)^2 \end{aligned}$$

Hence, we recognize that:

$$\begin{aligned} & y^* \in \mathcal{Y}^*_{\ell_2, -\alpha} \cap \mathcal{Y}^*_{\ell_2, \alpha} \\ & y^* \in \mathcal{Y}^*_{\ell_2} \end{aligned}$$

The last equation stands thanks to lemma (2). Therefore we have $\bigcap_{\lambda \in \mathbb{R}} \mathcal{Y}^*_{\ell_1, \lambda} \subseteq \mathcal{Y}^*_{\ell_2}$ which completes the proof.

3.5.6 A new primal-dual formulation of the ROF model

We now consider optimizing the ROF model. We propose a new primal dual formulation of the ROF model that we label L-ROF for short of linear ROF:

$$\min_{(x_i \in \mathbb{R})_i} \sum_{i \in \mathcal{V}} c_i x_i + \sum_{(i,j) \in \mathcal{E}} w_{ij} \left| x_i - x_j \right|. \tag{3.126}$$

The primal dual form of the L-ROF model is:

$$\min_{x_i \in \mathbb{R}} \max_{(y_{ij} \in [w_{ij}^-, w_{ij}^+])_{ij}} - \sum_{i \in \mathcal{V}} x_i \left(c_i + \sum_{j \in \mathcal{E}(i,.)} y_{ij} - \sum_{j \in \mathcal{E}(.,i)} y_{ji} \right) \tag{3.127}$$

The following theorem achieves the connexion with the ROF model:

Theorem 6. *Let $\mathcal{Y}^*_{\ell_2}$ be the space of optimal solution for the ROF model:*

$$\max_{\left(y_{ij}\in[w^-_{ij},w^+_{ij}]\right)_{ij}} -\sum_{i\in\mathcal{V}}\frac{1}{2}\left(c_i+\sum_{j\in\mathcal{E}(i,.)}y_{ij}-\sum_{j\in\mathcal{E}(.,i)}y_{ji}\right)^2$$

Let $\mathcal{Y}^*_{L-ROF}$ be the space of optimal solution for the L-ROF given by:

$$\min_{x_i\in\mathbb{R}}\ \max_{(y_{ij}\in[w^-_{ij},w^+_{ij}])_{ij}} -\sum_{i\in\mathcal{V}}x_i\left(c_i+\sum_{j\in\mathcal{E}(i,.)}y_{ij}-\sum_{j\in\mathcal{E}(.,i)}y_{ji}\right) \tag{3.128}$$

Then:

$$\mathcal{Y}^*_{\ell_2}=\mathcal{Y}^*_{L-ROF} \tag{3.129}$$

Proof: preliminary

First, we observe that both models share the same dual update rule:

$$y^{n+1}_{ij}=\left[y^n_{ij}+\sigma\left(x^n_i-x^n_j\right)\right]_{[w^-_{ij},w^+_{ij}]} \tag{3.130}$$

Furthermore, at optimality the primal and dual variables of the ROF model verify:

$$\begin{cases} y^{*,ROF}=w^-_{ij} & \text{if}\quad x^{*,ROF}_i\leqslant x^{*,ROF}_j \\ y^{*,ROF}\in\left]w^-_{ij},w^+_{ij}\right[& \text{if}\quad x^{*,ROF}_i= x^{*,ROF}_j \\ y^{*,ROF}=w^+_{ij} & \text{if}\quad x^{*,ROF}_i\geqslant x^{*,ROF}_j \end{cases} \tag{3.131}$$

Proof: $\mathcal{Y}^*_{\ell_2}\subseteq\mathcal{Y}^*_{L-ROF}$

Let's now assume that we have solved the ROF model. We use these variables as initialization for a primal and dual update of the L-ROF model:

$$\begin{aligned} x^{1,L-ROF}_i &= x^{*,ROF}_i-\tau\left(c_i+\sum_{j\in\mathcal{E}(i,.)}y^{*,ROF}_{ij}-\sum_{j\in\mathcal{E}(.,i)}y^{*,ROF}_{ji}\right)\\ &=(1+\tau)x^{*,ROF}_i \end{aligned}$$

and

$$\begin{aligned} y^{1,L-ROF}_{ij} &= \left[y^{*,ROF}_{ij}+\sigma\left(x^{1,L-ROF}_i-x^{1,L-ROF}_j\right)\right]_{[w^-_{ij},w^+_{ij}]}\\ &=\left[y^{*,ROF}_{ij}+\sigma(1+\tau)\left(x^{*,ROF}_i-x^{*,ROF}_j\right)\right]_{[w^-_{ij},w^+_{ij}]}\\ &=y^{*,ROF}_{ij} \end{aligned}$$

We obtain the last equation thanks to (3.131).
By induction we obtain:

$$x_i^{n,L-ROF} = (1+\tau)^n x_i^{*,ROF}$$

and

$$y_{ij}^{n,L-ROF} = y_{ij}^{*,ROF}$$

Hence, the dual optimal variables of the ROF model are also optimal for the L-ROF model.

Proof: $\mathcal{Y}^*_{L-ROF} \subseteq \mathcal{Y}^*_{\ell_2}$

Let us assume we have optimized the $L-ROF$ up to a point (k updates) where the dual variables are optimal (un-changed by any further updates). We use these dual variables as initialization for a primal and dual update of the ROF model:

$$\begin{aligned} x_i^{1,ROF} &= \frac{x_i^{k,L-ROF} - \tau\left(c_i + \sum_{j\in\mathcal{E}(i,.)} y_{ij}^{*,L-ROF} - \sum_{j\in\mathcal{E}(.,i)} y_{ji}^{*,L-ROF}\right)}{1+\tau} \\ &= \frac{x_i^{k+1,L-ROF}}{1+\tau} \end{aligned}$$

and

$$\begin{aligned} y_{ij}^{1,ROF} &= \left[y_{ij}^{*,L-ROF} + \sigma\left(x_i^{1,ROF} - x_j^{1,ROF}\right)\right]_{[w_{ij}^-, w_{ij}^+]} \\ &= \left[y_{ij}^{*,L-ROF} + \frac{\sigma}{(1+\tau)}\left(x_i^{k+1,L-ROF} - x_j^{k+1,L-ROF}\right)\right]_{[w_{ij}^-, w_{ij}^+]} \\ &= y_{ij}^{*,L-ROF} \end{aligned}$$

The last equation stands from the assumption that optimizing further the $L-ROF$ model does not modify its dual variables and from (3.131).
By induction we obtain:

$$x_i^{n,L-ROF} = \frac{x_i^{k+n,L-ROF}}{(1+\tau)^n}$$

and

$$y_{ij}^{n,L-ROF} = y_{ij}^{*,L-ROF}$$

Hence, the dual optimal variables of the L-ROF model are also optimal for the ROF model. This concludes the proof.

Properties of the L-ROF model

Property 1. Let us consider the global affine transforms of the following form: $z_i = ax_i + b$ for any $i \in \mathcal{V}$ and $(a, b) \in \mathbb{R}^{+,*} \times \mathbb{R}$. Then, the global affine transform lets the optimal dual space of the L-ROF model unchanged.

The proof follows from simple calculus using equation (3.127).

Property 2. Once the optimal dual variables of the L-ROF model have been computed then, the optimal primal variable of the ROF model can be obtained in linear time:

$$x_i^{*,ROF} = -\left(c_i + \sum_{j \in \mathcal{E}(i,.)} y_{ij}^{*,L-ROF} - \sum_{j \in \mathcal{E}(.,i)} y_{ji}^{*,L-ROF}\right)$$

The proof follows from the equality of optimal dual spaces given by Theorem 6 and from the optimal primal-dual relationship of the ROF model.

Primal scaling

To optimize the L-ROF model we propose to modify Algorithm 1 by introducing a linear scaling of the primal variables every k iterations. Thanks to Property 1 this has no impact on the dual variables.

The scaling guarantees the primal variables remain bounded. It also allows to give more weight to the current updates by progressively forgetting the past. One can think of it as a more gentle way to restart the smoothing of primal variables.

The L-ROF optimization is described in Algorithm 7.

3.5.7 Fused Lasso approximation on pairwise graph for various sparsifying strength

The fused lasso approximation problem

We now consider a direct application of the hierarchy of optimal dual spaces with the following problem:

$$\begin{aligned} &x_i \in \mathbb{R} \\ &M_i(x_i) = \frac{1}{2}x_i^2 + c_i x_i + \lambda |x_i|, \quad c_i \in \mathbb{R}, \quad \lambda \in \mathbb{R}^+, \quad \forall i \in \mathcal{V} \\ &R(x) = |x| \end{aligned} \tag{3.132}$$

The primal form of the fused lasso is:

$$\min_{(x_i \in \mathbb{R})_i} \sum_{i \in \mathcal{V}} \frac{1}{2}x_i^2 + c_i x_i + \lambda |x_i| + \sum_{(i,j) \in \mathcal{E}} w_{ij} \left|x_i - x_j\right|. \tag{3.133}$$

Algorithm 7: L-ROF primal dual optimization

Data: Inputs: $(c_i)_i$, $(w_{ij})_{ij}$, τ, σ, δ
Result: x
Initialize primal and dual variable $\rightarrow x^0 = 0$, $y^0 = 0$.
Set $\tilde{x} = x^0$ and $n = 0$
while $n \leqslant$ *max iteration* **do**
Optimize the dual variables:

$$y_{ij}^{n+1} = \left[y_{ij}^n + \sigma\left(\tilde{x}_i^n - \tilde{x}_j^n\right)\right]_{[-w_{ij}, w_{ij}]}, \quad \forall\,(i,j) \in \mathcal{E}$$

Optimize the primal variables:

$$x_i^{n+1} = x_i^n - \tau\left(c_i + \sum_{j \in \mathcal{E}(i,.)} y_{ij}^{n+1} - \sum_{j \in \mathcal{E}(.,i)} y_{ji}^{n+1}\right), \quad \forall i \in \mathcal{V}$$

Smooth variable:

$$\tilde{x}_i^{n+1} = x_i^{n+1} + \theta\left(x_i^{n+1} - x_i^n\right), \quad \forall i \in \mathcal{V}$$

Primal scaling:
if *mod*(n,k) *is* 0 **then**

$$x_i^{n+1} = \delta x_i^{n+1}$$
$$\tilde{x}_i^{n+1} = \delta \tilde{x}_i^{n+1}$$

Increment iteration counter: $n = n + 1$

An extended dual form is:

$$\max_{(y_{ij} \in [-w_{ij}, w_{ij}])_{ij}} \quad \max_{(z_i \in [-\lambda, \lambda])_i} \quad -\frac{1}{2}\sum_{i \in \mathcal{V}}\left(c_i + z_i + \sum_{j \in \mathcal{E}(i,.)} y_{ij} - \sum_{j \in \mathcal{E}(.,i)} y_{ji}\right)^2 \tag{3.134}$$

and

$$x_i^* = -\left(c_i + z_i^* + \sum_{j \in \mathcal{E}(i,.)} y_{ij}^* - \sum_{j \in \mathcal{E}(.,i)} y_{ji}^*\right) \tag{3.135}$$

Which simplifies to:

$$\max_{(y_{ij}\in[-w_{ij},w_{ij}])_{ij}} -\frac{1}{2}\sum_{i\in\mathcal{V}}\left(c_i+\sum_{j\in\mathcal{E}(i,.)}y_{ij}-\sum_{j\in\mathcal{E}(.,i)}y_{ji}\right)^2 + \left|c_i+\sum_{j\in\mathcal{E}(i,.)}y_{ij}-\sum_{j\in\mathcal{E}(.,i)}y_{ji}\right|_\lambda \tag{3.136}$$

Generalization of Friedmann Theorem

We generalize the theorem of Friedmann [57] to any pairwise graph:

Theorem 7. *Once the ROF model* (3.133)*, with* $\lambda = 0$*, has been solved, one can compute in linear time the solution of* (3.133) *for any* $\lambda \in \mathbb{R}^+$*.*

Proof

We notice that the dual problem of the fused lasso (3.136) is composed of ℓ_2 and huber terms. Hence, thanks to theorem (4), we know that the optimal dual variables of the ROF model are also optimal for the fused lasso (and for any λ).

Let suppose we dispose of $y^{*,ROF}$ by having solved the dual problem associated to the ROF model. We have to solve the remaining problem:

$$\min_{(x_i\in\mathbb{R})_i} \sum_{i\in\mathcal{V}}\frac{1}{2}x_i^2 + x_i\left(c_i+\sum_{j\in\mathcal{E}(i,.)}y_{ij}^{*,ROF}-\sum_{j\in\mathcal{E}(.,i)}y_{ji}^{*,ROF}\right)+\lambda|x_i|. \tag{3.137}$$

The problem (3.137) entirely decouples with respect to $(x_i)_i$. We obtain the solution of each subproblem by applying the soft-thresholding operator.

On the other hand, let us suppose we dispose of $x^{*,ROF}$ by having solved the primal problem associated to the ROF model. We will have a bit more work to do.

First, we obtain in linear time:

$$x_i^{*,ROF} = -\left(c_i+\sum_{j\in\mathcal{E}(i,.)}y_{ij}^{*,ROF}-\sum_{j\in\mathcal{E}(.,i)}y_{ji}^{*,ROF}\right) \tag{3.138}$$

We then introduce the Fenchel transform for the huber terms in (3.136):

$$\max_{(z_i\in[-\lambda,\lambda])_i} -\frac{1}{2}\sum_{i\in\mathcal{V}}\left(c_i+z_i+\sum_{j\in\mathcal{E}(i,.)}y_{ij}^{*,ROF}-\sum_{j\in\mathcal{E}(.,i)}y_{ji}^{*,ROF}\right)^2. \tag{3.139}$$

We obtain in linear time:

$$z_i^* = \left[-\left(c_i + \sum_{j \in \mathcal{E}(i,.)} y_{ij}^{*,ROF} - \sum_{j \in \mathcal{E}(.,i)} y_{ji}^{*,ROF} \right) \right]_{[-\lambda,\lambda]} \tag{3.140}$$

Hence, using equation (3.135) we obtain in linear time the optimal primal variables of (3.133). This concludes the proof.

3.5.8 ROF model with a Global regularization term

We now consider optimizing an ROF model with an additional global regularization term: The primal forms of the TV-ℓ_2 problem is:

$$\min_{(x_i \in \mathbb{R})_i} \sum_{i \in \mathcal{V}} \frac{1}{2} x_i^2 + c_i x_i + \sum_{(i,j) \in \mathcal{E}} w_{ij} \left| x_i - x_j \right| + G\left(\sum_{i \in \mathcal{V}} x_i \right). \tag{3.141}$$

where $G(.) : \mathbb{R} \to \mathbb{R}$ is a convex function possibly not smooth acting as a global regularizer.

Theorem 8. *A solution of problem* (3.141) *can be obtained in linear time from a solution of the associated ROF model if the global regularizer dual function* $G^*(.)$ *has a proximal operator which can be computed in linear time.*

Proof

The dual form of problem (3.141) is:

$$\max_{(y_{ij} \in [-w_{ij}, w_{ij}])_{ij}} \max_{z \in \mathbb{R}} -\frac{1}{2} \sum_{i \in \mathcal{V}} \left(c_i + z + \sum_{j \in \mathcal{E}(i,.)} y_{ij} - \sum_{j \in \mathcal{E}(.,i)} y_{ji} \right)^2 - G^*(z) \tag{3.142}$$

and the optimal primal variables verify:

$$x_i^* = -\left(c_i + z^* + \sum_{j \in \mathcal{E}(i,.)} y_{ij}^* - \sum_{j \in \mathcal{E}(.,i)} y_{ji}^* \right) \tag{3.143}$$

We develop the dual form to:

$$\max_{(y_{ij} \in [-w_{ij}, w_{ij}])_{ij}} \max_{z \in \mathbb{R}} -\frac{1}{2} \sum_{i \in \mathcal{V}} \left(c_i + \sum_{j \in \mathcal{E}(i,.)} y_{ij} - \sum_{j \in \mathcal{E}(.,i)} y_{ji} \right)^2 \tag{3.144}$$

$$+ \sum_{i \in \mathcal{V}} -\frac{1}{2} z^2 + z \left(c_i + \sum_{j \in \mathcal{E}(i,.)} y_{ij} - \sum_{j \in \mathcal{E}(.,i)} y_{ji} \right) - G^*(z) \tag{3.145}$$

Since we have:

$$\sum_{i\in\mathcal{V}}\left(\sum_{j\in\mathcal{E}(i,.)} y_{ij} - \sum_{j\in\mathcal{E}(.,i)} y_{ji}\right) = 0, \tag{3.146}$$

we can simplifies the dual to:

$$\max_{(y_{ij}\in[-w_{ij},w_{ij}])_{ij}} \max_{z\in\mathbb{R}} -\frac{1}{2}\sum_{i\in\mathcal{V}}\left(c_i + \sum_{j\in\mathcal{E}(i,.)} y_{ij} - \sum_{j\in\mathcal{E}(.,i)} y_{ji}\right)^2 \tag{3.147}$$

$$-\frac{N}{2}z^2 - z\left(\sum_{i\in\mathcal{V}}\frac{c_i}{N}\right) - G^*(z) \tag{3.148}$$

where N is the number of elements in $\mathcal{V}$.

The optimization over $(y_{ij})_{ij}$ and z is decoupled. Solving with respect to $(y_{ij})_{ij}$ is the same than solving a ROF model. The optimization with respect to z is nothing more than computing the proximal operator of G^*.

Assuming the proximal operator of G^* is simple to compute, we obtain in linear time the primal solution of (3.141):

$$x_i^* = x_i^{*,ROF} - z^* \tag{3.149}$$

This concludes the proof.

3.6 Examples of application

We illustrate features of algorithms 1 and 4 with two classic computer vision tasks: maxflow/mincut and image denoising. We could also have experimented with formulating image ressampling as an optimization problem as in [68].

3.6.1 Mincut/Maxflow

Mincut/Maxflow is ubiquitous in computer science [55, 74]. Indeed tasks such as edge-disjoint paths, vertex-disjoint paths, maximum matchings in bipartite graphs and some assignment problems can be formulated as either a maxflow or a mincut problem [47]. In our registration context, it is a key component of graph-cut for multi-label optimization.

Mincut as a non smooth continuous optimization problem

We are given a directed graph, $\mathcal{G} = [\mathcal{V}, \mathcal{E}]$, where $\mathcal{V}$ is the set of vertices and $\mathcal{E} \subset \mathcal{V} \times \mathcal{V}$ is a set of directed edges. A binary variable $x_i \in \{0, 1\}$ defines the configuration of each vertex $i \in \mathcal{V}$. The potential of the configuration of each vertex i is $\phi_i \in \mathbb{R}$. The potential of the configuration of each edge $(i, j) \in \mathcal{E}$ is $\psi_{i,j} \in \mathbb{R}^+$. We will present in more detail the mincut and maxflow problem in

the next chapter 4.3. To solve the mincut problem [173] proved that we can compute:

$$\arg\min_{x_i \in \{0,1\}} \sum_{i \in \mathcal{V}} \phi_i x_i + \sum_{(i,j) \in \mathcal{E}} \psi_{i,j} \max(x_i - x_j, 0) \tag{3.150}$$

The authors of [173] show that one can relax the support of all x_i from $\{0, 1\}$ to $[0, 1]$ and still recover the optimal binary solution of (3.150). This leads to compute:

$$\arg\min_{x_i \in [0,1]} \sum_{i \in \mathcal{V}} \phi_i x_i + \sum_{(i,j) \in \mathcal{E}} \psi_{i,j} \max(x_i - x_j, 0) \tag{3.151}$$

To recover previously introduced notation we identify:

- $M_i(x) = \phi_i x_i$ for $\forall i \in \mathcal{V}$
- $R(x) = \max(x, 0)$ which is a special case of the point wise asymmetric Huber function.
- $L = WG$ the weighted adjacency matrix representing the directed edges of $\mathcal{E}$ with potential ψ. Hence, we have $L_{ij} = \psi_{ij}$.

Hence, we can use algorithm 1 to solve problem (3.151).

Experiments

Settings We generate random mincut problems where the vertices are arranged on a grid of size $[100, 100]$ and the edges follow the 4-connectivity of the grid. We draw (ϕ_i) from a normal distribution $\mathcal{N}(0, 1)$ and we draw $(\psi_{i,j})$ uniformly in $[\rho, \lambda]$ with $(\rho, \lambda) \in \mathcal{R}^+ \times \mathcal{R}^+$ and $\rho \leqslant \lambda$.

For all experiments, we initialize all primal variables randomly in $[0, 1]$ and all dual variables to 0. We monitor the primal dual gap throughout the iterations of algorithm 1.

Conditioning In this first set of experiments, we investigate how conditioning affects the convergence rate. We generate 6 problems with $[\rho, \lambda] \in \{(0.99, 1.01), (0.9, 1.1), (0.75, 1.25), (0.5, 1.5), (0.25, 1.75), (0, 2)\}$. This progressively deteriorates the conditioning number of operator L. For each problem we run the algorithm with and without conditioning improvement as defined in Section 3.3.4. We hand pick step sizes τ and σ to ensure a fast convergence.

Figure 3.12 exemplifies the importance of conditioning for ensuring a fast convergence of algorithm 1. The simple diagonal preconditioning technique described in Section 3.3.4 helps to maintain a good conditioning. This results in a faster convergence illustrated by a smaller primal-dual gap (red curves) when compared with the initial problem (blue curves). In all circumstances, the preconditioning yields a lower primal-dual gap.

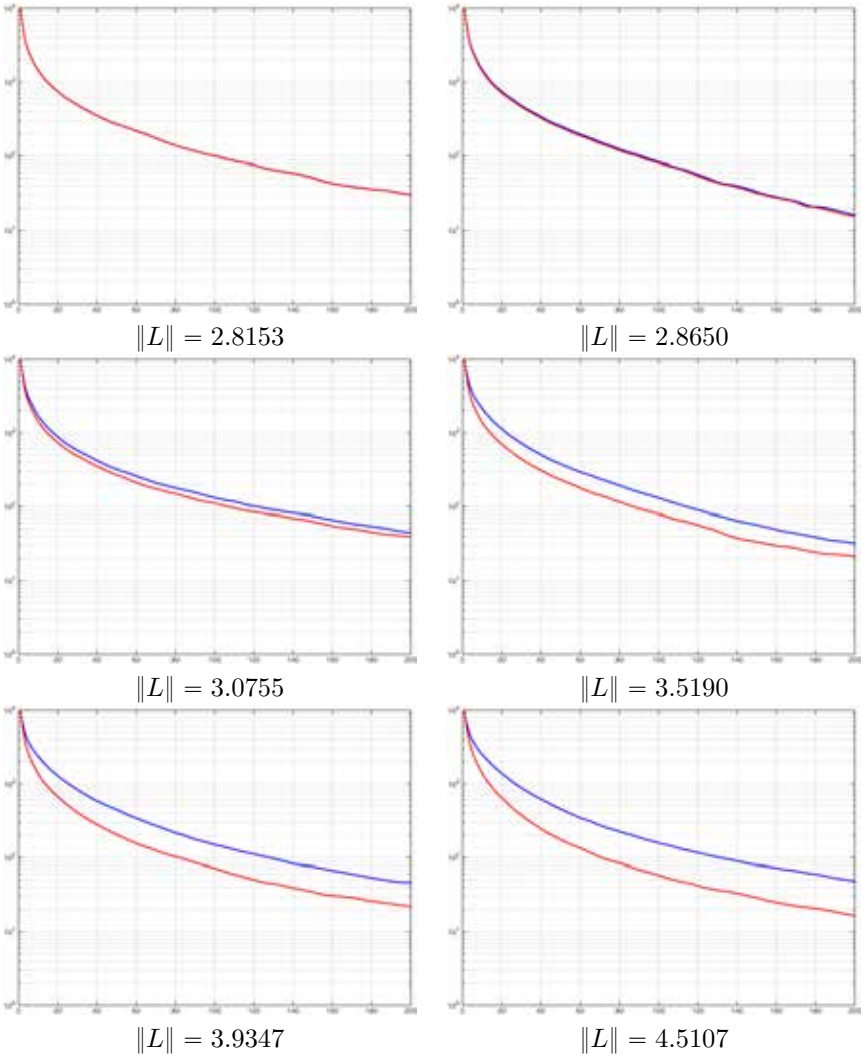


Figure 3.12 – Primal dual gap for various conditioned operator L: red curves with conditioning improvement, blue curves without.

Auto tuning of step sizes We now investigate the tuning of step sizes. To this end, we set $\rho = 1$ and $\lambda = 1$ and we run the algorithm with and without auto-tuning as defined in Section 3.3.4 for a collection of initial $\tau \in \{0.001, 0.01, 0.1, 1, 10, 100\}$. The other step size σ is set to the highest possible that guarantees convergence as defined by (3.30). We run the algorithm for 200 iterations.

Figure 3.13 pictures the sensitivity of algorithm 1 with respect to step size

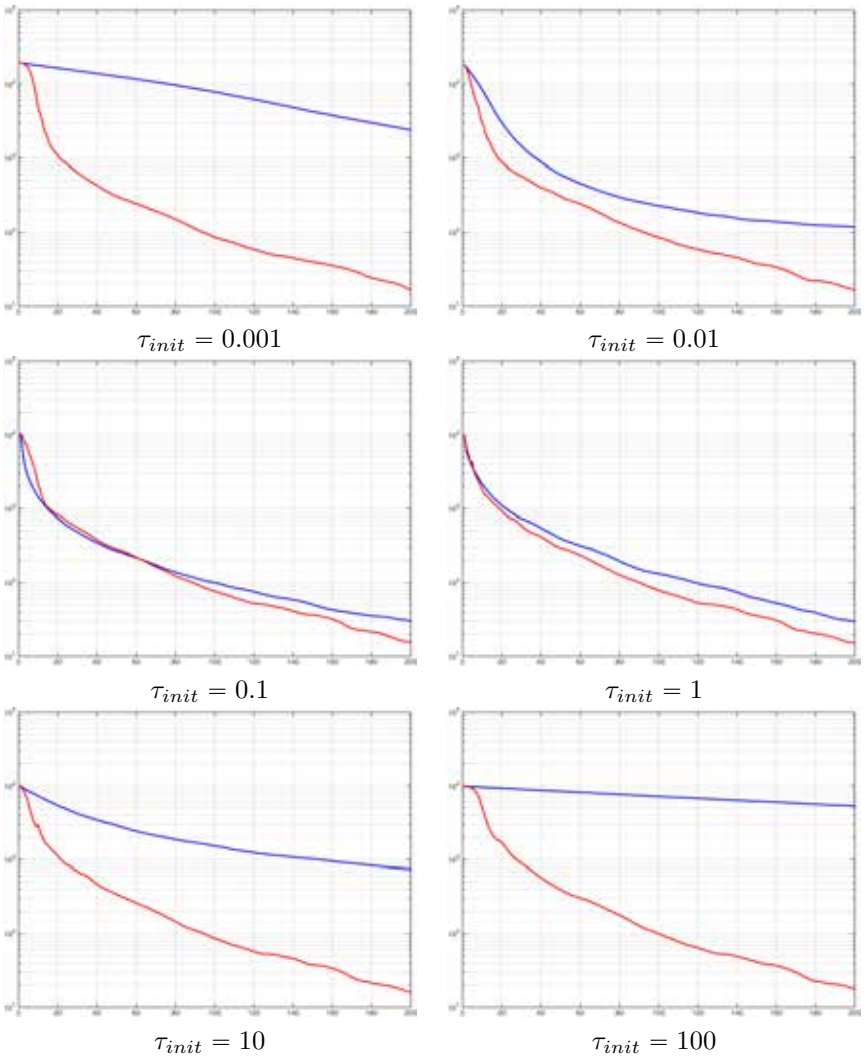


Figure 3.13 – Primal dual gap for various initial values of τ: red curves with auto tuning, blue curves without auto-tuning.

parameters τ and σ. In our experiments the sweet spot seems to be around $\tau = 0.5$. We notice that as soon as we move away from $\tau = 0.5$, the convergence rate greatly slows down as illustrated by the blue curves for $\tau = \{0.001, 0.01, 10, 100\}$. This underlines the importance of auto-tuning algorithm 4. Indeed, for any tested initial value of τ, the auto-tuning algorithm properly adjusts the values of τ and σ to ensure a fast convergence since all red curves end-up at a similar primal dual gap.

3.6.2 Image denoising

Image denoising is generally one of the first components of many vision system. It has been studied for decades by the computer vision community [70, 128, 24]. In our context, we sometimes have to process images contaminated by various degrees of additive or multiplicative noise. We can use denoising techniques either as a pre-processing step to improve the quality of image to process. But, we can also use denoising as a post-processing step to improve the quality of a disparity map for instance. Hence, in this chapter we present a very general formulation of denoising.

Image denoising as an optimization problem

In this task we are given an image $r : \Omega \to [0, 1]$ contaminated by Gaussian noise and we attempt to estimate the uncontaminated image x. To this end, we optimize the following equation:

$$\underset{x_i \in [0,1]}{\arg\min} \sum_{i \in \Omega} H_{1,\alpha}(x_i - r_i) + \sum_{i \in \Omega} \sum_{j \in \mathcal{N}_i(\rho)} H_{w_{ij},\beta}(x_i - x_j) \tag{3.152}$$

Where:

- $H_{a,\alpha}$ is a symmetric Huber function with a a-slope and α-curvature:

$$H_{a,\alpha} = \begin{cases} ax^2/(2\alpha) & \text{if } x \in [-\alpha, \alpha] \\ a(|x| - \alpha/2) & \text{otherwise} \end{cases} \tag{3.153}$$

- $\mathcal{N}_i$ is the set of neighbor pixels of i within a ρ-radius.
- w_{ij} are positive real scalars.

The equation (3.152) enforces two fundamental properties. First, the value of each pixel of the denoised image needs to be somehow close to the value of the pixel of the contaminated image. This is achieved by penalizing the distance of x_i to r_i under the Huber norm $H_{1,\alpha}$. Second, natural images exhibit some notion of smoothness. Therefore, for each pixel $i \in \Omega$ we penalize a weighted disagreement with its neighbors $\mathcal{N}_i$ through another Huber norm $H_{w_{ij},\beta}$.

The neighborhood $\mathcal{N}_i$ of a pixel i is the set of pixels of Ω within a given radius ρ:

$$\mathcal{N}_i(\rho) = \left[j \in \Omega \quad | \quad \|i - j\|_2 \leqslant \rho \right]. \tag{3.154}$$

The weights w_{ij} are defined as:

$$w_{ij} = \lambda \exp\left(-\beta |r_i - r_j|\right), \tag{3.155}$$

where λ and β are real positive scalar used to tune the regularization strength.

Experiments

As a toy example, we proceed to denoise an image contaminated with a moderate white Gaussian noise. We perform a first denoising with the neighborhood radius ρ set to 1 and another denoising with $\rho = 3$. All other parameters are adjusted to produce pleasant looking results.

Noise free image | Image contaminated with noise

1-radius denoised image | 3-radius denoised image

Figure 3.14 – Denoising of an image contaminated with white Gaussian noise.

We observe in figure 3.14 that both denoised images look smooth but exhibit the noise-free image structure. However, most of the fine details are lost. We notice that increasing the neighborhood radius creates a smoother result while preserving strong edges.

We could easily improve this simple yet effective denoising formulation. We point the curious reader to recent work that solely focus on denoising [22, 81, 136, 3, 116, 118, 117].

3.6.3 L-ROF model vs ROF model for denoising

We investigate the convergence rate of the L-ROF model through several numerical experiments. To this end, we consider the denoising task of an image $c : \Omega \to [0, 1]$ contaminated with white Gaussian noise, and we solve the following L-ROF optimization problem:

$$\min_{x_i \in \mathbb{R}} \max_{y_{ij} \in [-w,w]} \sum_{i \in \Omega} -c_i x_i + \sum_{i \in \Omega} \sum_{j \in \mathcal{N}_i(\rho)} y_{ij} \left| x_i - x_j \right| \tag{3.156}$$

As a baseline we use the ROF model optimized with Algorithm 2 that yields a $\frac{1}{N^2}$ convergence rate:

$$\min_{x_i \in \mathbb{R}} \sum_{i \in \Omega} \frac{1}{2} x_i^2 - c_i x_i + \sum_{i \in \Omega} \sum_{j \in \mathcal{N}_i(\rho)} w \left| x_i - x_j \right| \tag{3.157}$$

Settings

For all experiments, we always set the regularization parameter w to be equal to the standard deviation σ_n of the Gaussian noise. We consider three noise levels, low with $\sigma_n = 0.02$, medium with $\sigma_n = 0.1$ and high $\sigma_n = 0.25$ as illustrated in figure 3.15. We set the smoothing parameter θ to 1 for both primal-dual optimization algorithms. The primal and dual variables are initialized to 0 for both algorithms.

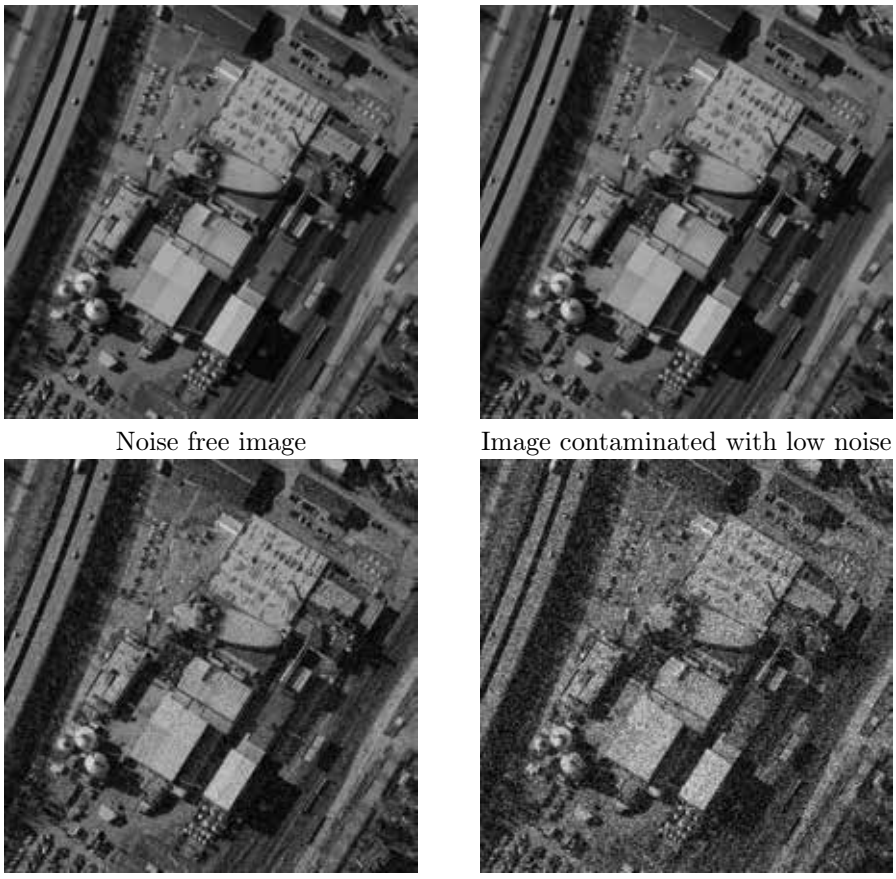
Experiments with no scaling

In this first set of experiments we compare the L-ROF model without any scaling, $\delta = 0$, to the ROF model. We sweep space of the initial primal update step size τ from 1000 to 1 and set the dual step size accordingly to ensure convergence. We remind that internally the algorithm optimizing the ROF model adjusts both the primal-dual step sizes along with the smoothing parameter θ. The L-ROF model maintains the initial values throughout the optimization.

For each noise level, we run each algorithm for 500 iterations and we display the primal dual gap at iterations 100, 250 and 500 in figures 3.16, 3.17 and 3.18.

For the L-ROF model, due to Property 1 and the constant initialization of the primal variable, the algorithm is insensitive to the primal update step size. We observe this behavior for the three noise regimes as all red curves of figures 3.16, 3.17 and 3.18 are constant. Since primal dual algorithms convergence are sensitive to rightly tuning the update step sizes, this is a nice property of the L-ROF model.

In terms of convergence, we see that without scaling the L-ROF model slightly outperforms the ROF model. This is somehow unexpected since the ROF model benefits of a $1/N^2$ convergence rate while the L-ROF model converges with $1/N$ rate. However, it has been observed in [27] that algorithm (1) from which the L-ROF optimization algorithm is derived obtains $1/N^2$ convergence rate if the



Noise free image | Image contaminated with low noise

Image contaminated with medium noise | Image contaminated with high noise

Figure 3.15 – Reference image and different levels of noise contamination.

primal-dual update step-sizes are properly tuned. Since the L-ROF model is insensitive to update step-size tuning, the optimization algorithm might always perform in its optimal regime. This could explain the observed exponential convergence rate.

Experiments with scaling

We now proceed with a second set of experiments where we compare the L-ROF model with scaling to the ROF model. We set the initial primal update step size τ to 5 since it seems optimal in the last set of experiments, and we set the dual step size accordingly to ensure convergence. We sweep space of the scaling factor δ from 0.5 to 1 by 0.1 increments and the space of scaling period from 2 to 12 by 1 increments.

For each noise level, we run each algorithm for 500 iterations and we display

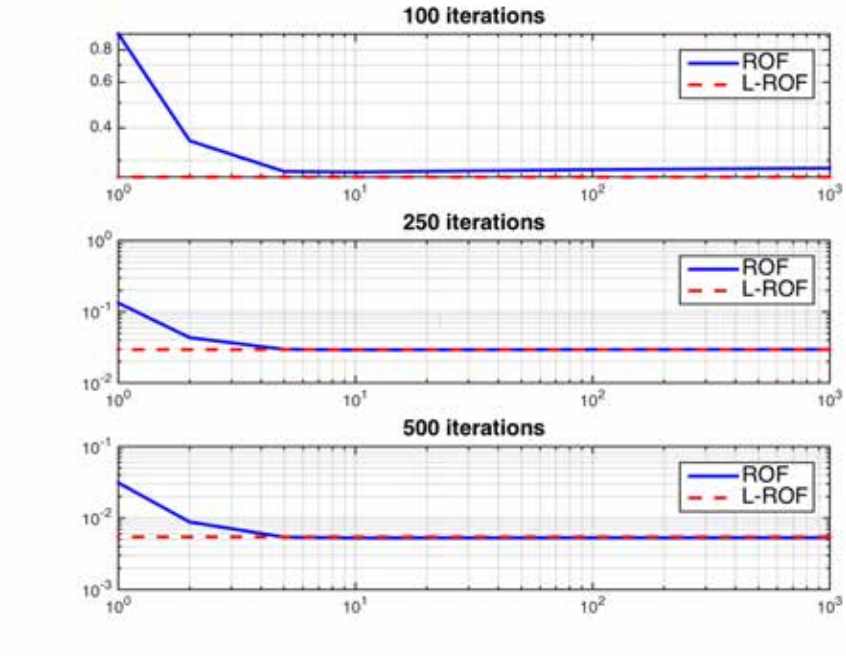


Figure 3.16 – Primal dual gap for low noise experiment without scaling at iteration 100, 200 and 500

the difference of log in base 10 of the primal dual gap at iterations 100, 250 and 500 in figures 3.20, 3.21 and3.22 between the L-ROF and ROF model. Negative values mean that the L-ROF model converges faster than the ROF model.

We see that the scaling greatly improves the convergence of the L-ROF model to up to two orders of magnitude. A typical primal dual gap evolution of the L-ROF model is shown in figure 3.19. Initially the convergence follows the one of the ROF model. Then, it stabilizes to an exponential convergence (linear trend in the log domain) while the ROF model keeps its quadratic convergence.

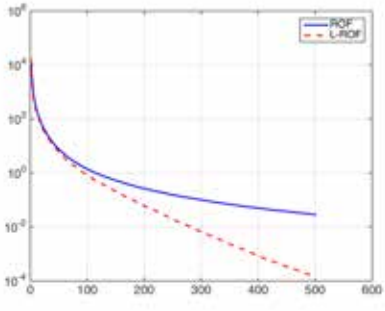


Figure 3.19 – Primal dual gap for the L-ROF model with scaling ($\delta = 0.7$ and scaling period of 10) and the ROF model

The optimal scaling factor varies with the scaling period. The more we scale

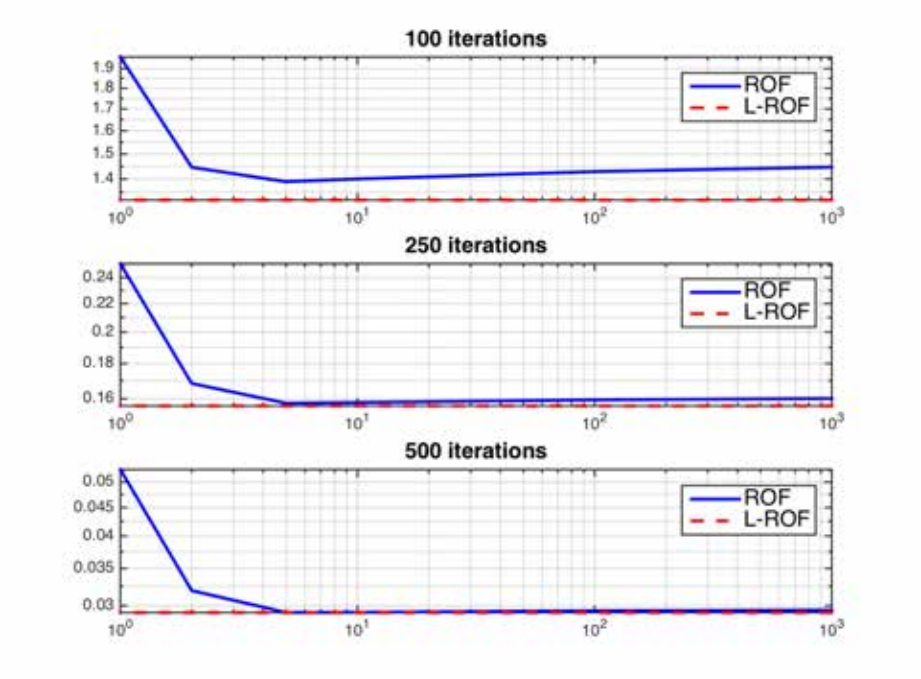


Figure 3.17 – Primal dual gap for medium noise experiment without scaling at iteration 100, 200 and 500

the less frequent we should do it. This is in line with the role of scaling. Indeed, the scaling decreases the influence of the past updates. A possible interpretation is that since the early updates are far from being optimal they can and should be progressively discarded. However, discarding too much of the past is counter productive. We let to future work the study of an optimal or heuristic criterion to dynamically tune the scaling factor parameter δ and the scaling period.

To conclude, the L-ROF model provides an interesting alternative to the classic ROF model when highly accurate solutions are required.

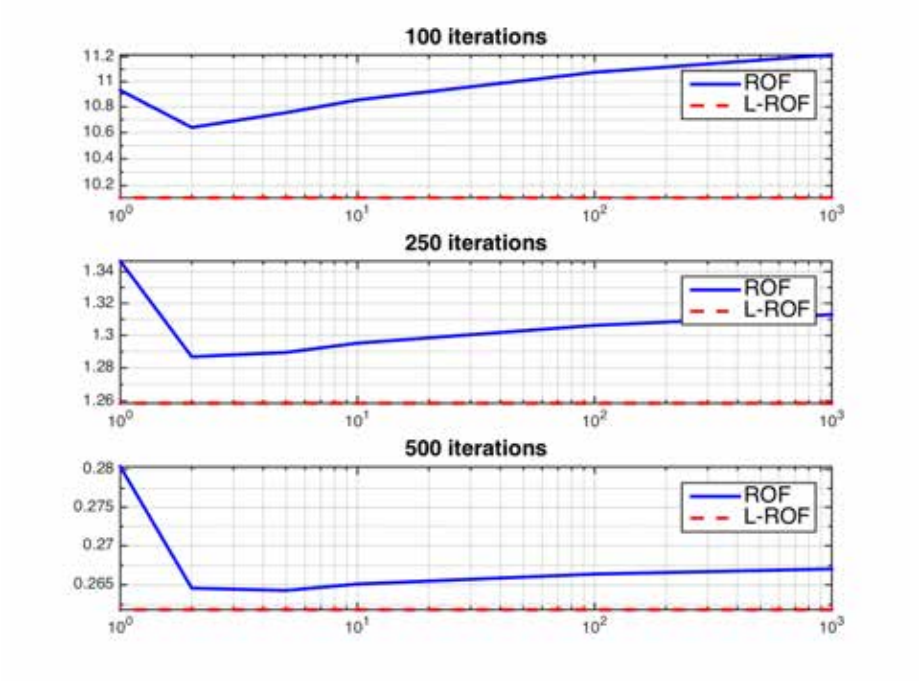


Figure 3.18 – Primal dual gap for high noise experiment without scaling at iteration 100, 200 and 500

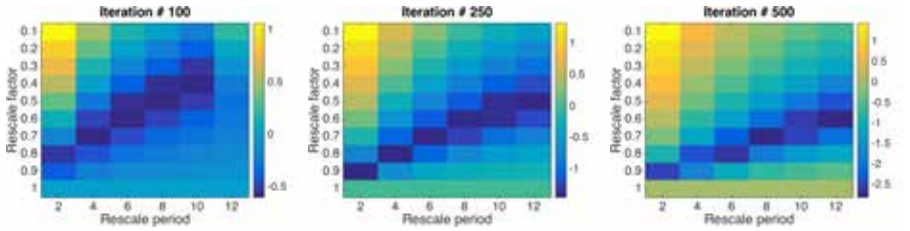


Figure 3.20 – Primal dual gap for low noise experiment without scaling at iteration 100, 200 and 500 (left to right)

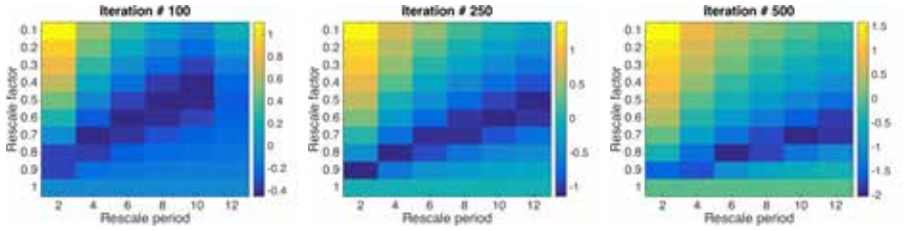


Figure 3.21 – Primal dual gap for medium noise experiment without scaling at iteration 100, 200 and 500 (left to right)

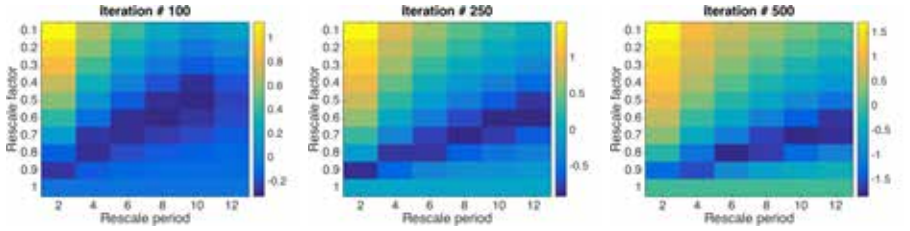


Figure 3.22 – Primal dual gap for high noise experiment without scaling at iteration 100, 200 and 500 (left to right)

3.7 Conclusion

This first technical chapter introduced the basic of convex optimization. We progressively yet precisely introduce the fundamentals of the First order Primal-Dual techniques for convex optimization. We thoroughly studied the dual solution space of TV regularized problems and we propose through some theorems a better understanding of how different TV models relate one to another.

However, not all tasks can be formulated as a convex optimization problem without scarifying to much modeling accuracy. To this end, we present in the next chapter techniques for non-convex optimization.

Chapter 4

Maxflow and Graph cuts techniques

4.1 Introduction and contributions

4.1.1 Introduction

In this chapter we consider the optimization of a non convex discrete energy. To this end we quickly introduce some basic background related to discrete optimization. Then, we introduce a key algorithm in computer science: mincut / maxflow. We study four different algorithms to solve the mincut / maxflow problem.

Finally, we study graph-cuts techniques for their known efficiency to optimize pairwise non convex discrete energies. We introduce two algorithms: Alpha Expansion and Fast PD. Fast PD is known to be significantly faster than the Alpha Expansion. However, its current implementation requires a large amount of memory which makes it unsuitable to our context. Hence, we investigate in details the implementation of the Fast-PD algorithm.

4.1.2 Chapter organization

The section 4.2 introduces the basics of discrete optimization and we describe the problem of interest. We discuss in section 4.3 the mincut and maxflow problems as the primal and dual form of the same task. The section 4.4 presents dedicated solver for the mincut and maxflow problems. The section 4.5 introduces the alpha-expansion and the Fast-PD algorithms for solving multi-label pairwise problems. In the section 4.6 we perform numerous experiments to compare different mincut / maxflow solvers and we evaluate our own implementation of Fast-PD.

4.1.3 Contributions

The main contribution of this chapter is the complete re-implementation of the famous Fast-PD algorithm. Our implementation non only drastically reduces the amount of memory required but it also runs faster than the implementation proposed by the original authors of Fast-PD. We also propose an even faster implementation dedicated to grid-like problems.

This allows us for instance to perform discrete optimization in the context of stereo-matching with large images while only using a recent laptop.

4.2 Discrete optimization in a tiny nutshell

We start by presenting the necessary knowledge to further study both mincut / maxflow algorithms and graph-cuts techniques. We refer the reader to [142], [130] and [140] for more details.

4.2.1 Sub-modularity

Sub-modularity plays a central role for discrete optimization in the same way that convexity is crucial for convex optimization [153].

In this work we make extensive use of binary functions. In this settings, the sub-modularity simplifies to:

Definition 12. Let $\phi : \{0,1\} \times \{0,1\} \rightarrow \mathbb{R}$ be a 2-binary function. We say that ϕ is sub-modular if and only if:

$$\phi(0,0) + \phi(1,1) \leqslant \phi(0,1) + \phi(1,0) \tag{4.1}$$

We introduce some useful properties about sub-modular functions.

Property 3. If ϕ is a sub-modular function, then $\forall \alpha \in \mathbb{R}^+$ and $\forall \beta \in \mathbb{R}$ the following function:

$$\alpha\phi(.) + \beta \tag{4.2}$$

is also sub-modular.

Property 4. If ϕ and ψ are sub-modular functions, then $\phi + \psi$ is a sub-modular function.

Property 5. Any finite positive weighted sum of sub-modular functions is a sub-modular function.

Theorem 9. *If ϕ is a sum of binary sub-modular functions, then one can compute its minimum in polynomial time by solving a min-cut problem.*

Hence, if we can formulate our discrete optimization problems as a sum of binary sub-modular functions to optimize, or as a sequence of sums of binary sub-modular functions to optimize, we can derive a polynomial time optimization scheme.

4.2.2 Problems of interest

We remind the problems studied in this chapter.

Notations

We make use of the following notations:

- The graph $\mathcal{G}$ is a directed graph $\mathcal{G} = [\mathcal{V}, \mathcal{E}]$ composed of a finite set of vertices $\mathcal{V}$ and a finite set of directed edges $\mathcal{E}$. The edge (i, j) of $\mathcal{E}$ creates a connexion from the vertex i to another vertex j.
- The discrete variable x_i defines the configuration of each vertex $i \in \mathcal{V}$. The domain of x_i is $\mathcal{L} = \{0, \dots, L-1\}$ with $L \in \mathbb{N}$.
- A discrete unary potential function ϕ_i defines the configuration cost of each vertex $i \in \mathcal{V}$: $\phi_i : \mathcal{L} \to \mathbb{R}$.
- A discrete potential function ϕ_{ij} defines the configuration cost of each directed edge (i, j) of $\mathcal{E}$: $\phi_{ij} : \mathcal{L} \times \mathcal{L} \to \mathbb{R}^+$. We assume that $\phi_{ij}(l, l) = 0,\ \forall l \in \mathcal{L}$.

Canonical problems

We are interested in solving the following problem:

$$(x_i^*)_i = \underset{(x_i \in \mathcal{L})_i}{\arg\min} \quad \sum_{i \in \mathcal{V}} \phi_i(x_i) + \sum_{(i,j) \in \mathcal{E}} \phi_{ij}(x_i, x_j). \tag{4.3}$$

4.2.3 Representation of pairwise binary sub-modular functions

We suppose we are given a pairwise binary sub-modular function $\psi_{ij} : \{0, 1\} \times \{0, 1\} \to \mathbb{R}$:

$$\psi_{ij} = \begin{bmatrix} A & B \\ C & D \end{bmatrix} \tag{4.4}$$

where $(A, B, C, D) \in \mathbb{R} \times \mathbb{R} \times \mathbb{R} \times \mathbb{R}$

Theorem 10. *Any pairwise binary sub-modular function $\psi_{ij} : \{0, 1\} \times \{0, 1\} \to \mathbb{R}$ can be represented as:*

$$\psi_{ij}(x_i, x_j) = c_i x_i + c_j x_j + w_{ij} \max(x_i - x_j, 0) + d \tag{4.5}$$

with: $c_i = C - A$, $c_j = D - C$, $w_{ij} = B + C - A - D$, and $d = A$.

Proof

The proof, given in [95], derives from basic calculus:

$$\begin{aligned} c_i x_i + c_j x_j &\equiv \begin{vmatrix} 0 & D-C \\ C-A & D-A \end{vmatrix} \\ c_i x_i + c_j x_j + w_{ij}\max(x_i - x_j, 0) &\equiv \begin{vmatrix} 0 & B-A \\ C-A & D-A \end{vmatrix} \\ c_i x_i + c_j x_j + w_{ij}\max(x_i - x_j, 0) + \mathrm{d} &\equiv \begin{vmatrix} A & B \\ C & D \end{vmatrix} \\ c_i x_i + c_j x_j + w_{ij}\max(x_i - x_j, 0) + \mathrm{d} &\equiv \psi_{ij}(x_i, x_j) \end{aligned} \tag{4.6}$$

4.2.4 A link between discrete and convex optimization through TV regularization

Theorem 11. *Let $(z_i^*)_i$ be the solution of the following problem:*

$$(z_i^*)_i = \underset{(z_i \in [0,1])_i}{\arg\min} \quad \sum_{i \in \mathcal{V}} c_i z_i + \sum_{(i,j) \in \mathcal{E}} w_{ij} \max\left(z_i - z_j, 0\right). \tag{4.7}$$

Then, $\forall t \in [0,1[$:

$$x_i^* = \begin{cases} 0 & \text{if} \quad z_i^* \leqslant t \\ 1 & \text{if} \quad z_i^* > t \end{cases} \tag{4.8}$$

The set of variables $(x_i^*)_i$ are a solution of the associated binary problem:

$$(x_i^*)_i = \underset{(x_i \in \{0,1\})_i}{\arg\min} \quad \sum_{i \in \mathcal{V}} c_i x_i + \sum_{(i,j) \in \mathcal{E}} w_{ij} \max\left(x_i - x_j, 0\right). \tag{4.9}$$

Proof

We introduce the following change of variables for $a \in \mathbb{R}^{*,+}$ and $t \in [0,1[$:

$$z_i^a = \frac{1}{a}(z_i - t), \quad \forall i \in \mathcal{V} \tag{4.10}$$

We apply this change of variable to (4.7):

$$\left(\frac{1}{a}(z_i^* - t)\right)_i = \underset{\left(z_i \in \left[\frac{-t}{a}, \frac{1-t}{a}\right]\right)_i}{\arg\min} \quad \sum_{i \in \mathcal{V}} c_i z_i + \sum_{(i,j) \in \mathcal{E}} w_{ij} \max\left(z_i - z_j, 0\right). \tag{4.11}$$

We notice that $\exists a > 0$ such that:

$$\begin{cases} \dfrac{1}{a}(z_i^* - t) \geqslant 1 \text{ and } \dfrac{1-t}{a} \geqslant 1 & \text{if} \quad z_i^* > t \\ \dfrac{1}{a}(z_i^* - t) \leqslant 0 \text{ and } \dfrac{-t}{a} \leqslant 0 & \text{if} \quad z_i^* \leqslant t \end{cases} \tag{4.12}$$

We apply the truncation theorem of TV regularized problems 3 with the $[0,1]$ interval to equation (4.11):

$$\begin{aligned}\left(\left[\frac{1}{a}(z_i^* - t)\right]_{[0,1]}\right)_i &= \underset{(x_i\in[0,1])_i}{\arg\min} \quad \sum_{i\in\mathcal{V}} c_i x_i + \sum_{(i,j)\in\mathcal{E}} w_{ij}\max\left(x_i - x_j, 0\right). \\ (x_i^*)_i &= \underset{(x_i\in\{0,1\})_i}{\arg\min} \quad \sum_{i\in\mathcal{V}} c_i x_i + \sum_{(i,j)\in\mathcal{E}} w_{ij}\max\left(x_i - x_j, 0\right).\end{aligned} \tag{4.13}$$

This completes the proof.

4.2.5 Primal dual scheme for integer Linear programming

To study Fast-PD, we need to introduce the fundamental of linear programming and the approximate primal-dual scheme of [64] and [125] for integer Linear programming optimization. We refer the curious reader to [40] for more details.

Primal and dual forms of a linear programming problem

Definition 13. A Linear Programming, LP, problem consists of minimizing a linear combination of variables under inequality constraints that can be expressed in canonical primal form as:

$$\begin{aligned}\min_{(x_i)_i} \quad & \sum_{i=1}^{I} c_i x_i \\ s.t. \quad & \sum_{i=1}^{I} a_{ji} x_i \leqslant b_j, \qquad \forall j \in [1, J] \\ & x_i \geqslant 0, \qquad \forall i \in [1, I]\end{aligned}$$

with: $(I, J) \in \mathbb{N}\times\mathbb{N}, (c_i)_i \in \mathbb{R}$, $(a_{ji})_{ij} \in \mathbb{R}$, and $(b_j)_j \in \mathbb{R}$.

Definition 14. The dual canonical form of LP is:

$$\begin{aligned}\max_{(y_j)_j} \quad & \sum_{j=1}^{J} b_j y_j \\ s.t. \quad & \sum_{j=0}^{J} a_{ji} y_j \leqslant c_i, \qquad \forall i \in [1, I] \\ & y_j \geqslant 0, \qquad \forall j \in [1, J]\end{aligned}$$

We note that any LP can be easily transformed into either a canonical primal or dual form.

Complementary slackness conditions

The complementary slackness plays a central role to certify the optimality of a primal-dual solution.

Property 6. The primal complementary slackness condition is:

$$\forall i \in [1, I], \quad x_i > 0 \quad \Rightarrow \quad \sum_{j=0}^{J} a_{ji} y_j = c_i$$

Property 7. The dual complementary slackness condition is:

$$\forall j \in [1, J], \quad y_j > 0 \quad \Rightarrow \quad \sum_{i=1}^{I} a_{ji} x_i = b_j$$

Theorem 12. *If a pair (x, y) of primal-dual pair verifies the primal and dual complementary slackness conditions, then x and y are solutions of the primal and dual LP problem.*

Hence, the primal and dual complementary slackness conditions can be used as an optimality certificate.

Relaxed complementary slackness conditions

Interestingly, we can also derive a more general definition for the complementary slackness conditions by relaxing the equality in the second term of each conditions.

Property 8. For a given $\alpha \in [1, +\infty]$ the primal relaxed complementary slackness condition is:

$$\forall i \in [1, I], \quad x_i > 0 \quad \Rightarrow \quad \frac{c_i}{\alpha} \leqslant \sum_{j=0}^{J} a_{ji} y_j \leqslant c_i$$

Property 9. For a given $\beta \in [1, +\infty]$ the dual relaxed complementary slackness condition is:

$$\forall j \in [1, J], \quad y_j > 0 \quad \Rightarrow \quad b_j \leqslant \sum_{i=1}^{I} a_{ji} x_i \leqslant \beta b_j$$

Theorem 13. *If a pair (x, y) of primal-dual solution verifies the primal and dual relaxed complementary slackness conditions with factor α and β:*

$$\sum_{i=1}^{I} c_i x_i \leqslant \alpha\beta \sum_{j=1}^{J} b_j y_j$$

then x and y are $\alpha\beta$ optimal solutions of the primal and dual LP problem

Primal dual scheme for binary LP

We now give a general framework to optimize approximately a given Binary LP, BLP, $\mathcal{B}$. The key idea is to relax this BLP to an LP, and to compute the dual problem. The framework makes use of both the primal and dual problems by iteratively improving the dual solution while generating integer primal solutions until no dual improvement can be made. The quality of the approximation of the integer primal solution is controlled by factors α and β. The algorithm 8 sums up the primal dual scheme for binary LP.

Algorithm 8: Approximate primal dual scheme for Binary Linear Programming problems

Data: Inputs: $\mathcal{B}$, α and β
Result: $(x_i)_i$, $(y_j)_j$
Relax the ILP to an LP: $\rightarrow \mathcal{P}$.
Compute the dual of $\mathcal{P} : \rightarrow \mathcal{D}$.
Initialize the primal variables to be binary.
Initialize the dual variables to be feasible.
while *Relaxed complementary slackness conditions are not all statisfied* **do**
    Select a subset of dual variables that does not satisfy the relaxed complementary slackness conditions.

    Update this subset of dual variables with a method of choice.

    Use the primal relaxed complementrary slackness conditions to update the primal variables by an integer quantity.

4.3 Max-flow and min-cut problems

In 1956, two research teams composed of P. Elias, A. Feinstein, and C.E. Shannon for the first one [46], and L.R. Ford, Jr. and D.R. Fulkerson [54] for the second demonstrated the celebrated max-flow / min-cut theorem. This theorem derives from the strong duality in linear programming problems [108].

In this section, we start by proving the max-flow / min-cut theorem, then we introduce simply equations for both the primal (min-cut), the primal-dual, and the dual (max-flow) problems. Those simplified equations ease the creation of max-flow / min-cut solvers dedicated to image processing tasks.

4.3.1 The max-flow / min-cut

4.3.2 From a min-cut problem to a max-flow problem

Using the representation defined by theorem (10), we can formulate, up to a constant term, any sum of binary pairwise sub-modular functions as :

$$\sum_{i\in\mathcal{V}} \phi_i(0)\,(1-x_i) + \phi_i(1)x_i + \sum_{(i,j)\in\mathcal{E}} w_{ij}\max\left(x_i - x_j, 0\right) \tag{4.14}$$

The min-cut problem

The binary min-cut problem is simply the configuration of minimum cost of the previous equation 4.14:

$$(x_i^*)_i = \underset{(x_i\in\{0,1\})_i}{\arg\min} \quad \sum_{i\in\mathcal{V}} \phi_i(0)\,(1-x_i) + \phi_i(1)x_i + \sum_{(i,j)\in\mathcal{E}} w_{ij}\max\left(x_i - x_j, 0\right) \tag{4.15}$$

Relaxing primal variables

Thanks to theorem 11 we can solve the relaxed problem while still being able to retrieve by a simple truncation an optimal solution of (4.15).

$$\min_{(l_i\in[0,1])_i} \quad \sum_{i\in\mathcal{V}} \phi_i(0)\,(1-l_i) + \phi_i(1)l_i + \sum_{(i,j)\in\mathcal{E}} w_{ij}\max\left(l_i - l_j, 0\right). \tag{4.16}$$

where s-indexed variables are connected to the source and t indexed variables are linked to the sink.

We observe that we now deal with a non smooth convex optimization problem. This transformation was observed by [28] and then [174].

Dualizing the potentials

We transform the previous problem by dualizing the potentials. We formulate both unary and pairwise potentials using the techniques of previous chapter:

$$\begin{aligned}
\phi_i(0) &= \max_{\rho_{si}\in[0,\phi_i(0)]} && \rho_{si} \\
\phi_i(1) &= \max_{\rho_{it}\in[0,\phi_i(1)]} && \rho_{it} \\
w_{ij}\max\left(x_i - x_j, 0\right) &= \max_{f_{ij}\in[0,w_{ij}]} && f_{ij}\left(x_i - x_j\right)
\end{aligned} \tag{4.17}$$

We obtain the following primal dual problem:

$$\begin{array}{lll} \min\limits_{(x_i)} \quad \max\limits_{(\rho_{si}),(\rho_{it}),(f_{ij})} & \sum\limits_{i\in\mathcal{V}} \rho_{si}\left(1-x_i\right)+\rho_{it}x_i+\sum\limits_{(i,j)\in\mathcal{E}} f_{ij}\left(x_i-x_j\right). & \\ \text{subject to} & x_i\in[0,1], \quad \forall i\in\mathcal{V} & \\ & f_{ij}\in\left[0,w_{ij}\right], \quad \forall\left(i,j\right)\in\mathcal{E} & \\ & \rho_{si}\in\left[0,\phi_i(0)\right], \quad \forall i\in\mathcal{V} & \\ & \rho_{it}\in\left[0,\phi_i(1)\right], \quad \forall i\in\mathcal{V} & \end{array} \tag{4.18}$$

We introduce the following notations to minimize clutter in equations:

$$\mathcal{E}(i,.)=\{j\in\mathcal{V}\,|\,(i,j)\in\mathcal{E}\}. \tag{4.19}$$

$$\mathcal{E}(.,i)=\{j\in\mathcal{V}\,|\,(j,i)\in\mathcal{E}\} \tag{4.20}$$

After factorization with respect to each x_i we obtain:

$$\begin{array}{lll} \min\limits_{(x_i)} \quad \max\limits_{(\rho_{si}),(\rho_{it}),(f_{ij})} & \sum\limits_{i\in\mathcal{V}}\rho_{si} & \\ & +\sum\limits_{i\in\mathcal{V}} x_i\left[\rho_{it}-\rho_{si}+\sum\limits_{j\in\mathcal{E}(i,.)} f_{ij}-\sum\limits_{j\in\mathcal{E}(.,i)} f_{ji}\right] & \\ \text{subject to} & x_i\in[0,1], \quad \forall i\in\mathcal{V} & \\ & f_{ij}\in\left[0,w_{ij}\right], \quad \forall\left(i,j\right)\in\mathcal{E} & \\ & \rho_{si}\in\left[0,\phi_i(0)\right], \quad \forall i\in\mathcal{V} & \\ & \rho_{it}\in\left[0,\phi_i(1)\right], \quad \forall i\in\mathcal{V} & \end{array} \tag{4.21}$$

Dualizing the constraints on primal variables

For each vertex $i\in\mathcal{V}$, we introduce two variables $(k_i,s_i)\in\mathbb{R}^+\times\mathbb{R}^+$ to enforce $x_i\in[0,1]$. Hence, we can drop the domain constraint for all $(x_i)_i$

$$\begin{array}{lll} \min\limits_{(x_i)} \quad \max\limits_{(\rho_{si}),(\rho_{it}),(f_{ij}),(k_i),(s_i)} & \sum\limits_{i\in\mathcal{V}}\rho_{si} & \\ & +\sum\limits_{i\in\mathcal{V}} x_i\left[\rho_{it}-s_i-\rho_{si}+k_i+\sum\limits_{j\in\mathcal{E}(i,.)} f_{ij}-\sum\limits_{j\in\mathcal{E}(.,i)} f_{ji}\right] & \\ & +\sum\limits_{i\in\mathcal{V}}\left(x_i-1\right)k_i-x_is_i & \\ \text{subject to} & f_{ij}\in\left[0,w_{ij}\right], \quad \forall\left(i,j\right)\in\mathcal{E} & \\ & \rho_{si}\in\left[0,\phi_i(0)\right], \quad \forall i\in\mathcal{V} & \\ & \rho_{it}\in\left[0,\phi_i(1)\right], \quad \forall i\in\mathcal{V} & \\ & k_i\geqslant 0, \quad \forall i\in\mathcal{V} & \\ & s_i\geqslant 0, \quad \forall i\in\mathcal{V} & \end{array} \tag{4.22}$$

We factorize again with respect to x_i to obtain:

$$\begin{aligned}
\min_{(x_i)} \quad \max_{(\rho_{si}),(\rho_{it}),(f_{ij}),(k_i),(s_i)} \quad & \sum_{i\in\mathcal{V}} \rho_{si} - k_i \\
& + \sum_{i\in\mathcal{V}} x_i \left[\rho_{it} - s_i - \rho_{si} + k_i + \sum_{j\in\mathcal{E}(i,.)} f_{ij} - \sum_{j\in\mathcal{E}(.,i)} f_{ji} \right] \\
\text{subject to} \quad & f_{ij} \in [0, w_{ij}], \quad \forall (i,j) \in \mathcal{E} \\
& \rho_{si} \in [0, \phi_i(0)], \quad \forall i \in \mathcal{V} \\
& \rho_{it} \in [0, \phi_i(1)], \quad \forall i \in \mathcal{V} \\
& k_i \geqslant 0, \quad \forall i \in \mathcal{V} \\
& s_i \geqslant 0, \quad \forall i \in \mathcal{V}
\end{aligned} \tag{4.23}$$

Introducing the source and sink flows

We introduce the flow variables from the source and the sink:

$$\begin{aligned}
f_{si} &= \rho_{si} - k_i, \quad \forall i \in \mathcal{V}. \\
f_{it} &= \rho_{it} - s_i, \quad \forall i \in \mathcal{V}.
\end{aligned} \tag{4.24}$$

We observe that:

$$\begin{aligned}
f_{si} &\leqslant \phi_i(0), \quad \forall i \in \mathcal{V}. \\
f_{it} &\leqslant \phi_i(1), \quad \forall i \in \mathcal{V}.
\end{aligned} \tag{4.25}$$

Hence, we obtain:

$$\begin{aligned}
\min_{(x_i)} \quad \max_{(f_{si}),(f_{it}),(f_{ij})} \quad & \sum_{i\in\mathcal{V}} f_{si} \\
& + \sum_{i\in\mathcal{V}} x_i \left[f_{it} - f_{si} + \sum_{j\in\mathcal{E}(i,.)} f_{ij} - \sum_{j\in\mathcal{E}(.,i)} f_{ji} \right] \\
\text{subject to} \quad & f_{ij} \in [0, w_{ij}], \quad \forall (i,j) \in \mathcal{E} \\
& f_{si} \leqslant \phi_i(0), \quad \forall i \in \mathcal{V} \\
& f_{it} \leqslant \phi_i(1), \quad \forall i \in \mathcal{V}
\end{aligned} \tag{4.26}$$

Solving for primal variables and refactoring

We recognize that each variables x_i acts as a Lagrangian multiplier to enforce the following equality constraint:

$$f_{it} - f_{si} + \sum_{j\in\mathcal{E}(i,.)} f_{ij} - \sum_{j\in\mathcal{E}(.,i)} f_{ji} = 0 \tag{4.27}$$

The maxflow equations

Finally we get the Maxflow equations:

$$\begin{array}{lll} \max\limits_{(f_{si}),(f_{it}),(f_{ij})} & \sum\limits_{i\in\mathcal{V}} f_{si} & \\ \text{subject to} & f_{ij}\in[0,w_{ij}]\,, & \forall\,(i,j)\in\mathcal{E} \\ & f_{si}\leqslant\phi_i(0), & \forall i\in\mathcal{V} \\ & f_{it}\leqslant\phi_i(1), & \forall i\in\mathcal{V} \\ & f_{it}-f_{si}+\sum\limits_{j\in\mathcal{E}(i,.)} f_{ij}-\sum\limits_{j\in\mathcal{E}(.,i)} f_{ji}=0, & \forall i\in\mathcal{V} \end{array} \tag{4.28}$$

The last constraint enforces the conservation of the flow.

4.3.3 Simplified equations

We now introduce simplified equations for primal, primal-dual and dual problems modeling the maxflow-mincut.

The primal problem

We start from the relaxed mincut equations. We introduce the following simple factorizations:

$$c_i=\phi_i(1)-\phi_i(0),\quad \forall i\in\mathcal{V}. \tag{4.29}$$

$$\phi_\Sigma(0)=\sum_{i\in\mathcal{V}}\phi_i(0) \tag{4.30}$$

$$E^*=\min_{(x_i\in[0,1])_i}\sum_{i\in\mathcal{V}}\phi_i(0)\,(1-x_i)+\phi_i(1)x_i+\sum_{(i,j)\in\mathcal{E}} w_{ij}\max\left(x_i-x_j,0\right). \tag{4.31}$$

$$=\min_{(x_i\in[0,1])_i}\sum_{i\in\mathcal{V}}\left(\phi_i(1)-\phi_i(0)\right)x_i+\phi_i(0)+\sum_{(i,j)\in\mathcal{E}} w_{ij}\max\left(x_i-x_j,0\right). \tag{4.32}$$

$$=\min_{(x_i\in[0,1])_i}\sum_{i\in\mathcal{V}} c_i x_i+\sum_{(i,j)\in\mathcal{E}} w_{ij}\max\left(x_i-x_j,0\right)+\phi_\Sigma(0). \tag{4.33}$$

During optimization we can discard the constant $\phi_\Sigma(0)$. The primal variables are the same as in (4.16). Hence, we can use the truncating technique 11 we recover an optimal solution for (4.15).

The primal-dual problem

We get the simplified primal dual problem by simply dualizing the pairwise terms:

$$E^*=\min_{x_i\in[0,1]}\ \max_{f_{ij}\in[0,w_{ij}]}\ \sum_{i\in\mathcal{V}} c_i x_i+\sum_{(i,j)\in\mathcal{E}} f_{ij}\left(x_i-x_j\right)+\phi_\Sigma(0). \tag{4.34}$$

The dual variable f_{ij} represents the flow from vertex i to vertex j.

The dual problem

For the simplified Maxflow problem we start by factorizing (4.34) with respect to each primal variable i and then we solve for the primal variables.

$$E^* = \min_{x_i \in [0,1]} \max_{f_{ij} \in [0, w_{ij}]} \sum_{i \in \mathcal{V}} \left(c_i + \sum_{j \in \mathcal{E}(i,.)} f_{ij} - \sum_{j \in \mathcal{E}(.,i)} f_{ji} \right) x_i + \phi_\Sigma(0). \quad (4.35)$$

$$= \max_{f_{ij} \in [0, w_{ij}]} \sum_{i \in \mathcal{V}} \min_{x_i \in [0,1]} \left(c_i + \sum_{j \in \mathcal{E}(i,.)} f_{ij} - \sum_{j \in \mathcal{E}(.,i)} f_{ji} \right) x_i + \phi_\Sigma(0). \quad (4.36)$$

$$= \max_{f_{ij} \in [0, w_{ij}]} \sum_{i \in \mathcal{V}} \min \left(c_i + \sum_{j \in \mathcal{E}(i,.)} f_{ij} - \sum_{j \in \mathcal{E}(.,i)} f_{ji}, 0 \right) + \phi_\Sigma(0). \quad (4.37)$$

$$= \max_{f_{ij} \in [0, w_{ij}]} -\frac{1}{2} \left[\sum_{i \in \mathcal{V}} \left| c_i + \sum_{j \in \mathcal{E}(i,.)} f_{ij} - \sum_{j \in \mathcal{E}(.,i)} f_{ji} \right| - c_i \right] + \phi_\Sigma(0). \quad (4.38)$$

The last equation is obtained by applying the ℓ_1 reformulation presented in 3.4.1 and noting that the terms out of the absolute values cancel out. We also note that the simplified equation of the Maxflow does not contain the flow from the source and the flow to the sink.

Max-flow / min-cut as a reparametrization

Suppose that we obtain dual variables $(f_{ij})_{ij}$ by solving, for instance, the problem (4.37). We can apply the following re-parametrization:

$$c'_i = c_i + \sum_{j \in \mathcal{E}(i,.)} f_{ij} - \sum_{j \in \mathcal{E}(.,i)} f_{ji} \quad (4.39)$$

This gives use the following primal dual problem:

$$E^* = \min_{(x_i \in [0,1])_i} \max_{f'_{ij} \in [-f_{ij}, w_{ij} - f_{ij}]} \sum_{i \in \mathcal{V}} c'_i x_i + \sum_{(i,j) \in \mathcal{E}} f'_{ij} (x_i - x_j) + \phi_\Sigma(0). \quad (4.40)$$

And the following primal problem:

$$E^* = \min_{(x_i \in [0,1])_i} \sum_{i \in \mathcal{V}} c'_i x_i + \sum_{(i,j) \in \mathcal{E}} (w_{ij} - f_{ij}) \max(x_i - x_j, 0) + f_{ij} \max(x_j - x_i, 0) + \phi_\Sigma(0). \quad (4.41)$$

Hence, solving the max-flow problem is equivalent to finding a re-parametrization of the min-cut problem where its sum of unary potentials has a minimum ℓ_1 norm. It is also equivalent to find a re-parametrization of the dual problem where the new optimal dual variables are equal to 0.

Recovering primal variables from dual variables

To obtain optimal primal variables from optimal dual variables we can solve the following problem:

$$\begin{aligned}
\min_{(x_i)} \quad & \sum_{i\in\mathcal{V}} R(x_i) \\
\text{subject to} \quad & x_i \in [0,1]\,, \quad \forall i \in \mathcal{V} \\
& x_i = 0, \qquad \text{if } \; c_i + \sum_{j\in\mathcal{E}(i,.)} f^*_{ij} - \sum_{j\in\mathcal{E}(.,i)} f^*_{ji} > 0 \\
& x_i = 1, \qquad \text{if } \; c_i + \sum_{j\in\mathcal{E}(i,.)} f^*_{ij} - \sum_{j\in\mathcal{E}(.,i)} f^*_{ji} < 0 \\
& x_i = x_j \qquad \text{if } \; f^*_{ij} \in \;]0, w_{ij}[\\
& x_i \leqslant x_j \qquad \text{if } \; f^*_{ji} = 0 \\
& x_i \geqslant x_j \qquad \text{if } \; f^*_{ji} = w_{ij}
\end{aligned} \tag{4.42}$$

where $R(.) : \mathbb{R} \to \mathbb{R}$ is a convex function acting as a regularizer.

We note that if $R(.) = 0$, then the problem (4.42) can be solved very easily by applying a depth first search scheme as described in the Ford-Fulkerson max-flow algorithm as explain in section 26.2 of [36]. For other regularization functions one can use the primal dual techniques of the previous chapter for instance.

Proof: The primal update rule states that $\forall i \in \mathcal{V}$:

$$x^*_i = \left[x^*_i - \tau \left(c_i + \sum_{j\in\mathcal{E}(i,.)} f^*_{ij} - \sum_{j\in\mathcal{E}(.,i)} f^*_{ji} \right) \right]_{[0,1]} \tag{4.43}$$

Hence, we can recover a partial set of optimal primal variables $(x_i)^*$:

$$x^*_i = \begin{cases} 0 & \text{if } \; c_i + \sum_{j\in\mathcal{E}(i,.)} f^*_{ij} - \sum_{j\in\mathcal{E}(.,i)} f^*_{ji} > 0 \\ 1 & \text{if } \; c_i + \sum_{j\in\mathcal{E}(i,.)} f^*_{ij} - \sum_{j\in\mathcal{E}(.,i)} f^*_{ji} < 0 \\ \in [0,1] & \text{if } \; c_i + \sum_{j\in\mathcal{E}(i,.)} f^*_{ij} - \sum_{j\in\mathcal{E}(.,i)} f^*_{ji} = 0 \end{cases} \tag{4.44}$$

The dual update rule states that $\forall\,(i,j) \in \mathcal{E}$:

$$f^*_{ij} = \left[f^*_{ij} + \sigma(x^*_i - x^*_j) \right]_{[0,w_{ij}]} \tag{4.45}$$

Hence, we gather the following additional constraints:

$$\begin{cases} x^*_i = x^*_j & \text{if } \quad f^*_{ij} \in \;]0, w_{ij}[\\ x^*_i \leqslant x^*_j & \text{if } \quad f^*_{ji} = 0 \\ x^*_i \geqslant x^*_j & \text{if } \quad f^*_{ji} = w_{ij} \end{cases} \tag{4.46}$$

Therefore, the optimal primal variable belongs to the space verifying constraints (4.44) and (4.46). Since, this space is not necessarily a singleton, we can enforce a particular solution by adding a regularization cost. This concludes the proof.

Recovering dual variables from primal variables

To obtain optimal dual variables from optimal primal variables we can solve the following problem:

$$\begin{aligned}
\min_{(f_{ij})} \quad & \sum_{(i,j)\in\mathcal{E}} R(f_{ij}) \\
\text{subject to} \quad & f_{ij} \in [0, w_{ij}], && \forall\,(i,j)\in\mathcal{E} \\
& f_{ij} = w_{ij}, && \text{if } \; x_i^* - x_j^* > 0 \\
& f_{ij} = 0, && \text{if } \; x_i^* - x_j^* < 0 \\
& c_i + \sum_{j\in\mathcal{E}(i,.)} f_{ij} - \sum_{j\in\mathcal{E}(.,i)} f_{ji} < 0, && \text{if } \; x_i^* > 0 \\
& c_i + \sum_{j\in\mathcal{E}(i,.)} f_{ij} - \sum_{j\in\mathcal{E}(.,i)} f_{ji} > 0, && \text{if } \; x_i^* = 0
\end{aligned} \tag{4.47}$$

where $R(.) : \mathbb{R} \to \mathbb{R}$ is a convex function acting as a regularizer.

The problem (4.47) can be solved with primal dual techniques presented in the previous chapter.

Proof: The dual update rule states that $\forall\,(i,j)\in\mathcal{E}$:

$$f_{ij}^* = \left[f_{ij}^* + \sigma(x_i^* - x_j^*)\right]_{[0,w_{ij}]} \tag{4.48}$$

Hence, we can easily derive a partial set of dual variables:

$$f_{ij}^* \in \begin{cases} \{w_{ij}\} & \text{if} \quad x_i^* - x_j^* > 0, \\ \{0\} & \text{if} \quad x_i^* - x_j^* < 0, \\ [0, w_{ij}] & \text{if} \quad x_i^* - x_j^* = 0. \end{cases} \tag{4.49}$$

However, for any couple $(i,j)\in\mathcal{E}$ where $x_i^* = x_j^*$ we have to find another set of constraints.

The primal update rule states that $\forall i \in \mathcal{V}$:

$$x_i^* = \left[x_i^* - \tau\left(c_i + \sum_{j\in\mathcal{E}(i,.)} f_{ij}^* - \sum_{j\in\mathcal{E}(.,i)} f_{ji}^*\right)\right]_{[0,1]} \tag{4.50}$$

Therefore, we deduce additional constraints for $(f_{ij})_{ij}$:

$$\begin{aligned}
& c_i + \sum_{j\in\mathcal{E}(i,.)} f_{ij}^* - \sum_{j\in\mathcal{E}(.,i)} f_{ji}^* < 0, \quad \text{if} \quad x_i^* > 0 \\
& c_i + \sum_{j\in\mathcal{E}(i,.)} f_{ij}^* - \sum_{j\in\mathcal{E}(.,i)} f_{ji}^* > 0, \quad \text{if} \quad x_i^* = 0
\end{aligned} \tag{4.51}$$

Hence, the optimal dual variable belongs to the space verifying constraints (4.49) and (4.51). Since, this space is not necessarily a singleton, we can enforce a particular solution by adding a regularization cost. This concludes the proof.

Recovering the flow from the source and to the sink

The optimal flow from the source and to the sink is given by:

$$f_{it}^* = \min(\phi_i(0) - \sum_{j \in \mathcal{E}(i,.)} f_{ij}^* + \sum_{j \in \mathcal{E}(.,i)} f_{ji}^*, \quad \phi_i(1)) \tag{4.52}$$

$$f_{si}^* = \min(\phi_i(1) + \sum_{j \in \mathcal{E}(i,.)} f_{ij}^* - \sum_{j \in \mathcal{E}(.,i)} f_{ji}^*, \quad \phi_i(0)) \tag{4.53}$$

Proof: For each vertex $i \in \mathcal{V}$ we know from equation (4.28) that:

$$f_{it}^* - f_{si}^* + \sum_{j \in \mathcal{E}(i,.)} f_{ij}^* - \sum_{j \in \mathcal{E}(.,i)} f_{ji}^* = 0 \tag{4.54}$$

$$f_{si}^* - f_{it}^* = \sum_{j \in \mathcal{E}(i,.)} f_{ij}^* - \sum_{j \in \mathcal{E}(.,i)} f_{ji}^* \tag{4.55}$$

$$f_{si}^* - f_{it}^* = f_{sit}^* \tag{4.56}$$

Since we already know the optimal flow between vertices, the canonical maxflow problem simplifies to:

$$\begin{array}{lll} \max\limits_{(f_{si}),(f_{it})} & \sum\limits_{i \in \mathcal{V}} f_{si} & \\ \text{subject to} & f_{si} \leqslant \phi_i(0), & \forall i \in \mathcal{V} \\ & f_{it} \leqslant \phi_i(1), & \forall i \in \mathcal{V} \\ & f_{si} = f_{sit}^* + f_{it}, & \forall i \in \mathcal{V} \end{array} \tag{4.57}$$

We can easily solve for all (f_{si}):

$$\begin{array}{lll} \max\limits_{(f_{it})} & \sum\limits_{i \in \mathcal{V}} f_{sit}^* + f_{it} & \\ \text{subject to} & f_{sit}^* + f_{it} \leqslant \phi_i(0), & \forall i \in \mathcal{V} \\ & f_{it} \leqslant \phi_i(1), & \forall i \in \mathcal{V} \end{array} \tag{4.58}$$

We can reformulate to:

$$\begin{array}{lll} \max\limits_{(f_{it})} & \sum\limits_{i \in \mathcal{V}} f_{sit}^* + f_{it} & \\ \text{subject to} & f_{it} \leqslant \phi_i(0) - f_{sit}^*, & \forall i \in \mathcal{V} \\ & f_{it} \leqslant \phi_i(1), & \forall i \in \mathcal{V} \end{array} \tag{4.59}$$

Finally, we obtain:

$$f_{it}^* = \min(\phi_i(0) - f_{sit}^*, \quad \phi_i(1)) \tag{4.60}$$

$$f_{si}^* = \min(\phi_i(1) + f_{sit}^*, \quad \phi_i(0)) \tag{4.61}$$

This completes the proof.

4.3.4 Characterization of obvious partial primal solutions

We study some sufficient conditions for a primal variable x_k to be equal to either 0 or 1. One can show that for a given vertex $k \in \mathcal{V}$, there exists an optimal solution that verifies:

$$x_k^* = \begin{cases} 0 & \text{if} \quad c_k \geqslant \sum_{j \in \mathcal{E}(.,k)} w_{jk} \\ 1 & \text{if} \quad c_k \leqslant -\sum_{j \in \mathcal{E}(k,.)} w_{kj} \end{cases} \tag{4.62}$$

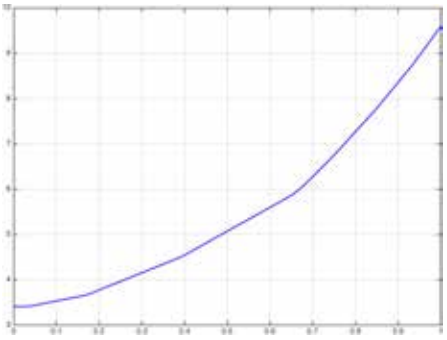


Figure 4.1 – Example of function $f(x_k)$ for $c_k \geqslant \sum_{j \in \mathcal{E}(.,k)} w_{jk}$.

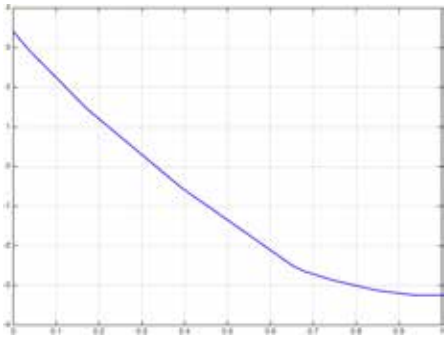


Figure 4.2 – Example of function $f(x_k)$ for $c_k \leqslant -\sum_{j \in \mathcal{E}(k,.)} w_{kj}$.

Proof for $x_k^* = 0$

Let us assume that $c_k \geqslant \sum_{j \in \mathcal{E}(.,k)} w_{jk}$. We can write for a certain $\delta^+ \in \mathbb{R}^+$:

$$c_k = \delta^+ + \sum_{j \in \mathcal{E}(.,k)} w_{jk} \tag{4.63}$$

We apply the reformulation trick of (3.59) introduced in previous chapter, we have:

$$f(x_k) = c_k x_k + \sum_{j \in \mathcal{E}(.,k)} w_{jk} \max(x_j - x_k, 0) + \sum_{j \in \mathcal{E}(k,.)} w_{kj} \max(x_k - x_j, 0) \tag{4.64}$$

$$= c_k x_k + \sum_{j \in \mathcal{E}(.,k)} w_{jk} (x_j - x_k) + \sum_{j \in \mathcal{E}(k,.)} (w_{kj} + w_{jk}) \max(x_k - x_j, 0) \tag{4.65}$$

$$= \delta^+ x_k + \sum_{j \in \mathcal{E}(k,.)} (w_{kj} + w_{jk}) \max(x_k - x_j, 0) + \sum_{j \in \mathcal{E}(.,k)} w_{jk} x_j \tag{4.66}$$

Hence, $f(x_k)$ is an increasing function and we have $x_k^* = 0$ an optimal solution. This is illustrated by figure 4.1.

Proof for $x_k^* = 1$

Let us assume that $c_k \leqslant -\sum_{j\in\mathcal{E}(k,.)} w_{kj}$. We can write for a certain $\delta^- \in \mathbb{R}^-$:

$$c_k = \delta^- - \sum_{j\in\mathcal{E}(k,.)} w_{kj} \tag{4.67}$$

We apply the reformulation trick of (3.59) introduced in previous chapter:

$$f(l_k) = c_k x_k + \sum_{j\in\mathcal{E}(.,k)} w_{jk} \max\left(x_j - x_k, 0\right) + \sum_{j\in\mathcal{E}(k,.)} w_{kj} \max\left(x_k - x_j, 0\right) \tag{4.68}$$

$$= c_k x_k + \sum_{j\in\mathcal{E}(.,k)} \left(w_{jk} + w_{kj}\right) \max\left(x_j - x_k, 0\right) + \sum_{j\in\mathcal{E}(k,.)} w_{kj} \left(x_k - x_j\right) \tag{4.69}$$

$$= \delta^- x_k + \sum_{j\in\mathcal{E}(.,k)} \left(w_{jk} + w_{kj}\right) \max\left(x_j - x_k, 0\right) - \sum_{j\in\mathcal{E}(k,.)} w_{kj} x_j \tag{4.70}$$

Hence, $f(x_k)$ is an decreasing function and we have $x_k^* = 1$ an optimal solution. This is illustrated by figure 4.2.

4.3.5 ROF and Maxflow

The previous chapter gives us a surrogate set of problems that give an optimal solution for the max-flow. Indeed, we see that the optimal solution space of the simplified max-flow problem (4.38) corresponds to $\mathcal{Y}_{\ell_1}^*$ of theorem (4). The theorem (4) guaranties any optimal dual solution of ROF model also gives a solution of the associated max-flow problem. This was previously observed in [26].

Hence, we can directly solve the following ROF in its dual form:

$$\max_{(f_{ij}\in[0,w_{ij}])_{ij}} -\frac{1}{2}\sum_{i\in\mathcal{V}} \left(c_i + \sum_{j\in\mathcal{E}(i,.)} f_{ij} - \sum_{j\in\mathcal{E}(.,i)} f_{ji} \right)^2 \tag{4.71}$$

However, if we solve the primal form of the ROF model:

$$\min_{(x_i\in\mathbb{R})_i} \sum_{i\in\mathcal{V}} \frac{1}{2} x_i^2 + c_i x_i + \sum_{(i,j)\in\mathcal{E}} w_{ij} \max\left(x_i - x_j,\ 0\right), \tag{4.72}$$

we can then recover the optimal dual variable by slightly adjusting problem (4.47)

to:

$$\begin{aligned}
\min_{(f_{ij})} \quad & \sum_{(i,j)\in\mathcal{E}} R(f_{ij}) & & \\
\text{subject to} \quad & f_{ij} \in [0, w_{ij}], & & \forall\,(i,j)\in\mathcal{E} \\
& f_{ij} = w_{ij}, & & \text{if } \; x_i^* - x_j^* > 0 \\
& f_{ij} = 0, & & \text{if } \; x_i^* - x_j^* < 0 \\
& c_i + \sum_{j\in\mathcal{E}(i,.)} f_{ij} - \sum_{j\in\mathcal{E}(.,i)} f_{ji} = -x_i^{ROF,*}, & & \forall i \in \mathcal{V}
\end{aligned} \tag{4.73}$$

where $R(.) : \mathbb{R} \to \mathbb{R}$ is a convex function acting as a regularizer.

4.4 Solvers for min-cut / max-flow problems

We now present solvers that optimally solve the min-cut / max-flow problem.

4.4.1 Solver for chain graphs

We devise a first solver when the graph $\mathcal{G}$ is a chain of C vertices. We describe algorithms that solve in $O(C)$ the primal and dual forms.

Primal algorithm

In its primal form, we need to optimize the following problem:

$$E^* = \min_{(x_i\in\{0,1\})_i} \sum_{i\in\{1,..,N\}} c_i x_i + \sum_{i\in\{2,..,C\}} w_i \max\left(x_{i+1} - x_i, 0\right) + \phi_\Sigma(0). \tag{4.74}$$

The problem (4.74) can be solved by the Viterbi algorithm [72], a dynamic programing algorithm [44]. We present in algorithm 9 a simplified version where the inner loops are unrolled since we work with binary variables.

Dual algorithm

For the dual form, we have to solve:

$$E^* = \max_{f_i\in[0,w_i]} -\frac{1}{2}\left[\sum_{i\in\{1,..,C\}} |c_i + f_i - f_{i-1}| - c_i\right] + \phi_\Sigma(0). \tag{4.75}$$

where we introduce $f_0 = 0$ and $w_0 = 0$ to ease the notations.

The problem (4.75) can also be solved by dynamic programing as in algorithm 10.

Algorithm 9: Primal algorithm for min-cut on chain graph

Data: Inputs: $(c_i)_i$, $(w_i)_i$
Result: $(x_i)_i$
Initialization: $c_{n0} = \{0, c_1\}$
Message passing on chain:
for $i \in \{2, \dots, C\}$ **do**
  Compute minimum cost for transition to next node:
  if $c_{n0}(1) < c_{n0}(2) + w_{i-1}$ **then**
    $c_{n1}(1) \leftarrow c_{n0}(1)$
    $\mathit{labels}(i-1, 1) \leftarrow 0$
  else
    $c_{n1}(1) \leftarrow c_{n0}(2) + w_{i-1}$
    $\mathit{labels}(i-1, 1) \leftarrow 1$
  if $c_{n0}(1) < c_{n0}(2)$ **then**
    $c_{n1}(2) \leftarrow c_{n0}(1)$
    $\mathit{labels}(i-1, 2) \leftarrow 0$
  else
    $c_{n1}(2) \leftarrow c_{n0}(2)$
    $\mathit{labels}(i-1, 2) \leftarrow 1$
  Add unary term: $c_{n1}(2) \leftarrow c_{n1}(2) + c_i$
  Copy temporary buffer: $c_{n0} \leftarrow c_{n1}$
Decoding:
if $c_{n0}(1) < c_{n0}(2)$ **then**
  $x_C = 0$
else
  $x_C = 1$
for $i \in \{C-1, \dots, 1\}$ **do**
  $x_i = \text{labels}(i, x_{i+1} + 1)$

Algorithm 10: Algorithm for max-flow on chain graph

Data: Inputs: $(c_i)_i$, $(w_i)_i$
Result: $(f_i)_i$
for $i = 1, C-1$ **do**
  Compute excess of flow:
  $$e \leftarrow c_i - f_{i-1}$$
  if $e \leqslant 0$ **then**
    $f_i \leftarrow \min(w_i, -e)$
  else
    $f_i \leftarrow 0)$

Dual algorithm with backtracking

We can even improve the dual algorithm to also return the optimal primal variables. To this end, we run the depth-first-search algorithm while computing the dual variables. As soon as we saturate a dual variable, we can backtrack to label the primal variables accordingly. This leads to algorithm 11.

Algorithm 11: Algorithm for max-flow on chain graph with backtracking

Data: Inputs: $(c_i)_i$, $(w_i)_i$
Result: $(f_i)_i$
Initialization for backtracking $s \leftarrow 1$
backtrack $\leftarrow 0$
for $i = 1, C-1$ **do**
  Compute excess of flow:
  $$e \leftarrow c_i - f_{i-1}$$
  if $e \leqslant 0$ **then**
    if $w(i) < -e$ **then**
      $f_i \leftarrow w_i$
      $x_i \leftarrow 1$
      backtrack $\leftarrow 1$
    else
      $f_i \leftarrow -e$
  else
    $f_i \leftarrow 0$
    $x_i \leftarrow 0$
    backtrack $\leftarrow 1$
  Backtrack to set optimal primal variables:
  if *backtrack* **then**
    for $j \in \{i-1, \dots, s\}$ **do**
      $x_j \leftarrow x_{j+1}$
    $s \leftarrow i+1$
    backtrack $\leftarrow 0$

4.4.2 Iterative solvers

TV-linear

We know focus on general graph. The primal dual formulation of the min-cut / max-flow problem is given by:

$$E^* = \min_{x_i \in [0,1]} \max_{f_{ij} \in [0, w_{ij}]} \sum_{i \in \mathcal{V}} c_i x_i + \sum_{(i,j) \in \mathcal{E}} f_{ij} (x_i - x_j) + \phi_\Sigma(0). \tag{4.76}$$

We can apply the primal dual techniques of the last chapter to solve the problem (4.76) using algorithm (12). This techniques returns both the (relaxed)

primal and dual variables. It can also be warm-started when we dispose of an initial guess of the primal and dual variables.

The convergence rate to an optimal solution in guaranteed in $O\left(\frac{1}{N}\right)$, where N is the number of iterations. However, the algorithm (12) does neither guaranty to decrease the primal energy nor increase the dual energy at each iteration. The algorithm 12 describes the TV-linear method for solving the maxflow / mincut problem.

Algorithm 12: Primal dual algorithm for min-cut / max-flow as TV-linear problem

Data: Inputs: $(c_i)_i$, $(w_{ij})_{ij}$, $\mathcal{G}$, τ, σ
Result: $(x_i)_i$, $(f_{ij})_{ij}$
Initialize primal and dual variable $\rightarrow x^0 = 0$, $f^0 = 0$.
Set $\tilde{x} = x^0$
while *Stopping criterion is not verified* **do**
Optimize the dual variables, $\forall (i,j) \in \mathcal{E}$:

$$f_{ij}^{n+1} = \left[f_{ij}^n + \sigma(\tilde{x}_i - \tilde{x}_j)\right]_{[0,w_{ij}]} \tag{4.77}$$

Optimize the primal variables, $\forall i \in \mathcal{V}$:

$$x_i^{n+1} = \left[x_i^n - \tau\left(c_i + \sum_{j \in \mathcal{E}(i,.)} f_{ij} - \sum_{j \in \mathcal{E}(.,i)} f_{ji}\right)\right]_{[0,1]} \tag{4.78}$$

Smooth variable, $\forall i \in \mathcal{V}$:

$$\tilde{x}_i = x_i^{n+1} + \theta\left(x_i^{n+1} - x_i^n\right) \tag{4.79}$$

TV-l2: ROF

We have demonstrated that the ROF model gives an alternative approach to solve the min-cut / max-flow problem. Hence, we make use of the primal-dual form of the ROF model:

$$\min_{(x_i \in \mathbb{R})_i} \max_{f_{ij} \in [0,w_{ij}]} \sum_{i \in \mathcal{V}} \frac{1}{2} x_i^2 + c_i x_i + \sum_{(i,j) \in \mathcal{E}} f_{ij}\left(x_i - x_j\right) \tag{4.80}$$

As for the TV-linear problem, we apply the primal dual technique of the last chapter. Therefore, we directly recover optimal dual variables that solve the max-flow problem (from which we can easily obtain optimal binary primal variables). The algorithm 13 describes the TV-ℓ_2 method for solving maxflow / mincut problems.

Algorithm 13: Primal dual algorithm for min-cut / max-flow as TV-ℓ_2 problem

Data: Inputs: $(c_i)_i$, $(w_{ij})_{ij}$, $\mathcal{G}$, τ, σ
Result: $(x_i)_i$, $(f_{ij})_{ij}$
Initialize primal and dual variable $\rightarrow x^0 = 0$, $f^0 = 0$.
Set $\tilde{x} = x^0$
while *Stopping criterion is not verified* **do**

Optimize the dual variables, $\forall (i,j) \in \mathcal{E}$:

$$f_{ij}^{n+1} = \left[f_{ij}^n + \sigma_n (\tilde{x}_i - \tilde{x}_j) \right]_{[0, w_{ij}]} \tag{4.81}$$

Optimize the primal variables, $\forall i \in \mathcal{V}$:

$$x_i^{n+1} = \frac{x_i^{n+1} + \tau_n \left(c_i + \sum_{j \in \mathcal{E}(i,.)} f_{ij} - \sum_{j \in \mathcal{E}(.,i)} f_{ji} \right)}{1 + \tau_n} \tag{4.82}$$

Update the smoothing and steps size parameters:

$$\theta_n = \frac{1}{\sqrt{1 + 2\gamma\tau_n}}, \quad \tau_{n+1} = \theta_n \tau_n, \quad \sigma_{n+1} = \frac{\sigma_n}{\theta_n} \tag{4.83}$$

Smooth variable, $\forall i \in \mathcal{V}$:

$$\tilde{x}_i = x_i^{n+1} + \theta_n \left(x_i^{n+1} - x_i^n \right) \tag{4.84}$$

For algorithm (13) the convergence rate to an optimal solution in guaranteed in $O\left(\frac{1}{N^2}\right)$, where N is the number of iterations.

4.4.3 Augmenting path solvers

Augmenting path solvers [147] are algorithms designed to solve the dual form of the max-flow / min-cut, i.e., problem (4.28). Here, we describe the augmenting path method of [17] that experimentally outperforms other variant for computer vision problems [166]. It sequentially alternates three stages:

1. Find an augmenting path: a chain subgraph $\mathcal{A}$ of the main graph $\mathcal{G}$ whose dual variables can be further optimized. For computer vision problems, this is generally done by growing two binary trees simultaneously from the source and sink by following edges that have positive capacities. An augmenting path is found when the two trees meet. If no augmenting path can be found, then the algorithm terminates.

2. The augmenting phase: optimize the dual variables of the chain subgraph $\mathcal{A}$. Apply a re-parametrization of the problem as describe in paragraph 4.3.3.

3. Adopt orphan: An optional step that greatly speeds up augmenting path is to update dynamically the growing trees after the augmentation phase. During the augmentation the re-parametrization might have set the capacities of some edge to 0, potentially invalidating parts of the source and sink trees. The adoption phase corrects this phenomenon.

The efficiency of augmenting path algorithms resides in their ability to quickly find augmenting paths. The celebrated BK algorithm uses a heuristic that deliver great performance for computer vision related maxflow problems. We refer the reader to its original publication [17] for more details.

4.5 Graph-cuts for non convex problems

We now look at graph-cuts techniques to optimize a sub class of non convex problems given by equation (4.3):

$$(x_i^*)_i = \underset{(x_i \in \mathcal{L})_i}{\arg\min} \quad \sum_{i \in \mathcal{V}} \phi_i(x_i) + \sum_{(i,j) \in \mathcal{E}} \phi_{ij}(x_i, x_j).$$

Unfortunately, such problems are known to be NP-hard [101]. Hence, theoretically the best we can get is an approximate solution with some mathematical guaranties on the goodness of the approximation.

4.5.1 Alpha-expansion

Overview of the method

A first technique to optimize equation (4.3) is to use the alpha expansion algorithm [20]. Instead of optimizing for all possible labels at the same time, the alpha expansion selects at a given time a unique label, that we refer to as alpha. Then, it tries to substitute (expand) the alpha label to some variables x_i of the current solution. Since the goal remains to decrease the energy represented by equation (4.3), the expansion is obtained by solving a binary problem where the zero values represent the choice to keep the current labeling and the one values indicate substitution by the alpha label.

The Alpha-expansion method requires that the functions $(\phi_{ij}(.,.))_{ij}$ are metric:

$$\begin{aligned} \phi_{ij}(a,a) &= 0, & \forall a \in \mathcal{L} \\ \phi_{ij}(a,b) &\geqslant 0, & \forall (a,b) \in \mathcal{L} \times \mathcal{L} \\ \phi_{ij}(a,c) &\leqslant \phi_{ij}(a,b) + \phi_{ij}(b,c) & \forall (a,b,c) \in \mathcal{L} \times \mathcal{L} \times \mathcal{L} \end{aligned}$$

Expansion as a maxflow problem

Let us assume we are given a current solution $(x_i)_i$ and a label to expand $\alpha \in \mathcal{L}$. We make use of the following binary functions: for each $i \in \mathcal{V}$:

$$\begin{aligned} \psi_i(0) &= \phi_i(x_i) \\ \psi_i(1) &= \phi_i(\alpha) \end{aligned}$$

for each $(i,j) \in \mathcal{E}$:

$$\psi_{ij} = \begin{vmatrix} \phi_{i,j}(x_i, x_j) & \phi_{i,j}(x_i, \alpha) \\ \phi_{i,j}(\alpha, x_j) & \phi_{i,j}(\alpha, \alpha) \end{vmatrix}$$

The functions $(\psi_i)_i$ are obviously submodular and since the functions $(\phi_{ij})_{ij}$ are metric, they make the functions $(\psi_{ij})_{ij}$ submodular:

$$\phi_{i,j}(x_i, x_j) + \phi_{i,j}(\alpha, \alpha) \leqslant \phi_{i,j}(x_i, \alpha) + \phi_{i,j}(\alpha, x_j)$$

Finally the binary function

$$\psi(z) = \sum_{i \in \mathcal{V}} \psi_i(z_i) + \sum_{(i,j) \in \mathcal{E}} \psi_{ij}(z_i, z_j)$$

is submodular thanks to property (4).

The theorem (10) informs us that optimizing $\psi(.)$ is equivalent to solving a mincut-maxflow problem. Hence, we can solve the expansion in polynomial time.

Alpha expansion algorithm

We now have a sub-routine to perform the expansion. The alpha expansion algorithm simply cycles through each possible label of $\mathcal{L}$ until no expansion changes the current solution. We summarize this method in algorithm 14.

Algorithm 14: Alpha expansion

Data: Inputs: $(\phi_i)_i$, $(\phi_{ij})_{ij}$, $\mathcal{G}$, $(x_i)_i$
Result: $(x_i)_i$
while *Until no expansion improves the solution* **do**
  for $\alpha \in \mathcal{L}$ **do**
    Create binary function for expansion:

$$\psi_i = \begin{vmatrix} \phi_i(x_i) \\ \phi_i(\alpha) \end{vmatrix} \quad \text{and} \quad \psi_{ij} = \begin{vmatrix} \phi_{i,j}(x_i, x_j) & \phi_{i,j}(x_i, \alpha) \\ \phi_{i,j}(\alpha, x_j) & \phi_{i,j}(\alpha, \alpha) \end{vmatrix}$$

    Solve mincut - maxflow problem using the representation forumula (4.5) of theorem (10):

$$z^* = \arg\min_{z_i\{0,1\}} \sum_{i \in \mathcal{V}} \psi_i(z_i) + \sum_{(i,j) \in \mathcal{E}} \psi_{ij}(z_i, z_j)$$

    Update current solution:

$$x_i = (1 - z_i^*)x_i + z_i^* \alpha, \quad \forall i \in \mathcal{V}.$$

4.5.2 Fast-PD

Overview of the method

We now move on to a different graph-cut technique: Fast-PD [102]. Fast-PD relies on first transforming the original multi-labels problem (4.3) to a binary linear programming problem as in [63] or [100]. Then, it makes use of the approximate primal dual scheme presented at the beginning of this chapter to derive a sub-optimal solution. Fast-PD shares similarity with the alpha expansion, since it also relies on solving a series of maxflow problems in a cycling fashion. However, Fast-PD can handle a larger set of pairwise functions since it only requires the following assumptions on the functions $(\phi_{ij}(.,.))_{ij}$:

$$\begin{aligned} \phi_{ij}(a,b) = 0 \Leftrightarrow a = b \qquad & \forall (a,b) \in \mathcal{L} \times \mathcal{L} \\ \phi_{ij}(a,b) \geqslant 0, \qquad & \forall (a,b) \in \mathcal{L} \times \mathcal{L} \end{aligned}$$

A binary LP formulation

As preliminary work, we formulate problem (4.3):

$$(x_i^*)_i = \arg\min_{(x_i \in \mathcal{L})_i} \quad \sum_{i \in \mathcal{V}} \phi_i(x_i) + \sum_{(i,j) \in \mathcal{E}} \phi_{ij}(x_i, x_j).$$

as a binary linear programing problem.

Potentials: We make use of vectors $(\theta_i)_i$ to represent the value of each discrete function $\phi_i(.)$ and vectors $(\theta_{ij})_i$ to represent the $\phi_{ij}(.,.)$:

$$\begin{aligned}\theta_i &= [\phi_i(0), \phi_i(1), \ldots, \phi_i(L-2), \phi_i(L-1)] \\ \theta_{ij} &= [\phi_{ij}(0,0), \ldots, \phi_{ij}(0,L-1), \ldots, \phi_{ij}(L-1,0), \ldots \phi_{ij}(L-1,L-1)] \\ \boldsymbol{\theta} &= [\theta_0, \ldots, \theta_i, \ldots, \theta_N, \ldots, \theta_{ij}, \ldots]\end{aligned}$$

Each vector θ_i has length of L and each vector θ_{ij} has L^2 elements. This makes the length of vector $\boldsymbol{\theta}$ equal to $NL + EL^2$.

Variables: We also make use of binary indicator variables $(z_i)_i$ and $(z_{ij})_{ij}$ to represent a solution x:

$$\begin{aligned}z_i &= \left[z_{i,0}, z_{i,1}, \ldots, z_{i,L-1}, z_{i,L}\right] \\ z_{ij} &= \left[z_{ij,0,0}, \ldots, z_{ij,0,L}, z_{ij,1,0}, \ldots, z_{ij,L-1,L}, z_{ij,L,0}, \ldots z_{ij,L,L}\right] \\ \boldsymbol{z} &= \left[z_0, \ldots, z_i, \ldots, z_N, \ldots, z_{ij}, \ldots\right]\end{aligned}$$

We need to add constraint to the solution space of $\boldsymbol{z}$. The uniqueness of label assignment constraint forces that an unique element of each z_i is equal to 1:

$$\sum_{l \in \mathcal{L}} z_{i,l} = 1, \quad \forall i \in \mathcal{V}$$

We make use of a sparse matrix U of size $N \times (NL + EL^2)$ to enforce the uniqueness constraints.

The consistency constraints enforce for any edge (i,j) and any label $(a,b) \in \mathcal{L} \times \mathcal{L}$ that if $z_{i,a} = 1$ and $z_{j,b} = 1$ then, we need to have $z_{ij,a,b} = 1$ and all other elements of z_{ij} must be equal to 0:

$$\sum_{l \in \mathcal{L}} z_{ij,a,l} - z_{i,a} = 0 \quad \text{and} \quad \sum_{l \in \mathcal{L}} z_{ij,l,b} - z_{j,b} = 0$$

We make use of sparse matrix C of size $2EL \times (NL + EL^2)$ to enforce the consistency constraint.

Binary problem: Putting everything together we get the following binary Linear Programming problem:

$$\begin{aligned}(z_i^*)_i = \underset{\boldsymbol{z} \in \{0,1\} \times \ldots \times \{0,1\}}{\arg\min} \quad & \boldsymbol{\theta}^T \boldsymbol{z} \\ s.t. \quad & U\boldsymbol{z} = \mathbf{1}, \quad \text{(uniqueness constraints)} \\ & C\boldsymbol{z} = \mathbf{0}, \quad \text{(consistency constraints)}\end{aligned}$$

A primal LP formulation

Solving the previous binary LP problem is very challenging. Hence, Fast-PD proceeds with the primal dual approximation framework by relaxing the domain of variables $\boldsymbol{z}$ to the real positive half space. Since, we also have the uniqueness constraints the feasible domain of each element of $\boldsymbol{z}$ is in fact $[0, 1]$. We get the following LP that we refer as the primal form of our problem:

$$\begin{aligned} \min_{\boldsymbol{z}} \quad & \boldsymbol{\theta}^T \boldsymbol{z} \\ s.t. \quad & U\boldsymbol{z} = \mathbf{1}, \quad \text{(uniqueness constraints)} \\ & C\boldsymbol{z} = \mathbf{0}, \quad \text{(consistency constraints)} \\ & \boldsymbol{z} \geqslant \mathbf{0} \end{aligned}$$

A dual LP formulation

We proceed to a second transformation by computing the dual of the primal LP. To this end, we introduce a set of NL variables, $\boldsymbol{s}$, for the uniqueness constraints. We use a set of $2EL$ variables $\boldsymbol{y}$ for the consistency constraints. The dual form of the LP problem is given by:

$$\begin{aligned} \max_{\boldsymbol{s},\boldsymbol{y}} \quad & \begin{bmatrix} \mathbf{1} \\ \mathbf{0} \end{bmatrix}^T \begin{bmatrix} \boldsymbol{s} \\ \boldsymbol{y} \end{bmatrix} \\ s.t. \quad & \begin{bmatrix} U \\ C \end{bmatrix}^T \begin{bmatrix} \boldsymbol{s} \\ \boldsymbol{y} \end{bmatrix} \leqslant \boldsymbol{\theta} \end{aligned}$$

that we can simplify to:

$$\begin{aligned} \max_{\boldsymbol{s},\boldsymbol{y}} \quad & \sum_{i \in \mathcal{V}} s_i \\ s.t. \quad & s_i \leqslant \theta_{i,a} + \sum_{j \in \mathcal{E}(i)} y_{ij,a}, \quad \forall (i,a) \in \mathcal{V} \times \mathcal{L} \\ & y_{ij,a} + y_{ji,b} \leqslant \theta_{ij,a,b} \quad \forall (ij,a,b) \in \mathcal{E} \times \mathcal{L} \times \mathcal{L} \end{aligned}$$

where $\mathcal{E}(i) = \mathcal{E}(i,.) \cup \mathcal{E}(.,i)$ is the set of all nodes of $\mathcal{V}$ linked to i by an edges in $\mathcal{E}$.

Using the relaxed complementary slackness conditions

Since we work in an approximate primal dual scheme, we only make use of binary primal solution. Assuming we start from a feasible binary primal solution, it is trivial during any primal update to maintain primal feasibility.

Hence, we can simplify the primal relaxed complementary slackness conditions associated to variables $(z_i)_i$ to:

$$z_{i,l} = 1 \quad \Rightarrow \quad \frac{\theta_{i,l}}{\alpha_1} + \sum_{j \in \mathcal{E}(i)} y_{ij,l} \leqslant s_i \leqslant \theta_{i,l} + \sum_{j \in \mathcal{E}(i)} y_{ij,l}, \quad \forall a \in \mathcal{L}$$

We can simplify the primal relaxed complementary slackness conditions associated to variables $(z_{ij})_{ij}$ to:

$$z_{i,a} = 1 \text{ or } \; z_{j,b} = 1 \quad \Rightarrow \quad \frac{\theta_{ij,a,b}}{\alpha_2} \leqslant y_{ij,a} + y_{ji,b} \leqslant \theta_{ij,a,b}, \quad \forall (a,b) \in \mathcal{L} \times \mathcal{L}$$

Due to the assumption on the regularization functions we have $\theta_{ij,l,l} = \phi_{ij}(l,l) = 0$ for any $l \in \mathcal{L}$. Hence, we get:

$$z_{i,l} = 1 \text{ and } \; z_{j,l} = 1 \quad \Rightarrow \quad y_{ij,l} + y_{ji,l} = 0, \quad \forall l \in \mathcal{L}$$

In the approximate primal dual framework, we seek to find a pair of binary primal and dual feasible solutions that satisfy these slackness conditions for some values of α_1 and α_2.

By setting $y_{ij,a} = -y_{ji,a}$ for any pair $(a,b) \in \mathcal{L} \times \mathcal{L}$ and any edge $(i,j) \in \mathcal{E}$, we simplify the optimization problem and the dual variables are now always guaranteed to verify the last complementary slackness condition.

Loosen up the dual constraints

A key observation is to note that we can always easily obtain dual feasible variables from in-feasible dual variables, i.e. variables not verifying their associated constraints, as long as the in-feasibility is bounded. Obtaining dual feasible variables from in-feasible dual variables is often refereed as dual-fitting by the LP community. The dual-fitting operation simply divides the dual variables by a certain amount to make them feasible.

The authors of Fast-PD make use of this scheme. They seek a solution that enforces the primal complementary slackness conditions with $\alpha_1 = 1$ and $\alpha_2 = 1$ while only imposing the dual constraints on variables $\boldsymbol{s}$ and simply controlling the in-feasibility of dual variables $\boldsymbol{y}$. Hence, they look for primal dual solution verifying:

$$\begin{aligned}
(x_i^*)_i &= \arg\min_{x_i \in \mathcal{L}} \sum_{i \in \mathcal{V}} \theta_{i,x_i} + \sum_{j \in \mathcal{E}(i)} y_{ij,x_i} \\
y_{ij,x_i^*} + y_{ji,x_j^*} &= \theta_{ij,x_i^*,x_j^*} \\
y_{ij,a} + y_{ji,b} &\leqslant 2 \max_{(c,d) \in \mathcal{L} \times \mathcal{L}} \theta_{ij,c,d} \quad \forall (ij,a,b) \in \mathcal{E} \times \mathcal{L} \times \mathcal{L}
\end{aligned}$$

Once such a solution is found, one can simply apply dual fitting by scaling each y_{ij} with factor $\dfrac{1}{\delta_{ij}}$ with:

$$\delta_{ij} = \max_{(a,b) \in \mathcal{L} \times \mathcal{L}, a \neq b} \frac{y_{ij,a} + y_{ji,b}}{\theta_{ij,a,b}}$$

It turns out that the fitted dual solution verifies the relaxed complementary slackness conditions with $\alpha_1 = \delta$ and $\alpha_2 = \delta$ for $\delta = \max_{(i,j) \in \mathcal{E}} \delta_{ij}$. Hence, the

primal solution is a δ-approximation of the optimal primal solution. In the original publication the authors derive different approximation bounds of Fast-PD for regularization functions such as weighted Potts or weighted ℓ_1. While those bounds are quite large, they show that in practice the approximation is less than 1 percent.

A primal dual problem

We now have to optimize the following problem:

$$\begin{aligned}(x_i^*)_i = \underset{(x_i \in \mathcal{L})_i}{\arg\min} \max_{\boldsymbol{y}} \quad & \sum_{i \in \mathcal{V}} \theta_{i,x_i} + \sum_{j \in \mathcal{E}(i)} y_{ij,x_i} \\ s.t. \quad & y_{ij,x_i^*} + y_{ji,x_j^*} = \theta_{ij,x_i^*,x_j^*}, \quad \forall (i,j) \in \mathcal{E} \\ & y_{ij,a} + y_{ji,b} \leqslant 2 \max_{(c,d) \in \mathcal{L} \times \mathcal{L}} \theta_{ij,c,d} \quad \forall (ij,a,b) \in \mathcal{E} \times \mathcal{L} \times \mathcal{L}\end{aligned}$$

To do so, we proceed by iteratively picking a label $\alpha \in \mathcal{L}$, and optimize simultaneously the dual variables belonging to this label and the primal variables. This scheme shares a lot of similarity with the Alpha-Expansion algorithm previously presented. Hence, we refer to this step as expansion.

Expansion

It turns out that by properly setting the capacities of a maxflow problem, one can jointly optimize the primal and dual variables for a given label l. The curious reader can find proof of correctness in the original publication [102].

Pre-editing dual variables: We need to ensure that after the maxflow the following constraints are verified whether x_i or x_j are updated or not:

$$\begin{aligned}& y_{ij,x_i^*} + y_{ji,x_j^*} = \theta_{ij,x_i^*,x_j^*}, \quad \forall (i,j) \in \mathcal{E} \\ & y_{ij,a} + y_{ji,b} \leqslant 2 \max_{(c,d) \in \mathcal{L} \times \mathcal{L}} \theta_{ij,c,d}, \quad \forall (ij,a,b) \in \mathcal{E} \times x_i, x_j, l \times x_i, x_j, l\end{aligned}$$

To this end, and before maxflow, we always force the dual variables $(y_{ij,l})_{ij}$ to verify:

$$-\theta_{ij,x_j,l} + y_{ij,x_i} \leqslant y_{ij,l} \leqslant \theta_{ij,l,x_j} + y_{ji,x_j}$$

Setting-up the maxflow problem: We set the capacities between nodes $(i,j) \in \mathcal{E}$ to:

$$\begin{aligned}cap_{ij} &= [\theta_{ij,l,x_j} - (y_{ij,l} + y_{ji,x_j})]_{\mathbb{R}^+} \\ cap_{ji} &= [\theta_{ij,x_j,l} - (y_{ij,x_i} + y_{ji,l})]_{\mathbb{R}^+}\end{aligned}$$

To simplify notations we introduce the height variable for each vertex $i \in \mathcal{V}$ and each label $l \in \mathcal{L}$:

$$h_{i,l} = \theta_{i,l} + \sum_{j \in \mathcal{E}(i)} y_{ij,l}$$

The capacities from the source to each vertex $i \in \mathcal{V}$ are given by:

$$cap_{si} = [h_{i,x_i} - h_{i,l}]_{\mathbb{R}^+}$$

while the capacities from each vertex $i \in \mathcal{V}$ to the sink are given by:

$$cap_{it} = [h_{i,x_i} - h_{i,l}]_{\mathbb{R}^+}$$

Post-updating primal and dual variables: Once the maxflow has been computed the outer flow variables $(f_{si})_i$ and the inner flow variables $(f_{ij})_{ij}$ indicate the update for the primal and dual variables.

The primal variables are updated to label l only if there is an unsaturated path from the source to the node:

$$x_i \leftarrow l \quad \text{if} \;\; cap_{si} > 0 \;\; \& \;\; cap_{si} - f_{si} > 0, \quad \forall (i) \in \mathcal{V}.$$

We update the dual variables using the inner flow $(f_{ij})_{ij}$ as follow:

$$\begin{aligned} y_{ij,l} &\leftarrow y_{ij,l} + cap_{ij} - f_{ij}, && \forall (i,j) \in \mathcal{E}. \\ y_{ij,l} &\leftarrow -y_{ij,l}, && \forall (i,j) \in \mathcal{E}. \end{aligned}$$

The Fast PD algorithm

Finally, we put everything together to get the method Fast-PD that we summarize in algorithm 15.

4.5.3 A note on Fast-PD implementation

We decided to implement Fast-PD from scratch in C++ for two main reasons: (1) the current implementation [96] is using way too much memory to run on our remote sensing tasks, and (2) both the architecture and readability need to be improved to allow others to easily tailor to their needs a custom version of Fast-PD.

We have to make several design choices when implementing Fast-PD. We want to make the implementation as fast as possible while maintaining a reasonable memory footprint since we are going to process large problems. Hence, we are making different choices than the implementation provided by the original authors.

Conjugate dual variables

The algorithm always enforces and maintains conjugacy between $y_{ij,l}$ and $y_{ji,l}$:

$$y_{ij,l} = -y_{ji,l}, \quad \forall (i,j) \in \mathcal{E} \text{ and } \; l \in \mathcal{L}$$

Hence, in the implementation of Fast-PD we only maintain in memory half of the dual variables: $(y_{ij,l})_{(ij,l)}$. When we need the other half we simply return minus its conjugate. This reduces the memory footprint of the dual variables by half at the price of a minor computation. The original implementation also made use of this.

Algorithm 15: Fast-PD

Data: Inputs: $(\phi_i)_i$, $(\phi_{ij})_{ij}$, $\mathcal{G}$, $(x_i)_i$
Result: $(x_i)_i$
Initialize dual variables to verify:

$$y_{ij,x_i} + y_{ji,x_j} = \phi_{ij}(x_i, x_j), \quad \forall (i,j) \in \mathcal{E}$$

while *Until no expansion improves the solution* **do**
  for $l \in \mathcal{L}$ **do**
    Pre-edit dual variables such that:

$$-\theta_{ij,x_j,l} + y_{ij,x_i} \leqslant y_{ij,l} \leqslant \theta_{ij,l,x_j} + y_{ji,x_j} \quad \forall (i,j) \in \mathcal{E}$$

    Setup and solve maxflow problem.
    Post-updating primal variables:

$$x_i \leftarrow l \quad \text{if} \quad cap_{si} > 0 \quad \& \quad cap_{si} - f_{si} > 0, \quad \forall (i) \in \mathcal{V}.$$

    Post-updating dual variables:

$$\begin{aligned} y_{ij,l} &\leftarrow y_{ij,l} + cap_{ij} - f_{ij}, && \forall (i,j) \in \mathcal{E}. \\ y_{ij,l} &\leftarrow -y_{ij,l}, && \forall (i,j) \in \mathcal{E}. \end{aligned}$$

Maintaining the height variables

During the expansion sub-routine of Fast PD, we make use of the height variables:

$$\begin{aligned} h_{i,l} &= \theta_{i,l} + \sum_{j \in \mathcal{E}(i)} y_{ij,l} \\ &= \phi_i(l) + \sum_{j \in \mathcal{E}(i,.)} y_{ij,l} - \sum_{j \in \mathcal{E}(.,i)} y_{ij,l} \end{aligned}$$

Since, we need to have access to the $(\theta_i)_i$ we are going to store in memory all the $(h_{i,l})_{i,l}$. However, contrary to the original implementation that uses the space allocated to the $(\theta_i)_i$ to store the $(h_{i,l})_{i,l}$, we are going to allocate the $(h_{i,l})_{i,l}$ in their own space. This is much better in terms of software architecture. If the $(\theta_i)_i$ are not needed by another process, the class that owns them can deallocate their space. In case they are needed after Fast-PD, we provide a method to retrieve the $(\theta_i)_i$ from the $(h_{i,l})_{i,l}$. This way, we keep a clean architecture, maintain the same memory footprint, and only add the cost of copying the $(\theta_i)_i$ (which is negligible).

Updating the source and sink trees for each maxflow

As previously explained the maxflow algorithm finds an augmenting path by simultaneously growing two trees, one from the source and one from the sink. This is efficient if we expect the amounts of nodes connected to the source and the sink to be balanced. However, as Fast-PD progresses, fewer nodes are going to be connected to the source. Hence, the original implementation makes use of a modified maxflow that only grows the source tree to find augmenting paths. This potentially avoids to grow a very large sink tree which could slow down the maxflow computation.

Unfortunately, to the best of our knowledge, no experiment quantifies the improvement gained by only growing the source tree. Hence, we investigate this during our experiments.

Maintaining the trees and capacities of each maxflow

As Fast-PD progresses fewer modifications are made to primal and dual variables. Hence, the original implementation proposed to keep in memory the capacities of each maxflow problem along with the tree structures for finding the augmenting path. This allows to only recompute the capacities that might have changed during a full cycle of expansion. Furthermore, the trees can also be used to warm start the computation of the maxflow since only a small part is likely to be obsolete.

However, the down side is that a large amount of extra memory is required for storing the capacities and the trees. Hence, we elect to recompute from scratch the capacities for each expansion. We maintain in memory only one maxflow that we reinitialize before each expansion. This allows us to avoid allocating and de-allocating at each iteration.

Cache friendly indexation

During each expansion phase of Fast-PD, the algorithm goes through all edges to pre-edit dual variables, computes the capacities, and then post-updates the dual variables. Hence, we need to pay attention that the pattern used to access the variables maximizes cache friendliness. Hence, we perform experiments to investigate and quantify the benefit of different memory layouts.

4.6 Experiments

In our context we mainly deal with neighborhood structures defined on a grid. Moreover, due to the nature of our applications, our problem tends to be quite large. For instance, a satellite image is generally of the order of ten thousand pixel per dimension. Hence, we limit our experiments to 4-connected grid.

We make use of the stereo matching task to illustrate the differences between maxflow solvers on the one hand, and between the Alpha Expansion and the different implementations of Fast PD on the other hand.

4.6.1 The stereo-Matching Problem

In the stereo-matching task [111] we are given a stereo-pair in epipolar geometry composed of a reference image I_r and a target image I_t, and a finite set of potential disparities $\mathcal{D} = [d_0, d_1, ...d_D]$, i.e., a one-dimensional apparent displacement along the epipolar line. The goal is to retrieve the most likely disparity map, i.e., the apparent motion of the pixel from the reference image to the target image. For each stereo-pair, the support graph $\mathcal{G} = [\mathcal{V}, \mathcal{E}]$ is inherited from the 4-connectivity of the reference image. The node i directly corresponds to pixels $p_i = (r_i, c_i)$ of row r_i and column c_i. The label set $\mathcal{L}$ simply indexes the set of potential disparities $\mathcal{D}$.

Defining the potentials: For the unary potential we rely on the *ZNCC*, Zero Normalized Cross Correlation, coefficient [113]. For each label $l \in \mathcal{L}$ and each node $i \in \mathcal{V}$, we compute:

$$\phi_i(l) = 1 - ZNCC_W(I_r, I_t, i, \mathcal{D}(l))$$

where $ZNCC$ computes the ZNCC coefficient between the square patch of width W extracted from image I_r around pixel (r_i, c_i) and the patch of width W extracted from image I_t around pixel $[r_i, c_i + \mathcal{D}(l)]$. We set $W = 5$ to get patches of size 5×5.

For the pairwise potential, we used the popular weighted l1-distance to favor a piecewise constant disparity maps. For each edge $(i, j) \in \mathcal{E}$ and each pair of labels $(a, b) \in \mathcal{L} \times \mathcal{L}$ we define:

$$\phi_{ij}(a, b) = (w_0 + w_1 \exp(-w_2 ||I_r(i) - I_r(j)||_2^2)) \times |\mathcal{D}(a) - \mathcal{D}(b)|$$

where w_0, w_1 and w_3 are positive real scalars.

Remote sensing stereo-pairs: We use a large stereo pair acquired with the *Ultracam* camera during an aerial survey above an urban environment to extract 4 subsets of size 1500×1500. The figure 4.3 displays the reference image of each subset. The stereo pair has been beforehand calibrated and globally registered. Hence, the remaining registration only consists of an horizontal offset ranging from -50 to 50 pixels. We remind that the goal of this experiments is to study the optimization algorithms with realistic problems and not to get the best disparity maps. The figure 4.4 displays the obtained disparity maps.

We create 3 different sizes of stereo matching problems by sampling if needed the stereo pair subsets:

- Small: the graph is 500×500 nodes and 21 labels are to be optimized.

Factory subset

Church subset

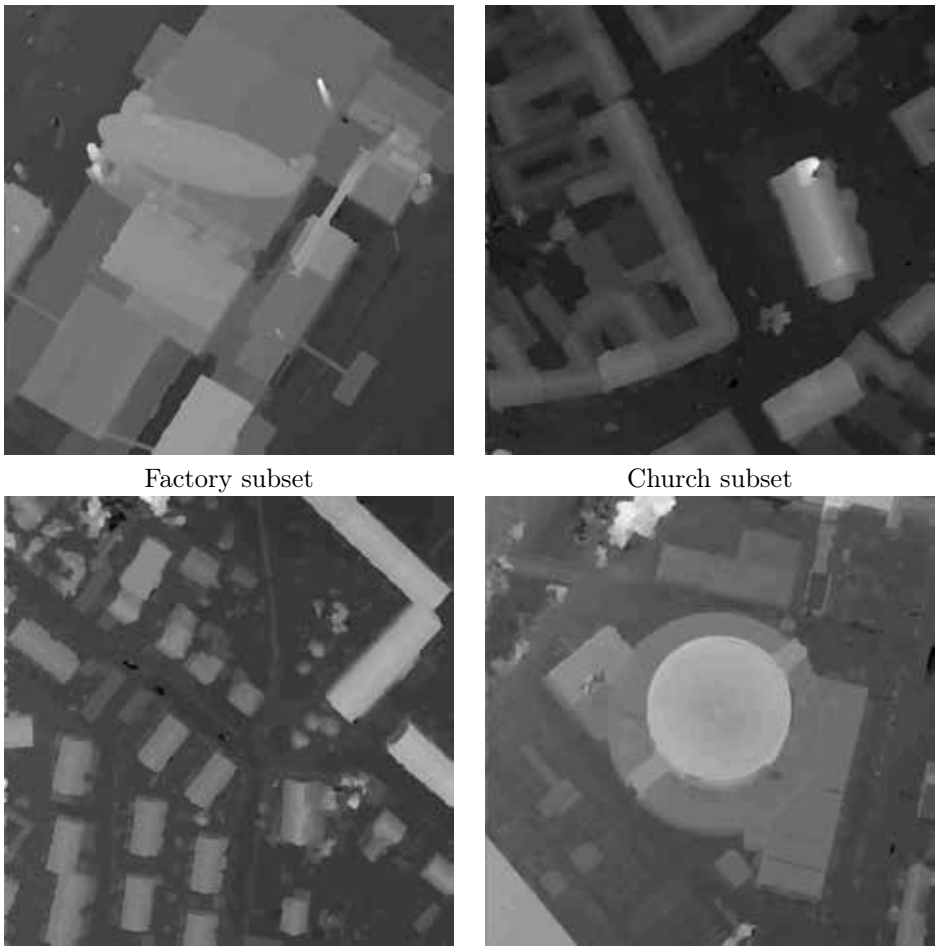
Buildings subset

Industry subset

Figure 4.3 – Reference images of stereo pair subsets for the stereo-matching experiments.

- Medium: the graph is 1000×1000 nodes and 51 labels are to be optimized.
- Large: the graph is 1500×1500 nodes and 101 labels are to be optimized.

Implementations: We consider only CPU C++ implementations compiled with Clang (Apple LLVM version 8.0.0) on the same device: Mac-book Pro mid 2015 with 2.2 GHz Intel Core i7, 16GB of RAM, and a 64bits operating system. All programs are compiled in 64bits.



Factory subset | Church subset

Buildings subset | Industry subset

Figure 4.4 – Disparity maps: Brighter grays represent higher elevation.

4.6.2 Maxflow experiments for 4 connected graph

We start by investigating the maxflow algorithm since it is a building block for the expansion phase of Fast PD.

Solvers

We evaluate 4 different maxflow-mincut algorithms: BK maxflow, GridCut (GC), TV-linear and the TV-L2 (ROF).

Augmenting path solvers: The first two belong to the class of Augmenting Path techniques. The famous BK maxflow [94], a standard in the computer

vision, makes use of a heuristic to find an augmenting path that is well suited for image derived problems. The original implementation assumes any connectivity and therefore, it requires extensive additional data structure.

The GridCut maxflow based on [88] and [114] is a re-implementation of BK maxflow that supports only grid graph. This additional assumption coupled with an efficient data structure favoring cache locality allows to reduce the quantity of additional data structure. We use the code provided by the authors [87].

Iterative solvers: The last two algorithms are iterative solvers based on primal dual optimization techniques. They are suited for both CPU and GPU implementation and they require minimal additional data structure. We do not expect these two algorithms to compete with the augmenting path technique. Indeed, the augmenting path technique is extremely well tuned for serial computation (single thread CPU implementation). We implement our own version of the TV-linear and the TV-L2 algorithms that favors cache friendliness. We stop the algorithm when the average update of the primal variables is less than 0.1% or if the number of iteration exceed 500.

Experiments settings

We create maxflow problems that correspond to the expansion phase of the Alpha expansion. We consider the expansion of each label against the labeling that minimizes the cost of unary terms. This helps us to create realistic problems to compare our different maxflow algorithms.

We monitor for each algorithm the runtime. To estimate the accuracy of the TV-linear and TV-L2 algorithm we compute the normalized PD-gap:

$$1000 \ \frac{\text{primal} - \text{dual}}{\text{primal}^*} \tag{4.85}$$

Results

We present the results of all experiments in tables 4.1, 4.2 and 4.3.

In all our experiments the grid-cut maxflow performs more than twice as fast as the BK-maxflow. This is consistent with the results announced by the grid-cut authors for stereo-matching based problems. Both iterative solvers TV-linear and TV-L2 perform significantly worse despite only returning an approximate solution. This was to be somehow expected since the iterative solver does not exploit the structure of dual solution space. The TV-L2 algorithm largely outperforms its TV-linear counterpart which is conform to the theoretical convergence rates of both methods. We note that the TV-L2 is only 4 times slower than the baseline. Hence, it remains a great candidate for GPU implementation.

Given the results, we will only keep the augmenting path based maxflow solver when implementing Fast-PD. While the grid-cut maxflow is faster, it lacks the generality of the BK maxflow.

	Small size problems				
	Church	Factory	Buildings	Industry	Average
BK (baseline)					
Runtime (ms)	45.16	44.68	45.40	43.38	44.66
Speed-up vs baseline	1	1	1	1	1
GC					
Runtime (ms)	20.37	20.34	19.97	18.90	19.90
Speed-up vs baseline	2.22	2.20	2.27	2.29	2.24
TV-linear					
Runtime (ms)	271.42	248.34	242.41	259.11	255.32
Speed-up vs baseline	0.17	0.18	0.19	0.17	0.17
Normalized PD-gap	4.79	2.10	5.80	3.76	4.11
TV-L2					
Runtime (ms)	175.60	177.34	173.82	169.81	174.14
Speed-up vs baseline	0.26	0.25	0.26	0.26	0.26
Normalized PD-gap	0.91	1.21	1.10	0.99	1.05

Table 4.1 – Comparing maxflow algorithm on small scale problems.

	Medium size problems				
	Church	Factory	Buildings	Industry	Average
BK (baseline)					
Runtime (ms)	187.31	188.95	187.03	190.22	188.38
Speed-up vs baseline	1	1	1	1	1
GC					
Runtime (ms)	187.31	188.95	187.03	190.22	188.38
Speed-up vs baseline	2.26	2.29	2.32	2.14	2.25
TV-linear					
Runtime (ms)	2430.40	2221.84	2270.64	2413.98	2334.21
Speed-up vs baseline	0.08	0.09	0.08	0.08	0.08
Normalized PD-gap	5.41	4.90	5.95	4.22	5.12
TV-L2					
Runtime (ms)	667.76	680.63	659.60	658.90	666.73
Speed-up vs baseline	0.28	0.28	0.28	0.29	0.28
Normalized PD-gap	0.74	0.78	0.78	0.81	0.78

Table 4.2 – Comparing maxflow algorithm on medium scale problems.

4.6.3 Fast PD implementation experiments

We propose to compare three different implementations of Fast-PD. As a baseline, we use the C++ implementation provided by the original authors based on a modified BK maxflow, we refer to it as *Fast PD (NK)*. We propose two alternative implementations: one also based on the BK maxflow, named *Fast PD (BK)*, and one based on the Grid-Cut maxflow, called *Fast PD (GC)*. We note that

	Large size problems				
	Church	Factory	Buildings	Industry	Average
BK (baseline)					
Runtime (ms)	462.92	454.19	451.51	490.31	464.74
Speed-up vs baseline	1	1	1	1	1
GC					
Runtime (ms)	204.19	199.41	199.14	227.83	207.64
Speed-up vs baseline	2.27	2.28	2.27	2.15	2.24
TV-linear					
Runtime (ms)	10282.85	9234.86	9718.06	10679.88	9978.91
Speed-up vs baseline	0.05	0.05	0.05	0.05	0.05
Normalized PD-gap	5.82	6.52	6.69	4.08	5.78
TV-L2					
Runtime (ms)	1494.28	1506.16	1486.56	1501.95	1497.24
Speed-up vs baseline	0.31	0.30	0.30	0.33	0.31
Normalized PD-gap	0.76	0.76	0.79	0.80	0.78

Table 4.3 – Comparing maxflow algorithm on large scale problems.

an implementation using the BK maxflow can handle any graph architecture while the implementation based on the Grid-Cut maxflow can only handle 4-connectivity grid. However, we note that it is relatively easy to extend the Grid-Cut version to 8-connectivity for instance.

Implementation memory footprint limitations

Since we work with large images in our context, we need to pay a close attention to the memory footprint of our algorithms. Hence, our first experiment quantifies the memory footprint of the different implementations. The table 4.4 gives the memory required by each algorithm.

The memory footprint required by the original implementation of Fast-PD makes it unsuitable in our context. Our implementations only needs on average less than 10% of the original implementation memory requirement. The Fast PD (GC) requires less memory than the Fast PD (BK) since the grid-cut maxflow can take advantage of the implicit grid structure to avoid storing the neighbors connectivity in memory. While our implementations still require a significant amount of memory, they can easily run even on modern laptops.

Cache friendly indexation

We now proceed to exhibit the importance of cache friendliness when implementing algorithm such as Fast-PD. To this end, we propose two experiments.

Fast PD (BK) experiment: The first one makes use of Fast PD (BK). We compare the runtime when the edges are randomly ordered and when they are

	Fast PD (NK) (baseline)	Fast PD (BK)	Fast PD (GC)
Small size problems:			
Memory footprint	981 MB	115 MB	92 MB
Ratio vs baseline	1	0.1172	0.0938
Medium size problems:			
Memory footprint	9066 MB	774 MB	648 MB
Ratio vs baseline	1	0.0854	0.0715
Large size problems:			
Memory footprint	40.02 GB	2.91 GB	2.61 GB
Ratio vs baseline	1	0.0727	0.0652

Table 4.4 – Memory footprint for different Fast PD implementations.

ordered to maximize cache friendliness. For both cases the memory layout of all edge related variables (weights and dual variables) matches the order of the edges. All node related variables (the primal and height variables) are stored as a continuous vector that follows the vertical lines of the graph.

The main cases for cache misses happen during the pre-edit phase of the expansion. In this phase the algorithm computes the inner capacities of the maxflow problem by scanning the dual variables following the order of the edges. It also needs to fetch the two primal variables associated to the edge, and it might also update the two height variables associated to the edge. If from one edge to the next, those two primal and two height variables are far away in memory (not in the same cache line), then a cache miss occurs, forcing to load a new cache line. Sometime this loading time can be hidden by some others computations. However, we illustrate that ordering the edges such that they follow the node memory layout favors cache hits, and as a result boosts performance. The results are presented in table 4.5.

Hence, just by properly ordering the memory layout we obtain up to 1.7 speed-up. Hence, from now on we only use the proper memory layout for our experiments.

Fast PD (GC) experiment: For our second experiment we propose two memory layouts for the node and edge related variables. The first layout is exactly the one we used for the Fast PD (BK) with proper ordering. However, the grid-cut maxflow algorithm uses a particular memory layout to favor cache locality during the growing and augmenting phases. Hence, we derive a second version of Fast PD (GC) that matches the grid-cut memory layout. The results are reported in the table 4.6.

As for the Fast PD (BK) experiment, the proper memory layout leads to improved performances up to 1.3. We note that for the pre-edit dual phase of

	Random ordering (baseline)	Proper ordering
Small size problems:		
Average runtime (sec)	1.7	1.1
Average speed-up vs baseline	1	1.52
Medium size problems:		
Average runtime (sec)	49.0	28.2
Average speed-up vs baseline	1	1.73
Large size problems:		
Average runtime (sec)	330.2	189.6
Average speed-up vs baseline	1	1.74

Table 4.5 – Cache friendliness experiments for Fast PD (BK)

	BK layout (baseline)	Grid-cut layout
Small size problems:		
Average execution time (sec)	1.2	1.0
Average speed-up vs baseline	1	1.24
Medium size problems:		
Average execution time (sec)	24.2	18.2
Average speed-up vs baseline	1	1.33
Large size problems:		
Average execution time (sec)	144.6	116.4
Average speed-up vs baseline	1	1.24

Table 4.6 – Cache friendliness experiments for Fast PD (GC)

Fast-PD the grid-cut memory layout is a bit less efficient than the BK layout. However, this is largely compensated by the gain obtained when filling the maxflow capacities. Therefore, for Fast PD (GC) we elect to use the same memory layout than grid-cut.

Growing source and sink trees

We now verify the claim of the original authors of Fast PD about growing only the source tree. To this end, we perform a simple experiment. We solve each of our stereo matching problem twice, once growing only the source tree and once growing both source and sink trees. We carry-out this experiment for both Fast PD (BK) and Fast PD (GC). The results are reported in table 4.7 and 4.8.

For Fast PD (BK) the results illustrates that as the problems grow larger

	Source and sink (baseline)	Source only
Small size problems:		
Average runtime (sec)	1.2	1.1
Average speed-up vs baseline	1	1.04
Medium size problems:		
Average runtime (sec)	32.2	28.5
Average speed-up vs baseline	1	1.13
Large size problems:		
Average runtime (sec)	267.5	191.6
Average speed-up vs baseline	1	1.40

Table 4.7 – Growing tree experiments for Fast PD (BK)

	Source and sink (baseline)	Source only
Small size problems:		
Average runtime (sec)	1.0	0.9
Average speed-up vs baseline	1	1.05
Medium size problems:		
Average runtime (sec)	21.2	18.5
Average speed-up vs baseline	1	1.15
Large size problems:		
Average runtime (sec)	136.4	117.9
Average speed-up vs baseline	1	1.16

Table 4.8 – Growing tree experiments for Fast PD (GC)

the speed-up increases up to 1.4. For Fast PD (GC) the results follow the same trend but the speed up stalls around 1.15. Both experiments support the claim of the original authors of Fast-PD. Hence, we elect to only grow the source tree.

Comparing Fast PD implementations

We now compare the run time between the different implementations of Fast PD. For Fast PD (NK) we did not run the experiments for large problems due to the memory footprint. For the medium size problem, it is possible that the OS decided to compress the memory allocated to Fast PD (NK). This would have an impact on the runtime but we decided not to discard this experiment since it is a normal behavior and we can expect our users to run in the same situation. We remind that compressing the memory penalizes the runtime far

less than swapping to hardrive even for SSD. We report the results in table 4.9.

	Small size problems				
	Church	Factory	Buildings	Industry	Average
Fast PD (NK) (baseline)					
Runtime (sec)	1.65	1.41	1.24	1.40	1.42
Speed-up vs baseline	1	1	1	1	1
Fast PD (BK)					
Runtime (sec)	1.48	1.10	1.08	1.29	1.24
Speed-up vs baseline	1.11	1.28	1.14	1.08	1.15
Fast PD (GC)					
Runtime (sec)	1.26	0.89	0.96	1.04	1.04
Speed-up vs baseline	1.30	1.57	1.29	1.35	1.37

	Medium size problems				
	Church	Factory	Buildings	Industry	Average
Fast PD (NK) (baseline)					
Runtime (sec)	28.35	25.96	18.73	23.24	24.07
Speed-up vs baseline	1	1	1	1	1
Fast PD (BK)					
Runtime (sec)	24.01	35.11	23.09	28.69	27.73
Speed-up vs baseline	1.18	0.74	0.81	0.81	0.87
Fast PD (GC)					
Runtime (sec)	15.80	22.05	15.86	18.37	18.02
Speed-up vs baseline	1.79	1.18	1.18	1.26	1.34

	Large size problems				
	Church	Factory	Buildings	Industry	Average
Fast PD (NK)					
Runtime (sec)	-	-	-	-	-
Fast PD (BK)					
Runtime (sec)	171.37	212.90	139.71	218.52	185.63
Fast PD (GC)					
Runtime (sec)	108.88	118.24	96.60	137.57	115.32
Speed-up vs Fast PD (BK)	1.57	1.80	1.45	1.59	1.61

Table 4.9 – Comparison of Fast PD different implementations

For all experiments, Fast PD (GC) outperforms the other two implementations. This is an ideal situation since it is also the implementation that requires the less memory. Hence, for 4-connected problems and in our context, Fast PD (GC) should always be the algorithm of choice.

It was a bit surprising to us that for small size problems Fast PD (NK), the original implementation, is the slowest version. In fact, the extra allocation time needed by Fast PD (NK) cannot be compensated by the speed-up obtained in

the last expansion phases. For medium size problem, we witness no evidence that the OS decided to compress memory. In these settings, Fast PD (NK) outperforms Fast PD (BK) by 10%. Given the amount of memory saved by Fast PD (BK), this is good trade-off since we reduce the memory consumption by 90% to only increase the computation by 10%. For large problems, Fast PD (GC) is on average 1.6 time faster than Fast PD (BK). This is mainly thanks to a faster maxflow algorithm. However, it is important to keep in mind that Fast PD (BK) can handle any graph connectivity.

To conclude, our two implementations are suitable for large size problems. In term of running time both are very competitive since they outperform the original problem except for the medium size problems for Fast PD (BK). For 4-connected problems, we elect to use Fast PD (GC). For any other graph connectivity, we use Fast PD (BK).

4.6.4 Fast PD vs Alpha-expansion

We now need to compare our implementation against the Alpha-expansion algorithm. To this end, for the Alpha-expansion we use the C++ implementation provided by [165] that we compile with the same optimization flags as our implementations. As stated in [102], both Alpha-expansion and Fast-PD obtain similar results in term of energy and solution. Hence, we only provide the results of our experiments on running time in table 4.10.

Both Fast PD (BK) and Fast PD (GC) outperform Alpha Expansion significantly. This is conform with the experiments of [102]. Fast PD (BK) is on average twice as fast as the Alpha Expansion and Fast PD (GC) runs between 3 and 4 times faster than Alpha Expansion. Moreover, Fast-PD is also capable of solving a larger class of problems than the Alpha Expansion. Hence, since the memory footprint of both Fast PD (BK) and Fast PD (GC) is suitable to our context, we elect to use Fast PD (BK) and Fast PD (GC) over Alpha Expansion.

	Small size problems				
	Church	Factory	Buildings	Industry	Average
Alpha Expansion (baseline)					
Runtime (sec)	3.49	2.55	3.22	3.39	3.16
Speed-up vs baseline	1	1	1	1	1
Fast PD (BK)					
Runtime (sec)	1.56	1.03	1.11	1.30	1.25
Speed-up vs baseline	2.24	2.47	2.90	2.60	2.53
Fast PD (GC)					
Runtime (sec)	1.28	0.90	0.96	1.14	1.07
Speed-up vs baseline	2.73	2.84	3.36	2.98	2.96

	Medium size problems				
	Church	Factory	Buildings	Industry	Average
Alpha Expansion (baseline)					
Runtime (sec)	77.74	86.83	71.37	69.14	76.27
Speed-up vs baseline	1	1	1	1	1
Fast PD (BK)					
Runtime (sec)	23.84	35.11	23.71	32.75	28.85
Speed-up vs baseline	3.26	2.47	3.01	2.11	2.64
Fast PD (GC)					
Runtime (sec)	16.13	22.10	16.49	19.57	18.57
Speed-up vs baseline	4.82	3.93	4.33	3.53	4.11

	Large size problems				
	Church	Factory	Buildings	Industry	Average
Alpha Expansion (baseline)					
Runtime (sec)	409.76	399.50	354.75	557.20	430.30
Speed-up vs baseline	1	1	1	1	1
Fast PD (BK)					
Runtime (sec)	171.78	213.84	140.24	227.14	188.25
Speed-up vs baseline	2.39	1.87	2.53	2.45	2.29
Fast PD (GC)					
Runtime (sec)	108.47	118.44	97.02	143.02	116.74
Speed-up vs baseline	3.78	3.37	3.66	3.90	3.69

Table 4.10 – Comparing Fast PD and graph cuts.

4.7 Conclusion

This second technical chapter introduced the basic of non-convex optimization. We presented and demonstrated the link between the maxflow and mincut problems. We surveyed different algorithm to solve either a maxflow or mincut problem. Then, we presented the α expansion algorithm along with Fast-PD. We extensively experiment with our own implementation of Fast-PD in the context

of stereo-matching where it drastically outperforms the original implementation.

Despite these improvement non convex optimization techniques remains computationally intensive. Hence, we investigate in the next chapter coarsening scheme for non convex optimization. We also provide a comparison in the context of remote sensing between the non convex optimization techniques of this chapter and the First order Primal-Dual methods of the previous chapter.

Chapter 5

Coarsening schemes for optimization techniques

5.1 Introduction and contributions

5.1.1 Introduction

In this chapter we consider coarsening scheme for the optimization techniques presented in the two previous chapters 3 and 4.

Hence, we study coarsening schemes suited for the First order Primal-Dual techniques for convex optimization. In this context, we review how to transform our initial non convex problem to a series of convex surrogate problems. We use the stereo-matching task of last chapter 4 to evaluate the performances and trade-offs of different schemes.

We also investigate coarsening schemes for Graph-Cuts techniques. In this context, the goal is to reduce the overall computational complexity. To this end, we present a new coarsening scheme that uses machine learning techniques to deliver impressive trade-off between speeding-up and precise optimization. As for the First order Primal-Dual techniques, we perform experiments with the stereo-matching task.

5.1.2 Chapter organization

The section 5.2 introduces and compares the smoothing and coarsening scheme for first order primal-dual techniques. We discuss in section 5.3 the image and energy pyramid schemes for graph-cuts optimization techniques. We also introduce a powerful optimization framework named *Inference by learning*.

5.1.3 Contributions

The main contribution of this chapters related to the coarsening scheme for Graph-Cuts techniques where we present different approaches to coarsen an MRF model. None only our coarsening scheme speeds up the optimization but it also delivers more accurate solution. Finally, we present a novel framework built on top of the coarsening scheme that drastically speeds-up the inference while maintaining impressive accuracy.

5.2 Smoothing and Coarsening scheme for first order primal dual optimization techniques

The first order primal dual scheme assumes that the function to optimize is convex. However, we can not guaranty the convexity of the functions of interest. Hence, we detail a scheme to approximate any given function with a convex surrogate function.

To this end, we first survey the causes of non convexity. Then, we show that we can easily create a convex surrogate function that locally approximates the given function. Finally, we analyze two schemes, smoothing and coarsening, to extend the approximation to larger support.

5.2.1 Preliminary work

Approximation measure

We propose to make use of the approximation measure to evaluate the quality of the approximation of the surrogate function. We use the following definition to measure how close a function f approximates a reference function f_{ref}:

$$100 \ \frac{\int |f(x) - f_{ref}(x)| dx}{\int |f_{ref}(x)| dx} \tag{5.1}$$

We note that many other approximation measures exist.

Origin of the non convexity

The non convexity can arise in both the matching and the regularization terms of the given function.

Non convex matching term The matching term is the main culprit of non-convexity for registration problems. The non convexity comes from the spatial ressampling and is amplified by the matching criterion. We also note that the matching criteria such as census, ZNCC or truncated norms are also non convex functions.

Image ressampling The spatial ressampling of an image I at point $p_i = (r_i, c_i) \in \Omega$, the spatial support of image, of row r_i and column c_i with a displacement vector $d_i = (d_{r_i}, d_{c_i})$ defines the following function:

$$f_{I,p_i}(d_i) = I(r_i + d_{r_i}, c_i + d_{c_i})$$

The spatial ressampling of any natural digital image is almost statistically guaranteed to yield a non convex function. We exemplify this phenomenon by ressampling along the horizontal axis images of aerial survey acquired with the *UltraCam* camera. We use the same images than in the stereo matching application of the previous chapter. The figure 5.1 illustrates typical ressampling functions $(f_{I,p_i})_i$ at four different points with the following ressampling techniques: nearest, linear and cubic. We point the interested reader to [2] for a detailed definition of these ressampling techniques.

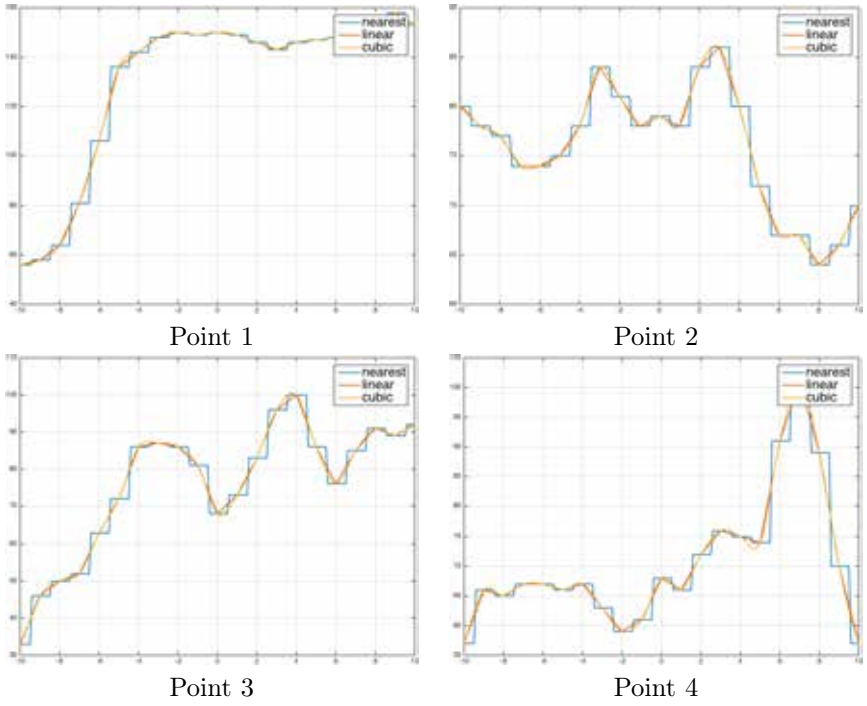


Figure 5.1 – Non convexity and image ressampling.

The curves highlights the non convexity of the ressampling function independently of the method. The non convexity is directly linked to image content. Hence, it generally drastically varies within the same image. The main difference between ressampling techniques is the smoothness. The cubic method always generates smooth curves while the nearest and linear methods generate non smooth curves with the nearest methods being the least smooth. We also note

that the algorithmic complexity increases between methods, with the nearest method being the fastest and the cubic method being the slowest. Yet, in our context this only accounts for a negligible part of the total computational complexity.

Matching criterion We survey how much the matching criterion amplifies the non convexity. To this end, in the context of stereo matching introduced in the last chapter we make use of the ℓ_1 and $ZNCC$ matching criteria. We display with figures 5.2 and 5.3 for the same points that figure 5.1 the curves associated to each matching criterion with the different ressampling methods.

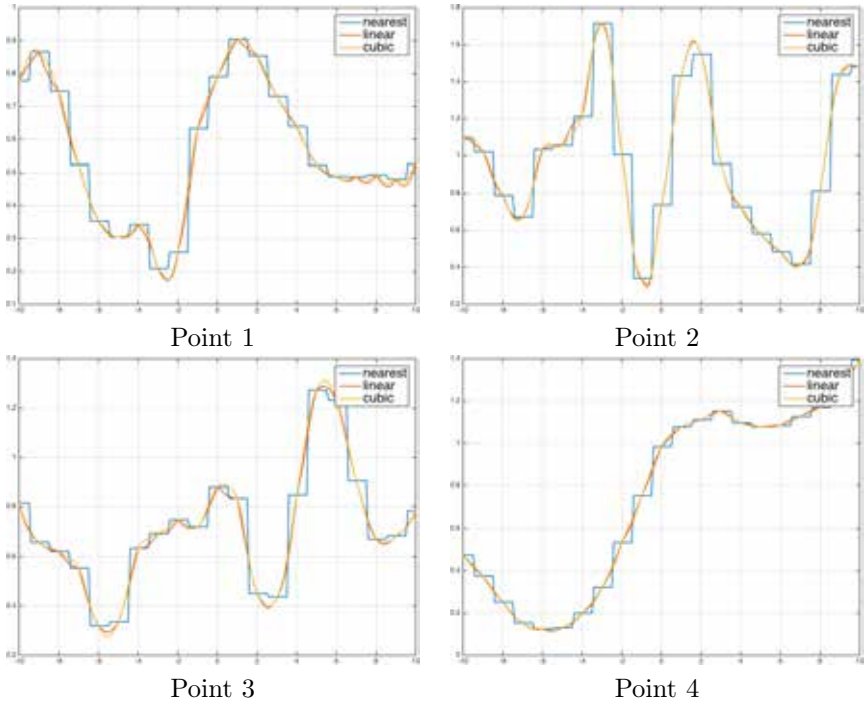


Figure 5.2 – Non convexity and $ZNCC$ matching criterion.

We observe that the matching criterion amplifies the non convexity. Moreover, the $ZNCC$ being a non-convex function accentuates the non-convexity more than the ℓ_1 matching criterion. For both criteria, we observe the function maintains the smoothness property of the ressampling method. The matching criterion also increases the difference between ressampling methods. We quantify this phenomenon by taking the cubic ressampling as reference and computing the approximation measure for the nearest and linear ressampling methods.

The table 5.1 details the approximation measure for all 4 subsets of the stereo matching application of the previous chapter. The figure 5.4 displays the

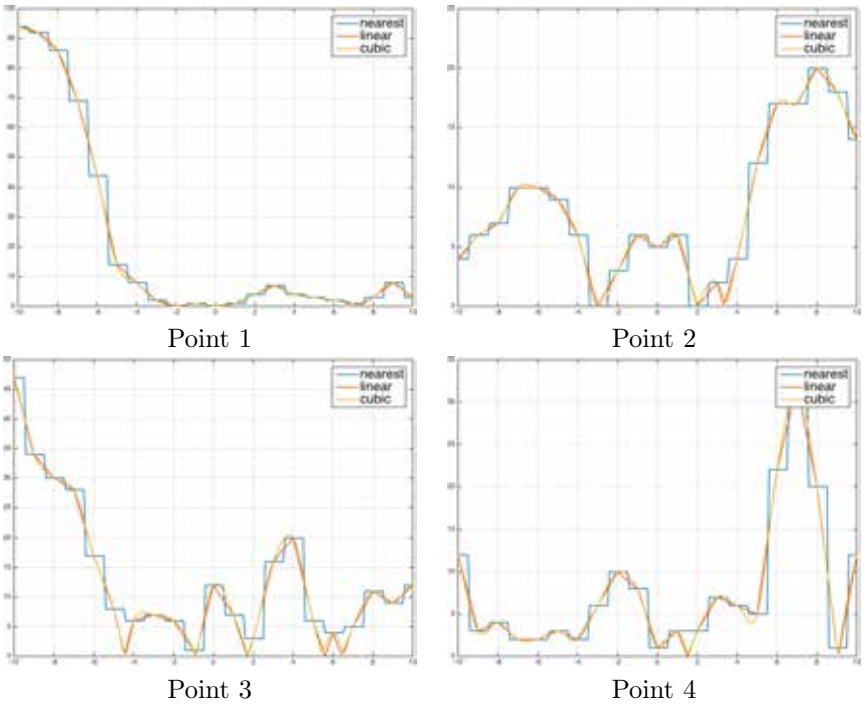


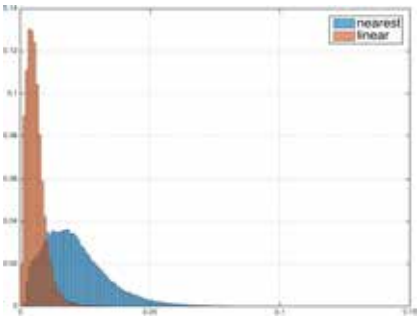
Figure 5.3 – Non convexity and ℓ_1 matching criterion.

histogram of the approximate measure for all 4 subsets.

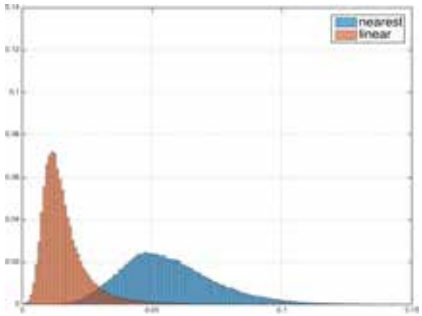
	Church	Factory	Buildings	Industry	Average
Ressampling					
Nearest	2.27	1.60	2.48	2.02	2.09
Linear	0.62	0.42	0.72	0.55	0.58
$ZNCC$					
Nearest	5.58	6.23	5.35	5.42	5.64
Linear	1.72	1.60	1.63	1.46	1.60
ℓ_1					
Nearest	8.57	9.48	9.82	9.40	9.31
Linear	2.51	2.67	3.01	2.74	2.73

Table 5.1 – Approximation measure for the ressampling and computation of unary terms using the ZNCC matching criterion.

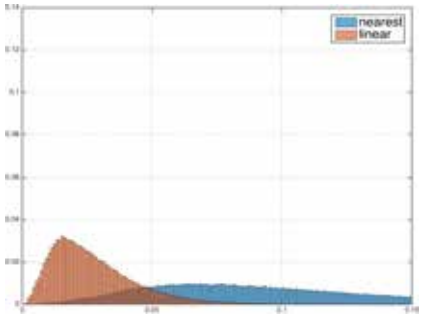
The table 5.1 confirms that the matching criteria are amplifying the differences between the ressampling methods. The difference between nearest and linear ressampling is as expected significant. Surprisingly, the ℓ_1 matching criterion is twice as more sensitive to the ressampling method than the $ZNCC$ matching



Ressampling



$ZNCC$



ℓ_1

Figure 5.4 – Histogram of approximation measures for ressampling methods and matching criteria.

criterion.

Non convex regularization terms In our context the regularization term $\phi_{ij} : \mathbb{R}^d \times \mathbb{R}^d \to \mathbb{R}$ is generally a multi-dimensional bi-variate function composed of a weighting term $w_{ij} \in \mathbb{R}^+$ and a non linear function $\psi : \mathbb{R} \to \mathbb{R}^+$:

$$\phi_{ij}(a, b) = w_{ij} \ \psi(||a - b||_\epsilon), \quad \forall (a, b) \in \mathbb{R}^d \times \mathbb{R}^d$$

Hence, the non-convexity actually entirely comes from the function $\psi(.)$ when it takes for instance one of the following forms:

$$\begin{aligned}
&\text{Truncated:} && \psi(x) = \min(x, \gamma), && \forall (x, \gamma) \in \mathbb{R} \times \mathbb{R}^+ \\
&\text{Cauchy:} && \psi(x) = \frac{1}{1 + \left(\frac{x}{\gamma}\right)^\alpha}, && \forall (x, \gamma, \alpha) \in \mathbb{R} \times \mathbb{R}^+ \times \mathbb{R}^+ \\
&\text{Welsch:} && \psi(x) = \exp\left(-\left(\frac{x}{\gamma}\right)^\alpha\right), && \forall (x, \gamma, \alpha) \in \mathbb{R} \times \mathbb{R}^+ \times \mathbb{R}^+
\end{aligned}$$

We display the regularization functions in figure 5.5.

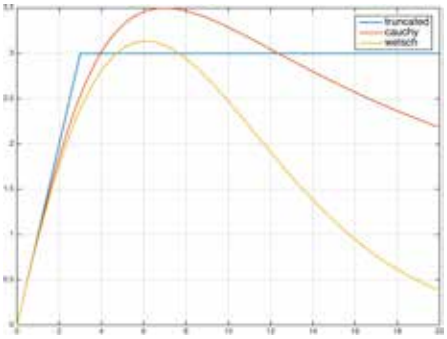


Figure 5.5 – Non convex regularization terms

- Truncated with $\gamma = 3$,
- Cauchy with $\gamma = 7$ and $\alpha = 2$,
- Welsch with $\gamma = 8$ and $\alpha = 1.5$.

Taylor approximation for unary terms

When we are only interested to get a convex surrogate function on a small interval we can rely on Taylor series.

Taylor series An infinite differentiable function f can be represented as an infinite sum of terms, a Taylor series, computed from its derivatives:

$$f(x) = \sum_{n=0}^{\infty} \frac{f^{(n)}(a)}{n!}(x-a)^n$$

We note that the high order terms progressively vanish as x gets closer to the neighborhood of a. This property allows us to build a polynomial of small order that approximately represents a function around a certain point of interest.

For the unary term, we can exploit this idea in two different ways depending on the convexity of the matching criterion.

1st order approximation Here, we assume that the matching criterion ρ is a convex non decreasing function. Moreover, we limit ourself to the first order polynomial functions to approximate the image ressampling. We obtain for a given point of interest $a \in \mathbb{R}^d$:

$$I(a+x) \approx I(a) + (\nabla I(a))^t x$$

This approximation is also sometimes referred to as linearization.

Hence, we obtain for a convex function ρ the resulting approximation:

$$\rho(I(a+x)) \approx \rho(I(a) + (\nabla I(a))^t x)$$

The resulting function is a convex function since it is composed of a convex function, the linear term, with a convex non decreasing function, the matching criterion.

2^{nd} **order approximation** If we can't make the previous assumption on the matching criterion, then we proceed with a second order approximation. We define as $f(.)$ the composition of the image ressampling and the matching criterion. By limiting the Taylor series of f to the second order we get:

$$f(x) \approx f(a) + (\nabla f(a))^t x + x^t \nabla^2 f(a) x$$

We note that this approximation is guaranteed to yield a convex function only if the Hessian matrix $\nabla^2 f(a)$ is semi definite positive for any point a. Hence, we not only set to zero all non diagonal elements but also the negative diagonal elements of the Hessian matrix as in[170]. We refer to this modified Hessian matrix as $\nabla^2_+ f(a)$. Finally, we obtain the following 2^{nd} order convex approximation:

$$f(x) \approx f(a) + (\nabla f(a))^t x + x^t \nabla^2_+ f(a) x$$

Discretization of derivative operators The computation of a Taylor series heavily relies on calculating derivatives of different orders. Considering we work with discretized images we need to define the discretized counterparts of the continuous derivative operators. Different discretization schemes exist. We limit the definition to the case of a mono-dimensional function defined on a rectangular uniform grid. The extension of each scheme to multi-dimensional variables is straight forward. Since we always assume that our data are uniform gridded, we will not discuss the extension of these definitions to non uniform grid or un gridded space. However, we point the curious reader to the following works [124] for more details on discretization.

For the following definitions we make use of the discrete function $f(.,.) : \mathcal{R} \times \mathcal{C} \rightarrow \mathbb{R}$ where $\mathcal{R} \subset \mathbb{N}$ and $\mathcal{C} \subset \mathbb{N}$ index respectively the row and columns of the grid.

First order derivatives The forward, backward and central schemes are the most used approximations to calculate the first order derivatives.

The forward method computes:

$$\nabla f(i,j) = \begin{bmatrix} f(i+1,j) - f(i,j) \\ f(i,j+1) - f(i,j) \end{bmatrix}$$

The backward method computes:

$$\nabla_i f(i,j) = \begin{bmatrix} f(i,j) - f(i-1,j) \\ f(i,j) - f(i,j-1) \end{bmatrix}$$

The central method computes:

$$\begin{aligned} \nabla f(i,j) &= \begin{bmatrix} f(i+0.5,j) - f(i-0.5,j) \\ f(i,j+0.5) - f(i,j-0.5) \end{bmatrix} \\ &= \begin{bmatrix} \frac{f(i+1,j) - f(i-1,j)}{2} \\ \frac{f(i,j+1) - f(i,j-1)}{2} \end{bmatrix} \quad \text{(with linear interpolation)} \end{aligned}$$

We display in figure 5.6 and 5.7 examples of each discretization scheme.

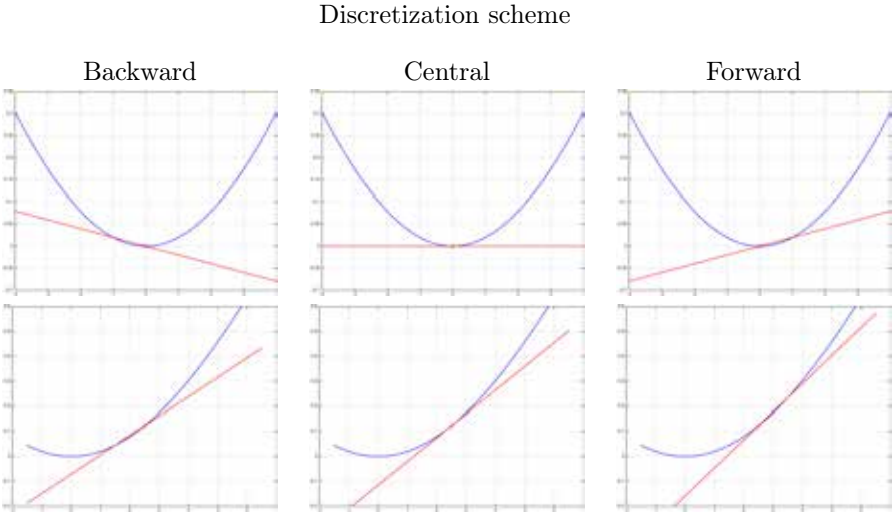


Figure 5.6 – Examples of discretization of the first order derivative: the red line plots the derivative of the blue curve at the point marked by the red diamond shape.

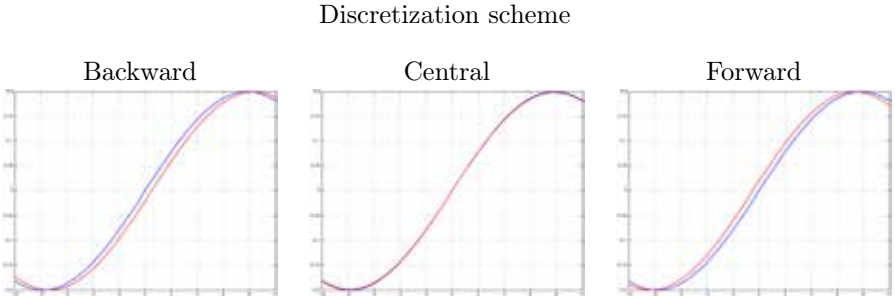


Figure 5.7 – Analytic derivatives vs discretized derivatives for $x \to 1 - \cos(0.2x)$: the red line plots the derivative estimated by the discretization scheme while the blue curve displays the analytic derived derivative.

In agreement with their respective equations, the forward and backward schemes are slightly out of phase. The central discretization scheme seems to perform better. During our experiments, we will pursue this investigation in more depth with functions derived from realistic unary terms.

Second order derivatives The forward, backward and central scheme also extend to second order derivatives. We only gives the diagonals elements

since we always discard the others in our Hessian approximation scheme:

The forward method computes:

$$\nabla^2_{ii} f(i,j) = f(i+2,j) - 2f(i+1,j) + f(i,j)$$
$$\nabla^2_{jj} f(i,j) = f(i,j+2) - 2f(i,j+1) + f(i,j)$$

The backward method computes:

$$\nabla^2_{ii} f(i,j) = f(i,j) - 2f(i-1,j) + f(i-2,j)$$
$$\nabla^2_{jj} f(i,j) = f(i,j) - 2f(i,j-1) + f(i,j-2)$$

The central method computes:

$$\nabla^2_{ii} f(i,j) = f(i+1,j) - 2f(i,j) + f(i-1,j)$$
$$\nabla^2_{jj} f(i,j) = f(i,j+1) - 2f(i,j) + f(i,j+1)$$

Comparison Once more we make use of the stereo matching task of last chapter with the four *UltraCam* subsets. We focus on the unary terms created by matching criteria derived from the Huber norm and the $ZNCC$ criterion. We propose a series of experiments to examine the various discretization schemes and the different orders of approximation. We use bicubic image ressampling as a baseline to compute the approximation measurement for disparities ranging from -1 to 1 along the horizontal axis.

Huber: 1^{st} order vs 2^{nd} order discretization schemes First, we define the matching criterion as the Huber norm with smoothing threshold in $\{0, 5, 10\}$ (images are quantized on 255 gray levels). We investigate the first order approximation scheme with different discretization methods of the differential operator, forward, backward and central schemes. We also study the second order approximation scheme but only for the central differentiation scheme.

The table 5.2 details the approximation measure for all 4 subsets of the stereo matching application of the previous chapter.

As expected, the 1^{st} order central scheme and 2^{nd} order central scheme outperform the backward and forward schemes. Interestingly, the 2^{nd} order central scheme outperforms its 1^{st} order counterparts for the ℓ_1. When the smoothing increases, the 1^{nd} order central scheme slightly outperforms its 2^{nd} order equivalent. Hence, we would only use either the 1^{st} or 2^{nd} order central schemes.

ZNCC: 2^{nd} order discretization scheme For the $ZNCC$ criterion we investigate how the approximation measure evolves with the size of patch from 3×3 to 7×7. We report our results in table 5.3.

The approximation improves with a growing size of the patch. In fact, the $ZNCC$ is made smoother when the patch grows larger. As a result the 2^{nd} order approximation becomes more accurate.

Smoothing threshold of 0: ℓ_1 norm

	Church	Factory	Buildings	Industry	Average
Discretization					
1^{st} order backward	20.14	28.42	19.06	19.30	21.73
1^{st} order central	17.99	24.95	16.89	17.12	19.24
1^{st} order forward	20.02	28.36	18.88	19.21	21.62
2^{nd} order central	16.97	24.96	15.96	16.28	18.54

Smoothing threshold of 10

	Church	Factory	Buildings	Industry	Average
Discretization					
1^{st} order backward	37.15	55.40	34.49	36.07	40.78
1^{st} order central	28.96	40.46	26.87	28.08	31.09
1^{st} order forward	36.78	53.48	34.25	36.05	40.14
2^{nd} order central	29.90	44.98	27.87	29.28	33.01

Smoothing threshold of 20

	Church	Factory	Buildings	Industry	Average
Discretization					
1^{st} order backward	39.13	57.44	36.82	37.74	42.78
1^{st} order central	30.10	41.38	28.17	29.18	32.21
1^{st} order forward	38.86	54.92	36.59	37.84	42.05
2^{nd} order central	31.33	46.43	29.42	30.57	34.44

Table 5.2 – Approximation measurement for different discretization schemes with Huber based matching criterion.

	Church	Factory	Buildings	Industry	Average
Patch size					
3×3	14.13	20.97	11.91	10.86	14.47
5×5	8.12	15.05	5.98	4.97	8.53
7×7	6.34	13.81	4.18	3.20	6.88

Table 5.3 – Approximation measurement for different patch sizes with $ZNCC$ based matching criterion.

Taylor approximation for pairwise terms

For the pairwise terms and the regularization criterion that we investigate, the $1st$ and $2nd$ order approximation lead to exactly the same results. Indeed, since the functions of interest are non convex the $2nd$ order term is always set to 0.

5.2.2 Approach

Ideally, we want to represent with as much accuracy as possible the energy manifold with a convex function around a current solution. This clearly looks like a chicken and egg problem. On the one hand we need a current solution to define the locality where to represent the energy manifold. On the other hand, we need to get a current solution that is as close as possible to the unknown minimum of the energy manifold. Hence, we rely on an iterative scheme that:

1. Creates a surrogate function that approximate the energy manifold within a certain range around the current solution.
2. Computes the minimum of this surrogate functions to get an updated solution.
3. Defines a new range for the next approximation.

Moreover, we also need to quickly compute the convex surrogate function otherwise any computational speed-up gained by using a primal-dual scheme over the graph-cut scheme will be negated. To this end, we propose to work directly on the images used to compute the energy manifold with two classic approaches: Filtering and Coarsening.

5.2.3 Surrogate function via Filtering scheme

The first approach, that we refer to as the Filtering scheme, only modifies the unary terms.

Filtering

We suppose we are given a function $f : \mathbb{R} \to \mathbb{R}$ to smooth and a continuous filter $k : \mathbb{R} \to \mathbb{R}$. The convolution of f by k gives the filtered function $f_s : \mathbb{R} \to \mathbb{R}$:

$$f_s(x) = \int k(y) f(x - y) dy, \quad \forall x \in \mathbb{R}$$

Since we mainly work with discretized functions we remind the definition of the discrete convolution operator. The convolution of a discrete function $f : \mathbb{Z} \to \mathbb{R}$ by a filter $k : \mathbb{K} \to \mathbb{R}$ with limited support $\mathbb{K} \subseteq \mathbb{Z}$ computes:

$$f_s(i) = \sum_{j \in \mathcal{K}} k(j) f(i - j), \quad \forall i \in \mathbb{Z}$$

To simplify notations, we use the operator $\star$ to indicate convolution indifferently of the continuous and discrete domain. For instance the convolution of the function $f(.)$ by filter $k(.)$ is noted $k \star f$ and we also write for all $i \in \mathbb{Z}$, $f_s(i) = (k \star f)(i)$.

If we assume that $k(.)$ is a low pass filter, it reduces the average curvature of the function it is convoluted to. Indeed, the second order derivatives gives

the curvature along each directions. Since the second order derivative is a linear operator we have:

$$(k \star f)'' = k \star (f'')$$

Gaussian filtering

We choose to use a Gaussian filter as low pass filter. The Gaussian filter minimizes group delay making it an ideal time domain filter as explained in [13]. We remind the impulse response function of a Gaussian filter for a real d dimensional variables x:

$$g(x) = \frac{1}{\sqrt{\det(2\pi\Sigma)}} \exp\left(-\frac{1}{2} x^t \Sigma^{-1} x\right) \tag{5.2}$$

The matrix Σ is a *SDP* $d \times d$ matrix controlling the filtering in each of the d direction. For our filtering scheme, we only use some diagonal matrix $\Sigma = \text{diag}(\sigma_1^2, \sigma_2^2, \ldots, \sigma_d^2)$. Hence, the Gaussian filter simplifies to:

$$g(x) = \frac{1}{\sqrt{(2\pi)^d \prod_{j=1}^d \sigma_d}} \exp\left(-\sum_{j=1}^d \frac{x_j^2}{2\sigma_j^2}\right) \tag{5.3}$$

By nature the Gaussian filter has an infinite support. However, we notice that when x grows large the value of $g(x)$ tends to 0. Hence, in our context we only use a limited support for the Gaussian filter. However, we properly renormalize the output of filtering such that a constant signal is unmodified.

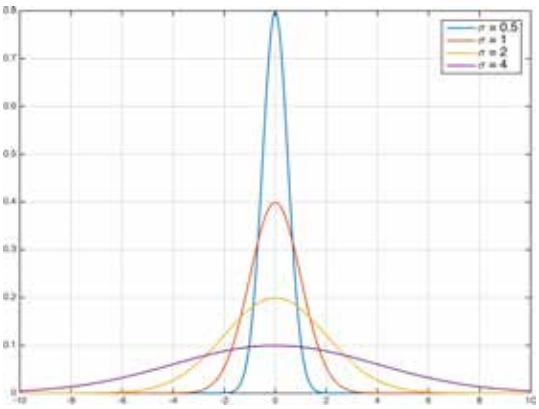


Figure 5.8 – 1-dimensional Gaussian filters for different values of σ.

We display in figure 5.8 the impulse response of 1-dimensional Gaussian filters for different values of σ. We notice that as sigma grows larger, the Gaussian filter has a larger spread, and hence, a lower low pass frequency cut-off.

Reducing curvature and non convexity

Filtering with a low pass filter helps to reduce the curvature. If we now assume that the function $f(.)$ is close to be convex, we can find a low pass filter $k(.)$ that after convolution creates a convex function. And, then we can use this created convex function as a surrogate function in the first order primal dual scheme.

We note that if the function $f(.)$ is far from being convex we then need the filter $k(.)$ to be low pass filter with a potentially very low frequency cut-off to get a convex function. The worst case scenario is when $f(.)$ is strongly concave. The only convex function we can obtain with filtering is the constant function with value corresponding to the average of $f(.)$ over its entire domain.

However, in our context, we are going to filter the images instead of filtering the energy to maintain a low computational complexity. Hence, after filtering we do not seek to obtain a fully convex function but only attempt to reduce the initial non convexity. Moreover, even if the images are smooth, the matching criterion can potentially create matching potentials with high curvature. The figure 5.9 displays the matching term obtained with the $ZNCC$ matching criterion in the context of stereo matching.

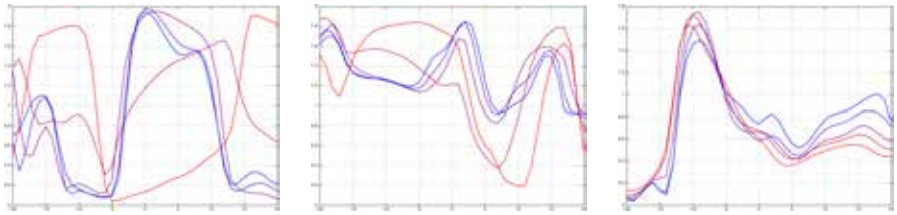


Figure 5.9 – $ZNCC$ matching cost. From blue to red the images used to compute the $ZNCC$ are more and more smoothed. While the smoothing can reduce the non convexity it also creates poor approximation of the original curve (bluest curve).

The filtering scheme

To simplify notations in the algorithm 16 description we use the function $\phi_{i,I_r,I_t,d_i} : [-1,1] \times [-1,1] \rightarrow \mathbb{R}$ to represent the combination of the Taylor approximation of image I_t around d_i and the matching criterion ϕ.

5.2.4 Surrogate function via coarsening

The second approach, referred as the coarsening scheme, is well established in the computer vision community. Hence, we only give a brief description of it. We refer the curious reader to [1] for more details.

Coarsening

The coarsening scheme simply downsamples the images used to compute the energy manifold by a given factor, and then computes a coarsen energy manifold from the downsampled images. We remind that image downsampling simply consists of a low pass filtering followed by a decimation. Hence, the coarsening scheme can be seen as an extension of the Filtering scheme. To simplify the

Algorithm 16: Image filtering scheme

Data: Inputs: I_r, I_t, $(\phi_i)_i$, ψ, Σ_{list}
Result: $(d_i)_i$

Initialize displacement field $\rightarrow d = \mathbf{0}$

Compute weights of regularization $(w_{ij})_{ij}$ from I_r.
for $\sigma \in \Sigma_{list}$ **do**
  Compute the Gaussian filter $\rightarrow g_\sigma$

  Apply filtering on both image:

$$I_{r,\sigma} = g_\sigma \star I_r \quad \text{and} \quad I_{t,\sigma} = g_\sigma \star I_t$$

  while $(d_i)_i$ *is updated* **do**
    Compute the Taylor approximation of unary term $\rightarrow \phi_{i,I_{r,\sigma},I_{t,\sigma},d_i}$

    Solve optimization problem:

$$\min_{x_i \in [-1,1]^2} \sum_{i \in \mathcal{V}} \phi_{i,I_{r,\sigma},I_{t,\sigma},d_i}(x_i) + \sum_{(i,j) \in \mathcal{E}} w_{ij} \psi(||d_i + x_i - d_j - x_j||_\epsilon)$$

    Update displacement:

$$d_i \leftarrow d_i + x_i, \quad \forall i \in \mathcal{V}$$

notations we denote the upsampling by the $\uparrow\uparrow$ symbol and the downsampling with the $\downarrow\downarrow$ symbol.

We describe the Coarsening scheme in algorithm 17.

5.2.5 Experiments

We proceed to evaluate the Filtering and the Coarsening schemes. We make use of the stereo matching application of last chapter. Since we optimize non-convex energy we use a baseline reference the solution obtained with the graph-cuts optimization technique of chapter 4. For both schemes, we monitor the energy ratio with respect to the energy obtained with Fast-PD. We also compute the mean error in pixel using ℓ_1 norm with respect to the solution of Fast-PD.

Filtering vs coarsening

Our first experiment directly compares the filtering scheme to the coarsening scheme.

Algorithm 17: Coarsening scheme

Data: Inputs: I_r, I_t, $(\phi_i)_i$, ψ, Σ_{list}
Result: $(d_i)_i$

Initialize displacement field at coarsest scale $\rightarrow d = \mathbf{0}$
for $\sigma \in \Sigma_{list}$ **do**
  Compute the Gaussian filter $\rightarrow g_\sigma$

  Downsample both image:

$$I_{r,\sigma} = \downarrow\downarrow (g_\sigma \star I_r) \quad \text{and} \quad I_{t,\sigma} = \downarrow\downarrow (g_\sigma \star I_t)$$

  Compute weights of regularization $(w_{ij,\sigma})_{ij}$ from $I_{r,\sigma}$.
  while $(d_i)_i$ *is updated* **do**
    Solve optimization problem:

$$\min_{x_i \in [-1,1]^2} \sum_{i \in \mathcal{V}_\sigma} \phi_{i,I_{r,\sigma},I_{t,\sigma},d_i}(x_i) + \sum_{(i,j)\in\mathcal{E}_\sigma} w_{ij,\sigma}\psi(||d_i + x_i - d_j - x_j||_\epsilon)$$

    Update displacement field:

$$d_i \leftarrow d_i + x_i, \quad \forall i \in \mathcal{V}_\sigma$$

  Upsample displacement field for next scale:

$$d \leftarrow \uparrow\uparrow d$$

For both schemes, we compute 3 successive Taylor approximations per scale and we perform 30 iterations of the Primal-Dual solver for each Taylor approximation. We found that increasing the number of iterations of the Primal-Dual solver is only slightly modifying the results. For the filtering scheme we use a 2D Gaussian filter with standard deviation ranging from 7 to 0.5 with a decrement of 0.5. A last iteration is performed with the unfiltered image. For the coarsening scheme we use a 0.66 downsampling factor with 8 scales. We use the bicubic filter to interpolate the images.

Table 5.4 summarizes the results.

From table 5.4 we see that for both schemes the energy ratio is mostly under 1.10 and the mean error is under 0.6. Both schemes are always out-performed by the Fast-PD baseline. Hence, the non-convexity of the energy manifold clearly needs to be taken into account by the optimization scheme. The figure 5.10 highlights the difference between the two schemes and the baseline.

For most experiments the Filtering scheme yields a lower energy than the Coarsening scheme. However, we see that Mean Error criterion is often significantly better for the Coarsening scheme. This indicates that the Filtering

	Small size problems				
	Church	Factory	Buildings	Industry	Average
Filtering					
Energy (ratio)	1.02	1.03	1.04	1.02	1.03
Mean Error	0.30	0.28	0.27	0.27	0.28
Coarsening					
Energy (ratio)	1.13	1.12	1.15	1.11	1.13
Mean Error	0.33	0.30	0.30	0.30	0.31

	Medium size problems				
	Church	Factory	Buildings	Industry	Average
Filtering					
Energy (ratio)	1.03	1.06	1.04	1.03	1.04
Mean Error	0.40	0.45	0.40	0.40	0.41
Coarsening					
Energy (ratio)	1.09	1.08	1.12	1.08	1.09
Mean Error	0.44	0.34	0.42	0.42	0.40

	Large size problems				
	Church	Factory	Buildings	Industry	Average
Filtering					
Energy (ratio)	1.05	1.11	1.07	1.03	1.06
Mean Error	0.62	0.93	0.65	0.59	0.70
Coarsening					
Energy (ratio)	1.08	1.07	1.10	1.06	1.08
Mean Error	0.53	0.42	0.57	0.57	0.52

Table 5.4 – Comparing the filtering and coarsening scheme.

scheme has found a solution of relative low energy, but higher than Fast-PD, that significantly differs from the Fast-PD solution.

A visual inspection of the disparity maps confirms this phenomenon. For instance, we see in figure 5.10 that the roof of the church is poorly estimated by the filtering scheme.

The filtering scheme creates artifacts for the church and factory subsets. Hence, for the following experiments we discard the filtering scheme to only retain the coarsening scheme.

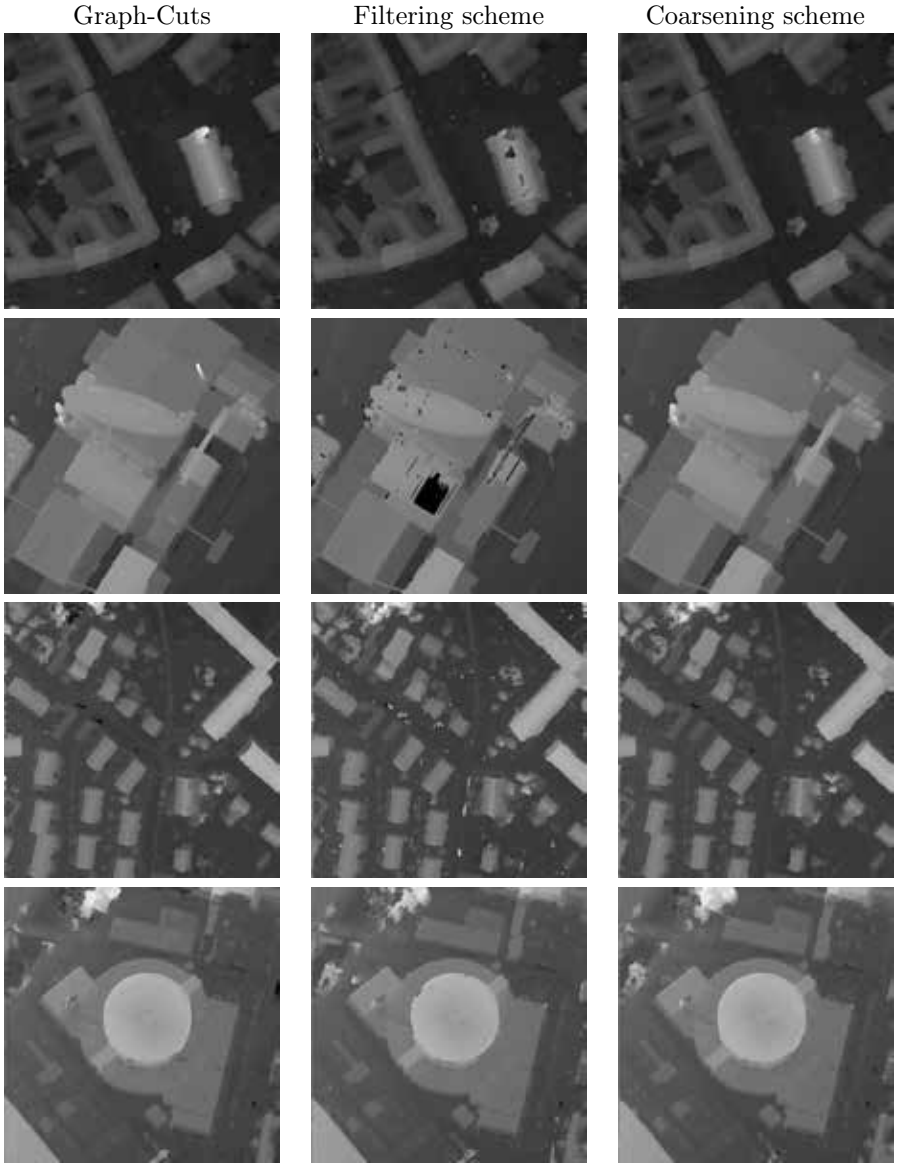


Figure 5.10 – Comparing the filtering and coarsening schemes for the subsets at full resolution. We notice that neither the Filtering and Coarsening scheme resolve fine details such as the church's bell tower or the factory chimney. Moreover, the Filtering scheme produces unacceptable artifacts.

Coarsening 1D vs 2D

We now investigate the differences between the 1D and 2D coarsening. In the context of the stereo-matching, we deal with a 1D problem. Hence, instead of 2D

filtering and decimation we can only filter and decimate in the axis confounded with the epipolar lines.

	Small size problems				
	Church	Factory	Buildings	Industry	Average
Coarsening 1D					
Energy (ratio)	1.12	1.13	1.17	1.12	1.13
Mean Error	0.33	0.31	0.30	0.30	0.31
Coarsening 2D					
Energy (ratio)	1.13	1.12	1.15	1.11	1.13
Mean Error	0.33	0.30	0.30	0.30	0.31

	Medium size problems				
	Church	Factory	Buildings	Industry	Average
Coarsening 1D					
Energy (ratio)	1.09	1.08	1.13	1.08	1.09
Mean Error	0.44	0.35	0.44	0.41	0.41
Coarsening 2D					
Energy (ratio)	1.09	1.08	1.12	1.08	1.09
Mean Error	0.44	0.34	0.42	0.42	0.40

	Large size problems				
	Church	Factory	Buildings	Industry	Average
Coarsening 1D					
Energy (ratio)	1.08	1.07	1.12	1.06	1.08
Mean Error	0.57	0.43	0.65	0.55	0.55
Coarsening 2D					
Energy (ratio)	1.08	1.07	1.10	1.06	1.08
Mean Error	0.53	0.42	0.57	0.57	0.52

Table 5.5 – Comparing 1D and 2D coarsening schemes.

For both coarsening scheme the results are very similar. Both the visual inspection of figure 5.11 and the review of table 5.5 show no significant improvement of the 1D coarsening scheme over the 2D coarsening scheme. From now on, we only retain the latter because of its better computational cost.

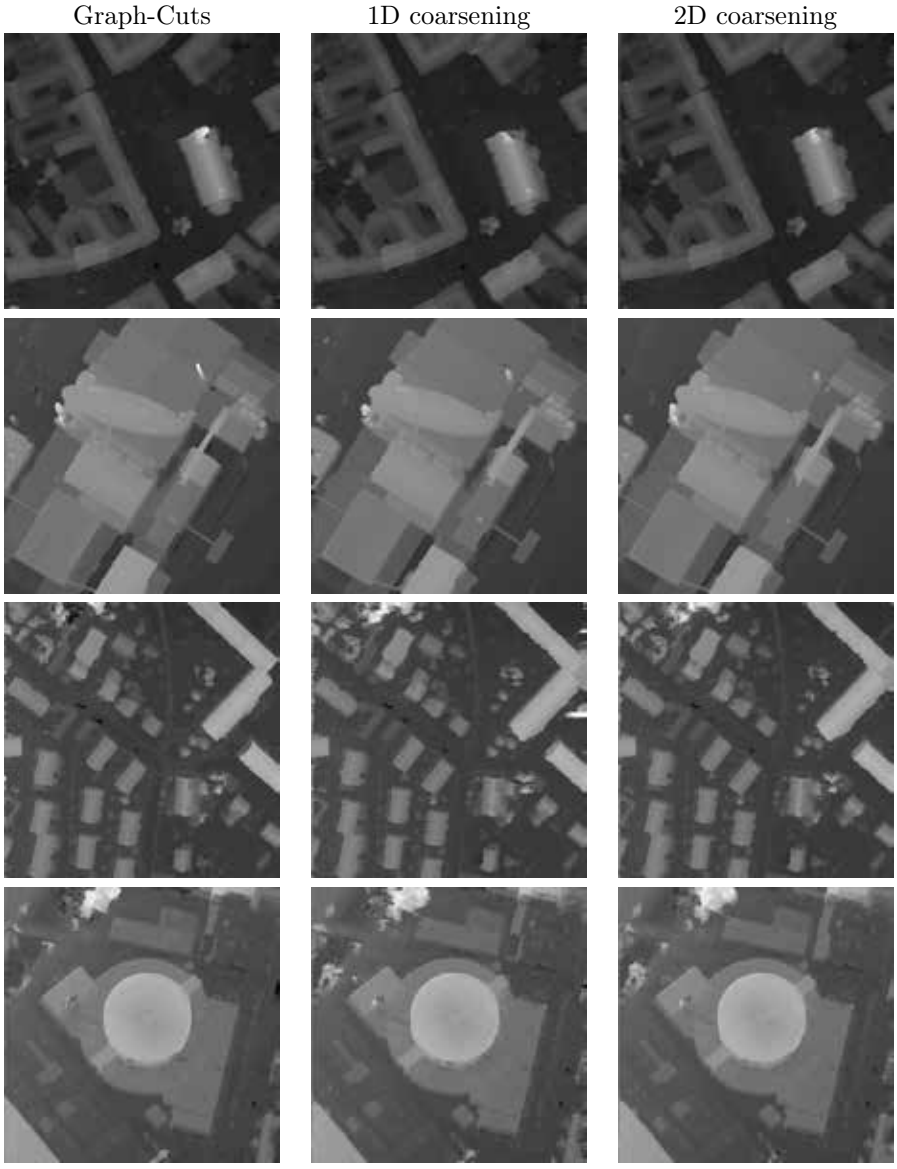


Figure 5.11 – Comparing 1D and 2D coarsening scheme. Both schemes give essentially the same results.

Downsampling

We perform a last experiment that investigates the impact of the downsampling factor. We vary the downsampling ratio from 0.5 to 0.8 with 0.1 increment. For each ratio we adequately tune the number of scales such that the coarsest scale

represents approximately the same downsampling factor with respect to the full resolution images.

	Small size problems				
	Church	Factory	Buildings	Industry	Average
Coarsening 0.5					
Energy (ratio)	1.13	1.13	1.17	1.12	1.14
Mean Error	0.33	0.31	0.30	0.30	0.31
Coarsening 0.6					
Energy (ratio)	1.13	1.12	1.16	1.12	1.13
Mean Error	0.33	0.30	0.30	0.30	0.31
Coarsening 0.7					
Energy (ratio)	1.13	1.12	1.16	1.11	1.13
Mean Error	0.33	0.30	0.30	0.30	0.31
Coarsening 0.8					
Energy (ratio)	1.12	1.12	1.15	1.11	1.12
Mean Error	0.33	0.30	0.30	0.30	0.31

Table 5.6 – Comparing downsampling factor of the coarsening scheme for small size problems.

	Medium size problems				
	Church	Factory	Buildings	Industry	Average
Coarsening 0.5					
Energy (ratio)	1.10	1.09	1.13	1.09	1.10
Mean Error	0.45	0.36	0.43	0.44	0.42
Coarsening 0.6					
Energy (ratio)	1.10	1.09	1.12	1.08	1.09
Mean Error	0.44	0.35	0.42	0.42	0.41
Coarsening 0.7					
Energy (ratio)	1.09	1.08	1.11	1.07	1.09
Mean Error	0.43	0.33	0.41	0.41	0.40
Coarsening 0.8					
Energy (ratio)	1.09	1.07	1.11	1.07	1.08
Mean Error	0.43	0.33	0.41	0.40	0.39

Table 5.7 – Comparing downsampling factor of the coarsening scheme for medium size problems.

The tables 5.6, 5.7 and 5.8 all indicate that a progressive downsampling yields better results. However, the improvement is relatively small as we can see in figure 5.12. Moreover, high value downsampling factors require more scales which lead to larger computational complexity. For those reasons we advocate to choose a downsampling factor between 0.5 and 0.6.

	Large size problems				
	Church	Factory	Buildings	Industry	Average
Coarsening 0.5					
Energy (ratio)	1.09	1.08	1.11	1.07	1.09
Mean Error	0.57	0.46	0.60	0.60	0.56
Coarsening 0.6					
Energy (ratio)	1.09	1.07	1.10	1.07	1.08
Mean Error	0.54	0.44	0.57	0.58	0.53
Coarsening 0.7					
Energy (ratio)	1.08	1.07	1.09	1.06	1.07
Mean Error	0.53	0.41	0.56	0.58	0.52
Coarsening 0.8					
Energy (ratio)	1.07	1.06	1.08	1.06	1.07
Mean Error	0.52	0.40	0.56	0.58	0.51

Table 5.8 – Comparing downsampling factor of the coarsening scheme for large size problems.

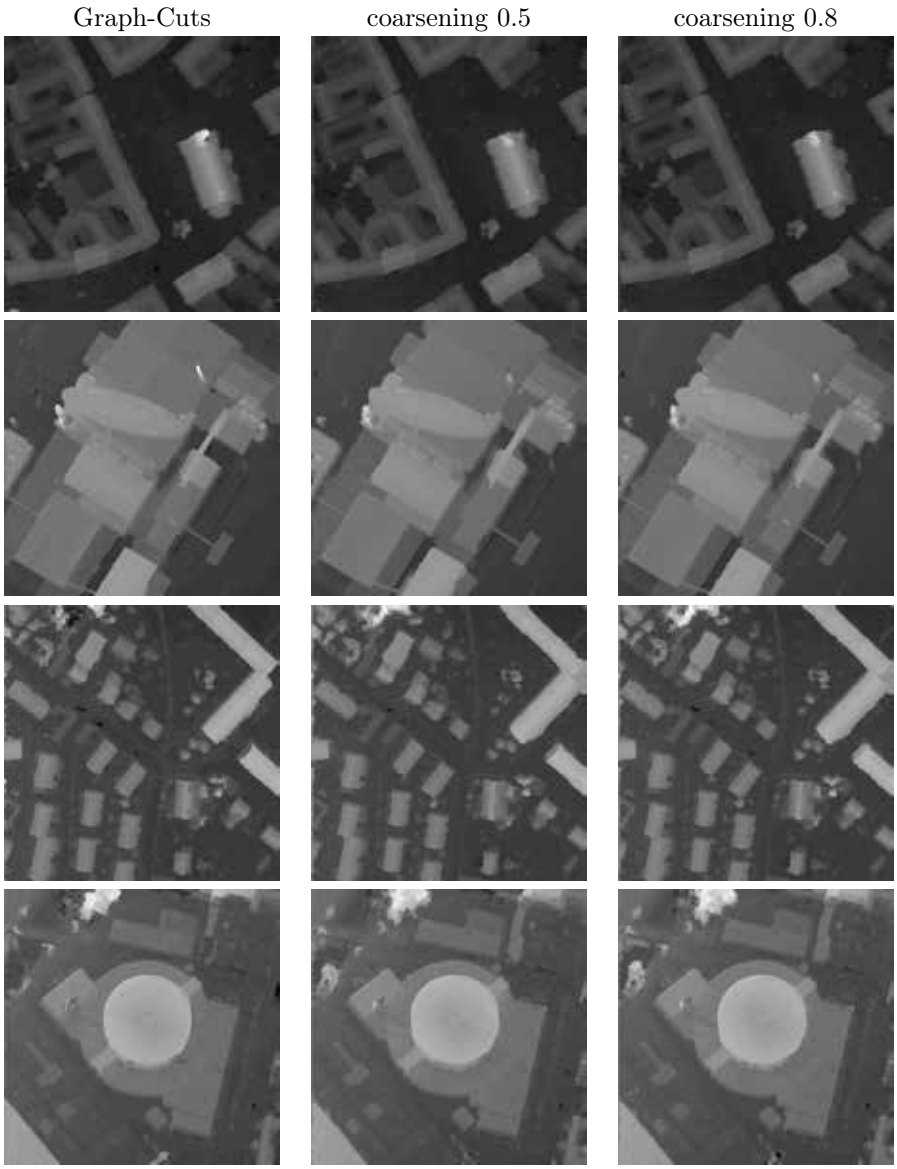


Figure 5.12 – Impact of downsampling factor for the coarsening scheme.

5.3 Coarsening scheme for Graph-Cuts solvers

We now introduce coarsening schemes for Graph-Cuts solvers.

5.3.1 Pairwise undirected discrete MRF

Let us recall the notations. We represent a pairwise undirected discrete MRF model $\mathcal{M}$ as follow:

$$\mathcal{M} = \left(\mathcal{V}, \mathcal{E}, \mathcal{L}, \{\phi_i\}_{i\in\mathcal{V}}, \{\phi_{ij}\}_{(i,j)\in\mathcal{E}}\right). \tag{5.4}$$

The support graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ is composed of a set of nodes $\mathcal{V}$ and a set of undirected edges $\mathcal{E}$. The random variable $x = (x_i)_{i\in\mathcal{V}}$ defines the state of each node of $\mathcal{V}$, and takes values in the discrete label space $\mathcal{L} \subset \mathbb{N}$: we assume without loss of generality the same state space for all random variables with each label of $\mathcal{L}$ representing one of these possible states. For each node $i \in \mathcal{V}$, the potential function $\phi_i : \mathcal{L} \to \mathbb{R}$ encodes the unary cost of variables x_i. For each edge $(i, j) \in \mathcal{E}$, the potential function $\phi_{ij} : \mathcal{L}^2 \to \mathbb{R}$ encodes the pairwise cost of variable x_i and x_j. The collection of functions $\{\phi_i\}_{i\in\mathcal{V}}$ and $\{\phi_{ij}\}_{(i,j)\in\mathcal{E}}$ form the unary potentials and, respectively, the pairwise potential. We also assume that the unary and pairwise potentials are computed from the reference and target images as for the stereo-matching application.

The energy of the MRF, given a solution x, computes the sum of the potentials:

$$E(x|\mathcal{M}) = \sum_{i\in V} \phi_i(x_i) + \sum_{(i,j)\in\mathcal{E}} \phi_{ij}(x_i, x_j) \tag{5.5}$$

The MAP inference computes a configuration of minimum energy over the entire solution space $\mathcal{L}^{|\mathcal{V}|}$:

$$x_{\text{MAP}} = \arg\min_{x\in\mathcal{L}^{|\mathcal{V}|}} E(x|\mathcal{M}) \tag{5.6}$$

5.3.2 Image pyramid

This first coarsening scheme mimics the image pyramid scheme for first order primal dual method. The key idea is to create a coarse MRF model by computing the potentials from downsampled reference and target images as in [146] or [62].

5.3.3 Energy pyramid

Another coarsening method introduced in [35], [34], [49] or [97] relies on first computing the potential of the MRF model at the finest scale, and then directly apply the coarsening on the MRF model.

Coarsening

To coarsen an MRF model, we need to define how to coarsen the nodes (spatial component) but also how to coarsen the labels (domain component).

Algorithm 18: Image coarsening scheme for Graph-Cuts optimization

Data: Inputs: I_r, I_t, Σ_{list}
Result: $(d_i)_i$

Initialize displacement field at coarsest scale $\rightarrow d = \mathbf{0}$
for $\sigma \in \Sigma_{list}$ **do**
  Compute the Gaussian filter $\rightarrow g_\sigma$

  Downsample both image:

$$I_{r,\sigma} = \downarrow\downarrow (g_\sigma \star I_r) \quad \text{and} \quad I_{t,\sigma} = \downarrow\downarrow (g_\sigma \star I_t)$$

  Compute coarsen MRF model $\mathcal{M}_\sigma$ from $I_{r,\sigma}$ and $I_{t,\sigma}$.

  Convert the displacement field d to label x_σ

  Optimize MRF model to obtain updated labeling x_σ

  Update displacement field d from labeling x_σ

  Upsample displacement field for next scale:

$$d \leftarrow \uparrow\uparrow d$$

Node coarsening The nodes coarsening reduces the solution space by grouping together nodes of the input MRF's support graph. From the support graph $\mathcal{G}$ of an input model $\mathcal{M}$, the node grouping function $g_n : \mathcal{V} \rightarrow \mathcal{V}'$ constructs the coarsen support graph $\mathcal{G}' = (\mathcal{V}', \mathcal{E}')$:

$$\mathcal{V}' = \{g_n(i) : i \in \mathcal{V}\} \tag{5.7}$$
$$\mathcal{E}' = \{(g_n(i), g_n(j) : (i,j) \in \mathcal{E}, g_n(i) \neq g_n(j)\} \tag{5.8}$$

An example of grouping function and its associated coarsening is shown in Fig. 5.13.

The coarse model $\mathcal{M}' = \left(\mathcal{V}', \mathcal{E}', \mathcal{L}, \{\phi'_i\}_{i \in \mathcal{V}'}, \{\phi'_{ij}\}_{(i,j) \in \mathcal{E}'}\right)$ naturally inherits the original support graph and its potentials are computed from the potentials of the input model $\mathcal{M}$ and the node coarsening function. The coarse unary potentials are computed as:

$$(\forall i' \in \mathcal{V}'), \quad \phi'_{i'}(l) = \sum_{i \in \mathcal{V} | i' = g_n(i)} \phi_i(l) + \sum_{(i,j) \in \mathcal{E} | i' = g_n(i), i' = g_n(j)} \phi_{ij}(l,l) \tag{5.9}$$

The coarse pairwise potentials are defined as:

$$(\forall (i',j') \in \mathcal{E}'), \quad \phi'_{i'j'}(l_0, l_1) = \sum_{(i,j) \in \mathcal{E} | i' = g_n(i), j' = g_n(j)} \phi_{ij}(l_0, l_1) \tag{5.10}$$

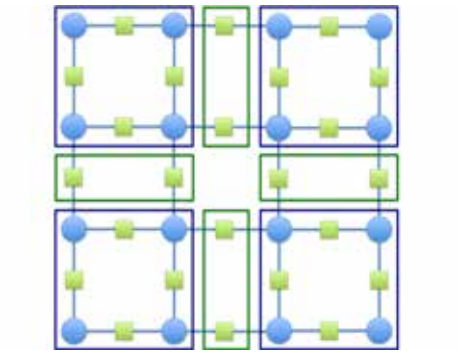


Figure 5.13 – The blue circles and the green square represent the unary potentials attached to nodes of $\mathcal{V}$ and the pairwise potentials associated to edges $\mathcal{E}$ respectively. The blue squares represent the coarse unary potentials of nodes in $\mathcal{V}'$ and the green rectangle are the coarse pairwise potentials of edges belonging to $\mathcal{E}'$.

Slightly abusing mathematical notation, we denote the coarse model $g_n(\mathcal{M}) = \mathcal{M}'$. We also make use of $g_n^{-1}(x') = x$, where $x_i = x'_{g_n(i)}$ for each $i \in \mathcal{V}$, to "upsample" a solution x' of $\mathcal{M}'$ to a solution of x of $\mathcal{M}$.

The node coarsening implies that the nodes of $\mathcal{V}$ grouped together are assigned the same state.

Label coarsening The label space coarsening reduces the solution space by coarsening the label space of each random variables. From the input label space $\mathcal{L}$ we make use of a label grouping function $g_l : \mathcal{L} \to \mathcal{N}$ to compute the coarse label space $\mathcal{L}'$:

$$\mathcal{L}' = \{g_l(l) : l \in \mathcal{L}\}. \tag{5.11}$$

The coarse model $\mathcal{M}' = \left(\mathcal{V}, \mathcal{E}, \mathcal{L}', \{\phi'_i\}_{i\in\mathcal{V}}, \{\phi'_{ij}\}_{(i,j)\in\mathcal{E}}\right)$ inherits the coarse label space $\mathcal{L}'$ and its potentials are computed from the potentials of the input model $\mathcal{M}$ and the label coarsening function g_l. The coarse unary potentials are computed as:

$$(\forall l' \in \mathcal{L}'), \quad \phi'_i(l') = \sum_{l'\in\mathcal{L}|l'=g_l(l)} \phi_i(l) \tag{5.12}$$

The coarse pairwise potentials computation follows:

$$(\forall (l_0, l_1) \in \mathcal{L}\times\mathcal{L}), \quad \phi'_{ij}(l'_0, l'_1) = \sum_{(l_0,l_1)\in\mathcal{L}\times\mathcal{L}|l'_0=g_l(l_0),l'_1=g_l(l_1)} \phi_{ij}(l_0, l_1) \tag{5.13}$$

Composing node and label coarsenings One interesting property of the node and label coarsening operators is that they are commutative with respect to the composition operator. This means that applying the node coarsening

before the label coarsening gives exactly the same coarse model as applying the label coarsening before the node coarsening. In the following paragraph, we will simply use the term of coarsening and the coarsening function g where $g = g_n \circ g_l$.

Energy pyramid scheme

Label pruning In this coarsening framework we keep track of the status of each label of each node with the *pruning matrix* $A : \mathcal{V} \times \mathcal{L} \to \{0, 1\}$. This matrix indicates whether a label of a node is active or pruned. Only active label can be part of the optimization solution.

$$A(i,l) = \begin{cases} 1 & \text{if label } l \text{ is active at vertex } i \\ 0 & \text{if label } l \text{ is pruned at vertex } i \end{cases} \tag{5.14}$$

During the coarse-to-fine scheme, we also make use of a pruned solution space that restricts the initial solution space to solutions formed with only active labels:

$$\mathcal{S}(\mathcal{M}, A) = \left\{ x \in \mathcal{L}^{|\mathcal{V}|} \mid (\forall i), A(i, x_i) = 1 \right\} \ .$$

Pruning function To define whether a label is active or pruned we make use of the heuristically defined pruning function p. For instance, in the stereo-matching context, for a given node, labels that represent a potential displacement that are far from the current solution is good pruning candidates.

Coarse-to-fine optimization Given an input model $\mathcal{M}$ and a sequence of N grouping functions $(g^{(s)})_{0 \leqslant s < N}$, our framework first computes a series of $N + 1$ progressively coarser models $(\mathcal{M}^{(s)})_{0 \leqslant s \leqslant N}$:

$$\mathcal{M}^{(0)} = \mathcal{M} \quad \text{and} \quad (\forall s),\ \mathcal{M}^{(s+1)} = g^{(s)}(\mathcal{M}^{(s)}) \ . \tag{5.15}$$

This builds a coarse-to-fine representation of the input model, where each scale $s \in 0 \leqslant s < N$ is populated by a model $\mathcal{M}^{(s)}$ and a pruning matrix $A^{(s)}$. Each coarse model is defined as:

$$\mathcal{M}^{(s)} = \left(\mathcal{V}^{(s)}, \mathcal{E}^{(s)}, \mathcal{L}, \{\phi_i^{(s)}\}_{i \in \mathcal{V}^{(s)}}, \{\phi_{ij}^{(s)}\}_{(i,j) \in \mathcal{E}^{(s)}} \right) \tag{5.16}$$

Our progressive MAP estimation framework starts with the coarsest scale N, and initializes all elements of its pruning matrix $A^{(N)}$ to 1, i.e., all labels are active. From this point, we repeat an iterative procedure at each scale from the coarsest to the finest scale. At scale s, our framework applies the following steps:

i. Compute the MAP-solution or its approximation (via any existing MRF optimization method) of the model $\mathcal{M}^{(s)}$ over the reduced solution space $\mathcal{S}(\mathcal{M}^{(s)}, A^{(s)})$:

$$x^{(s)} \approx \arg\min_{x \in \mathcal{S}(\mathcal{M}^{(s)}, A^{(s)})} E(x | \mathcal{M}^{(s)}) \ .$$

ii. Given the estimated MAP-solution $x^{(s)}$, compute the feature map, defined in 5.3.4, $f^{(s)} : \mathcal{V}^{(s)} \times \mathcal{L} \to \mathbb{R}^K$

iii. Label pruning: Update the pruning matrix $A^{(s)}$ with the pruning function.

iv. Upsample the pruning matrix $A^{(s)}$ to the next scale:

$$(\forall i \in \mathcal{V}^{(s-1)},\ \forall l \in \mathcal{L}), \quad A^{(s-1)}(i,l) = A^{(s)}(g_n^{(s-1)}(i), g_l^{(s-1)}(l)) \ .$$

v. Upsample solution $x^{(s)}$ to the next scale to warm start next MAP-inference $\forall i \in \mathcal{V}^{(s-1)}$: $x_i^{(s-1)} \leftarrow x^{(s)}_{g^{(s-1)}(i)}$

The Energy pyramid scheme algorithm

The pseudocode of the resulting algorithm appears in Algo. 19.

Algorithm 19: Energy pyramid scheme

Data: Model $\mathcal{M}$, grouping functions $(g^{(s)})_{0 \leqslant s < N}$, classifiers $(z^{(s)})_{0 < s \leqslant N}$
Result: $x^{(0)}$
Compute the coarse to fine sequence of MRFs:
$\mathcal{M}^{(0)} \leftarrow \mathcal{M}$
for $s = [0 \dots N-1]$ **do**
  $\mathcal{M}^{(s+1)} \leftarrow g^{(s)}(\mathcal{M}^{(s)})$
Optimize the coarse to fine sequence of MRFs over pruned solution spaces:
Initialize $x^{(N)}$ and $A^{(N)} \equiv 1$
for $s = [N...0]$ **do**
  Update $x^{(s)}$ by MAP inference: $x^{(s)} \approx \arg\min_{x \in \mathcal{S}(\mathcal{M}^{(s)}, A^{(s)})} E(x|\mathcal{M}^{(s)})$
  if $s \neq 0$ **then**
    Label pruning: update current pruning matrix $A^{(s)}$
    Upsample to next finest pruning matrix: $A^{(s-1)} \leftarrow [g^{(s-1)}]^{-1}(A^{(s)})$
    Upsample $x^{(s)}$ to initialize solution $x^{(s-1)}$ of next scale:
    $x^{(s-1)} \leftarrow [g^{(s-1)}]^{-1}(x^{(s)})$

5.3.4 Inference by Learning

We propose the enhance the energy pyramid scheme by revisiting how the labels are pruned. Instead of relying on pruning heuristics we make use of learning techniques to define the pruning function as in [35].

Pruning: features and classifiers

Rightly deciding on which label to prune at each scale is essential for the energy pyramid scheme. Wrongly pruning labels belonging to the MAP solution leads to decreasing the accuracy of the estimated solution at the finest scale. Maintaining

too many labels active means to explore at each scale a large solution space during inference, resulting in a poor speed-up for the overall MAP estimation. The efficiency of our pruning approach relies on both the feature map $f^{(s)}$ and the off-line trained classifiers $z^{(s)}$. We need the feature map to be discriminative but efficient to compute and the classifiers to be precise but fast to apply.

Features We form the feature map $f^{(s)} : \mathcal{V}^{(s)} \times \mathcal{L} \to \mathbb{R}^K$ by stacking K individual real-valued features defined on $\mathcal{V}^{(s)} \times \mathcal{L}$. As in [35], we focus on generality rather than any dedicated computer vision task. Hence, we only compute features that depend on the energy function and its current solution $x^{(s)}$, letting to future research the definition of task specific features. At each scale we compute the following features:

Strength of discontinuity We compute the $\mathrm{SOD}^{(s)}$ feature to account where the solution $x^{(s)}$ is going against the regularization prior enforcing smoothness. For each node in $\mathcal{V}^{(s)}$ we compute the sum of the potential of its related pairwise terms:

$$\mathrm{SOD}^{(s)}(i,l) = \sum_{j|ij\in\mathcal{E}^{(s)}} \phi_{ij}(x_i^{(s)}, x_j^{(s)}) \tag{5.17}$$

Local energy variation This feature computes the normalized discretized gradient of the energy manifold around the current solution $x^{(s)}$. The local energy variation feature, $\mathrm{LEV}^{(s)}$, is defined for any $i \in \mathcal{V}^{(s)}$ and $l \in \mathcal{L}$ as follows:

$$\mathrm{LEV}^{(s)}(i,l) = \frac{\phi_i^{(s)}(l) - \phi_i^{(s)}(x_i^{(s)})}{N_{\mathcal{V}}^{(s)}(i)} + \sum_{j:(i,j)\in\mathcal{E}^{(s)}} \frac{\phi_{ij}^{(s)}(l, x_j^{(s)}) - \phi_{ij}^{(s)}(x_i^{(s)}, x_j^{(s)})}{N_{\mathcal{E}}^{(s)}(i)} \tag{5.18}$$

with $N_{\mathcal{V}}^{(s)}(i) = \mathrm{card}\{i' \in \mathcal{V}^{(s-1)} : g^{(s-1)}(i') = i\}$ and $N_{\mathcal{E}}^{(s)}(i) = \mathrm{card}\{(i',j') \in \mathcal{E}^{(s-1)} : g^{(s-1)}(i') = i, g^{(s-1)}(j') = j\}$.

Unary "coarsening" This feature $\mathrm{UC}^{(s)}$ aims at estimating the amount of information lost during the coarsening. It is defined for any $i \in \mathcal{V}^{(s)}$ and $l \in \mathcal{L}$:

$$\mathrm{UC}^{(s)}(i,l) = \sum\nolimits_{i'\in\mathcal{V}^{(s-1)}|g^{(s-1)}(i')=i} \frac{|\phi_{i'}^{(s-1)}(l) - \frac{\phi_i^{(s)}(l)}{N_{\mathcal{V}}^{(s)}(i)}|}{N_{\mathcal{V}}^{(s)}(i)} \tag{5.19}$$

Distance to the current label This feature $\mathrm{DL}^{(s)}$ is defined for pairwise terms (ϕ_{ij}) that can be expressed as a combination of a so-called distance function $d : \mathcal{L} \times \mathcal{L} \to \mathbb{R}^+$ and a scalar weight w_{ij}:

$$\phi_{ij}(l_0, l_1) = w_{ij} d(l_0, l_1) \tag{5.20}$$

It computes for each node i and each label the minimum "distance" given by d to the active label over all nodes of the k-ring of node i [1]. We denote $\mathcal{R}_k^{(s)}(i)$ all the nodes belonging to the k-ring of node $i \in \mathcal{V}^{(s)}$. For each node $i \in \mathcal{V}^{(s)}$ and label $l \in \mathcal{L}$, we compute:

$$\mathrm{DL}^{(s)}(i,l) = \min_{j \in \mathcal{R}_k^{(s)}(i)} d(x^{(s)}(j), l) \tag{5.21}$$

Feature normalization For a given computer vision task, parameters might change from one input to the other leading to scaling factors independently applied on the unary and pairwise terms. Hence, we normalize the $\mathrm{SOD}^{(s)}$, $\mathrm{LEV}^{(s)}$, $\mathrm{UC}^{(s)}$ and $\mathrm{DL}^{(s)}$ features to make them insensitive to such scaling. To this en, we divide each of these feature by the average of the minimum values over all labels for each node.

Learning the cascade of classifiers

Defining the pruning ground truth We train the classifiers from a given training set of MRFs all formulating the same class of computer vision tasks, e.g., stereo-matching. For each MRF of the training set, we apply the algorithm 21 without any pruning (i.e., $A^{(s)} \equiv 1$). At the finest scale, we obtain an (approximate) MAP solution. To compute the pruning ground truth, we make use at each scale of the binary function $X_{\mathrm{MAP}}^{(s)} : \mathcal{V}^{(s)} \times \mathcal{L} \to \{0,1\}$. For the finest scale, we can simply convert the approximate MAP solution found:

$$(\forall i \in \mathcal{V}, \forall l \in \mathcal{L}), \quad X_{\mathrm{MAP}}^{(0)}(i,l) = \begin{cases} 1, & \text{if } l \text{ is the MAP label of node } i \\ 0, & \text{otherwise} \end{cases} \tag{5.22}$$

To obtain the pruning ground truth at each scale $s > 0$, we simply iteratively apply the grouping functions:

$$(\forall i \in \mathcal{V}^{(s)}, \forall l \in \mathcal{L}), \quad X_{\mathrm{MAP}}^{(s)}(i,l) = \bigvee_{i' \in \mathcal{V}^{(s-1)} : g^{(s)}(i') = i} X_{\mathrm{MAP}}^{(s-1)}(i', l) \tag{5.23}$$

where $\bigvee$ denotes the binary OR operator.

Cascade training Since for each scale the feature map heavily depends on current solution, we need to anticipate the impact of the pruning of previous coarse scales. This is a major difference from the training method proposed in [35]. To this end we propose the following learning framework:

Classifier training At each scale we need to train a classifier $z^{(s)}$ from a feature map $f^{(s)}$ and a pruning ground truth $X_{\mathrm{MAP}}^{(s)}$. The pruning ground truth $X_{\mathrm{MAP}}^{(s)}$ defines the class c_0 from c_1, where c_0 corresponds to the 0 values in $X_{\mathrm{MAP}}^{(s)}$,

[1] The set of nodes that are reachable by a path starting from i and traversing at most k edges

Algorithm 20: Cascade learning of pruning classifier framework

Data: Set of training models $(\mathcal{M}_m)_{0 \leqslant m < M}$, grouping functions $(g_m^{(s)})_{0 \leqslant s < N}$, pruning ground truth $(X_{\text{MAP},m}^{(s)})_{0 < s \leqslant N}$
Result: $(z^s)_{0 \leqslant s < N}$
for $s = [N...0]$ **do**
  for $m = [0...M]$ **do**
    Update $x_m^{(s)}$ by MAP inference:
    $x_m^{(s)} \approx \arg\min_{x_m \in \mathcal{S}(\mathcal{M}_m^{(s)}, A_m^{(s)})} E(x_m | \mathcal{M}_m^{(s)})$
    if $s \neq 0$ **then**
      Compute the feature MAP $f_m^{(s)}$ from current solution $x_m^{(s)}$
  if $s \neq 0$ **then**
    Train pruning classifier z^s from $(f_m^{(s)})_m$ and $(X_{\text{MAP},m}^{(s)})_m$
    for $m = [0...M]$ **do**
      Update pruning matrix $A_m^{(s)}(i,l) = z_{post}^{(s)}(f_{post,m}^{(s)}(i,l))$
      Upsample to next finest pruning matrix:
      $A_m^{(s-1)} \leftarrow [g_m^{(s-1)}]^{-1}(A_m^{(s)})$
      Upsample $x_m^{(s)}$ to initialize solution $x_m^{(s-1)}$ of next scale:
      $x_m^{(s-1)} \leftarrow [g_m^{(s-1)}]^{-1}(x_m^{(s)})$

i.e., the label can be pruned, while c_1 corresponds to the 1 values in $X_{\text{MAP}}^{(s)}$, i.e., the labels that should remain active since they are part of the ground truth estimated MAP at the finest scale.

We normalize all features of $f^{(s)}$ to the $[0, 1]$ interval to avoid numerical instability. We treat nodes sitting on the border of a strong discontinuity separately from the nodes laying in smooth regions. Indeed, a discontinuity at coarse scale is very likely to be refined during the next finest scales. Hence, there is more uncertainty on the solution in the vicinity of discontinuities than on smooth areas. Using the SOD feature, we split the feature map $f^{(s)}$ into a first group $f_0^{(s)}$ containing only features where $\text{SOD}^{(s)} \leqslant \rho_s$ (smooth area), and a second group $f_1^{(s)}$ containing only features where $\text{SOD}^{(s)} > \rho$ (strong discontinuity). We also split the ground truth $X_{\text{MAP}}^{(s)}$ the same way.

Since we need to compromise between good enough accuracy during training and fast evaluation at test time, we rely on linear classifiers for each group. To that matter, we train a standard linear C-SVM, Support Vector Machine, classifier with l_2-norm regularization. We refer the reader to [37] for details about SVM. As we have many more samples in class c_0, we randomly trim c_0 such that its cardinal, $\text{card}(c_0)$, reaches a 10 to 1 ratio with respect to $\text{card}(c_1)$, the cardinal of c_1 ($\text{card}(\cdot)$ counts the number of samples in each class). This greatly speeds-up the training without compromising its quality. Nevertheless, we still

need to balance the classes c_0 and c_1 so that a misclassification in each class accounts for the same energy in the SVM object function. We also want to limit misclassification in c_1 since this kind of error prunes labels that are part of the MAP. Hence, we weigh c_0 to 1 and weigh c_1 to $\lambda\frac{\text{card}(c_0)}{\text{card}(c_1)}$ with $\lambda \in \mathbb{R}^+$. We name the parameter λ the pruning aggressiveness factor as it relates to proportion of labels that get pruned.

To determine the best value for the C_s parameter common to both SVM objective functions and the best value for the threshold ρ_s, we equally split the samples into a training set and a validation set. We introduce a quality factor that accounts for the percentage of samples properly classified:

$$QF(C_s, \rho_s) = \lambda\frac{\text{card}(c_0)}{\text{card}(c_1)} \sum_{(i,l)\in c_1} z^{(s)}(f^{(s)}(i,l)) + \sum_{(i,l)\in c_0} (1 - z^{(s)}(f^{(s)}(i,l))) \quad (5.24)$$

We perform a simple grid search over $C_s \in [0.01, 0.1, 1, 10, 100, 1000]$ and $\rho_s \in [0.0001, 0.001, 0.01, 0.1, 0.25, 0.5]$ and we retain the couple $[C_s, \rho_s]$ that maximizes the QF factor on the validation set.

During on-line testing, the classifier $z^{(s)}$ applies the linear classifier learned for group $f_0^{(s)}$ if $\text{SOD}^{(s)} \leqslant \rho$ or group $f_1^{(s)}$ if $\text{SOD}^{(s)} > \rho_s$ as summarized in 20.

Inference by Learning algorithm

Finally, Inference by Learning algorithm built on the multi-scale approach to speed-up the MAP inference where we iteratively reduce the solution space by progressively estimating the MAP solution. We proceed by:

(i) Building a coarse to fine representation of the input model.

(ii) At each scale, we alternate between:

- Refining the MAP solution with the current scale model.
- Pruning the solution space by cleverly leveraging information of the current scale MAP estimation.

The pseudocode of the resulting algorithm appears in Algorithm 21.

5.3.5 Experiments

For this set of experiments we first evaluate and compare the Image and Energy pyramid schemes. Then, we evaluate the Inference by Learning method. Again, we make use of the stereo matching application of last chapter. As a baseline, we use the solution obtained with the α-expansion optimization technique. For both scheme, we monitor the energy ratio with respect to the energy obtained with α-expansion. We also compute the mean error using ℓ_1 norm with respect to the solution of the Graph-Cuts method.

Algorithm 21: Inference by learning framework

Data: Model $\mathcal{M}$, grouping functions $(g^{(s)})_{0 \leqslant s < N}$, classifiers $(z^{(s)})_{0 < s \leqslant N}$
Result: $x^{(0)}$
Compute the coarse to fine sequence of MRFs:
$\mathcal{M}^{(0)} \leftarrow \mathcal{M}$
for $s = [0 \dots N-1]$ **do**
  $\mathcal{M}^{(s+1)} \leftarrow g^{(s)}(\mathcal{M}^{(s)})$
Optimize the coarse to fine sequence of MRFs over pruned solution spaces:
Initialize $x^{(N)}$ and $A^{(N)} \equiv 1$
for $s = [N...0]$ **do**
  Update $x^{(s)}$ by MAP inference: $x^{(s)} \approx \arg\min_{x \in \mathcal{S}(\mathcal{M}^{(s)}, A^{(s)})} E(x|\mathcal{M}^{(s)})$
  if $s \neq 0$ **then**
    Compute feature map $f^{(s)}$
    Label pruning: update current pruning matrix
    $A^{(s)}(i,l) = z^{(s)}(f^{(s)}(i,l))$
    Upsample to next finest pruning matrix: $A^{(s-1)} \leftarrow [g^{(s-1)}]^{-1}(A^{(s)})$
    Upsample $x^{(s)}$ to initialize solution $x^{(s-1)}$ of next scale:
    $x^{(s-1)} \leftarrow [g^{(s-1)}]^{-1}(x^{(s)})$

Image and Energy pyramids

Image pyramid For the Image pyramid, we vary the downsampling factor from 0.5 to 0.8. Furthermore, at each scale, we set the label range to be within 5 pixels of the current solution.

We present the obtained results in tables 5.9, 5.10 and 5.11. For all experiments the Image pyramid scheme for Graph-Cuts delivers performances very close to the baseline and significantly outperforms the Image pyramid scheme for Primal dual techniques. This indicates that the non-convexity of the function to optimize plays a central role.

	Small size problems				
	Church	Factory	Buildings	Industry	Average
Coarsening 0.5					
Energy (ratio)	1.00	1.01	1.00	1.00	1.00
Mean Error	0.02	0.02	0.01	0.01	0.02
Coarsening 0.6					
Energy (ratio)	1.00	1.00	1.00	1.00	1.00
Mean Error	0.02	0.01	0.01	0.01	0.02
Coarsening 0.7					
Energy (ratio)	1.00	1.00	1.00	1.00	1.00
Mean Error	0.02	0.01	0.01	0.02	0.02
Coarsening 0.8					
Energy (ratio)	1.00	1.00	1.00	1.00	1.00
Mean Error	0.02	0.01	0.02	0.01	0.02

Table 5.9 – Image pyramid scheme for small size problems.

	Medium size problems				
	Church	Factory	Buildings	Industry	Average
Coarsening 0.5					
Energy (ratio)	1.02	1.01	1.01	1.01	1.01
Mean Error	0.09	0.06	0.10	0.08	0.08
Coarsening 0.6					
Energy (ratio)	1.01	1.01	1.01	1.00	1.01
Mean Error	0.10	0.06	0.08	0.08	0.08
Coarsening 0.7					
Energy (ratio)	1.01	1.01	1.01	1.00	1.01
Mean Error	0.06	0.06	0.10	0.08	0.07
Coarsening 0.8					
Energy (ratio)	1.01	1.01	1.01	1.00	1.01
Mean Error	0.06	0.05	0.08	0.09	0.07

Table 5.10 – Image pyramid scheme for medium size problems.

	Large size problems				
	Church	Factory	Buildings	Industry	Average
Coarsening 0.5					
Energy (ratio)	1.02	1.01	1.02	1.01	1.01
Mean Error	0.21	0.12	0.19	0.25	0.19
Coarsening 0.6					
Energy (ratio)	1.02	1.01	1.01	1.01	1.01
Mean Error	0.25	0.11	0.16	0.20	0.18
Coarsening 0.7					
Energy (ratio)	1.01	1.01	1.01	1.01	1.01
Mean Error	0.18	0.10	0.16	0.22	0.17
Coarsening 0.8					
Energy (ratio)	1.01	1.01	1.01	1.00	1.01
Mean Error	0.17	0.10	0.16	0.19	0.16

Table 5.11 – Image pyramid scheme for large size problems.

Energy pyramid For the Energy pyramid, we evaluate both the node and label coarsening. For node coarsening, we investigate different geometric grouping functions g_n that groups the nodes in a $[k \times k]$ fashion, with k being a positive integer in $\{1, 2, 3, 4\}$. For the label coarsening, we set the grouping function g_l to group the labels of $\mathcal{L}$ by set of m labels with m being a positive integer among $\{1, 2, 3, 4\}$. We select one every m labels to form the set of coarse labels $\mathcal{L}'$. Unselected labels are grouped with the closest selected label. If a tie appears for an unselected label, we arbitrarily associate it to the closest selected label of smaller index. As for the image pyramid, at each scale, we only activate the labels that are within a 5 label range of the current labeling solution.

Due to the number of experiments we only report the average results over the four subset *Church*, *Factory*, *Buildings* and *Industry*.

Small size problems

	Label coarsening			
Node coarsening	1	2	3	4
1				
Energy (ratio)	NA	1.01	1.00	1.00
Mean Error	NA	0.01	0.01	0.01
2				
Energy (ratio)	1.01	1.01	1.00	1.00
Mean Error	0.03	0.01	0.01	0.01
3				
Energy (ratio)	1.01	1.01	1.00	1.00
Mean Error	0.05	0.01	0.01	0.01
4				
Energy (ratio)	1.01	1.01	1.00	1.00
Mean Error	0.05	0.01	0.01	0.01

Table 5.12 – Energy pyramid scheme for small size problems.

The 5.12, 5.13 and 5.14 summarize all the experiments. In our setting, without any label coarsening the energy scheme is only exploring the immediate neighborhood of the current solution. Hence, if the initialization is good, we get acceptable results as demonstrated by small size experiments or for experiments with label coarsening set to 1 and node coarsening set to 2. However as we increasing the node coarsening to 3 or 4, we deteriorate the initialization during the node coarsening. As a result we get poor performance since the method might not explore the relevant label range.

When we combine both node and label coarsening, we obtain results close to be on par with the baseline and that significantly outperform the image pyramid scheme for both *alpha*-expansion and Primal-Dual optimization as pictured by figure 5.15. We explain this improvement due to the fact that the energy pyramid scheme represents the initial energy with better precision than the image pyramid approaches. This is illustrated by figure 5.14 where at coarse scales the solutions

Medium size problems

	Label coarsening			
Node coarsening	1	2	3	4
1				
Energy (ratio)	NA	1.00	1.00	1.00
Mean Error	NA	0.04	0.04	0.03
2				
Energy (ratio)	1.01	1.00	1.00	1.00
Mean Error	0.08	0.04	0.03	0.03
3				
Energy (ratio)	1.05	1.00	1.00	1.00
Mean Error	0.23	0.05	0.04	0.04
4				
Energy (ratio)	1.13	1.00	1.00	1.00
Mean Error	0.48	0.05	0.04	0.04

Table 5.13 – Energy pyramid scheme for medium size problems.

Large size problems

	Label coarsening			
Node coarsening	1	2	3	4
1				
Energy (ratio)	NA	1.00	1.00	1.00
Mean Error	NA	0.06	0.06	0.06
2				
Energy (ratio)	1.02	1.00	1.00	1.00
Mean Error	0.19	0.08	0.06	0.06
3				
Energy (ratio)	1.08	1.00	1.00	1.00
Mean Error	0.51	0.10	0.06	0.06
4				
Energy (ratio)	1.26	1.01	1.00	1.00
Mean Error	1.78	0.10	0.07	0.06

Table 5.14 – Energy pyramid scheme for large size problems.

obtained by the Energy pyramid scheme blatantly outperform the solution of the Image pyramid scheme.

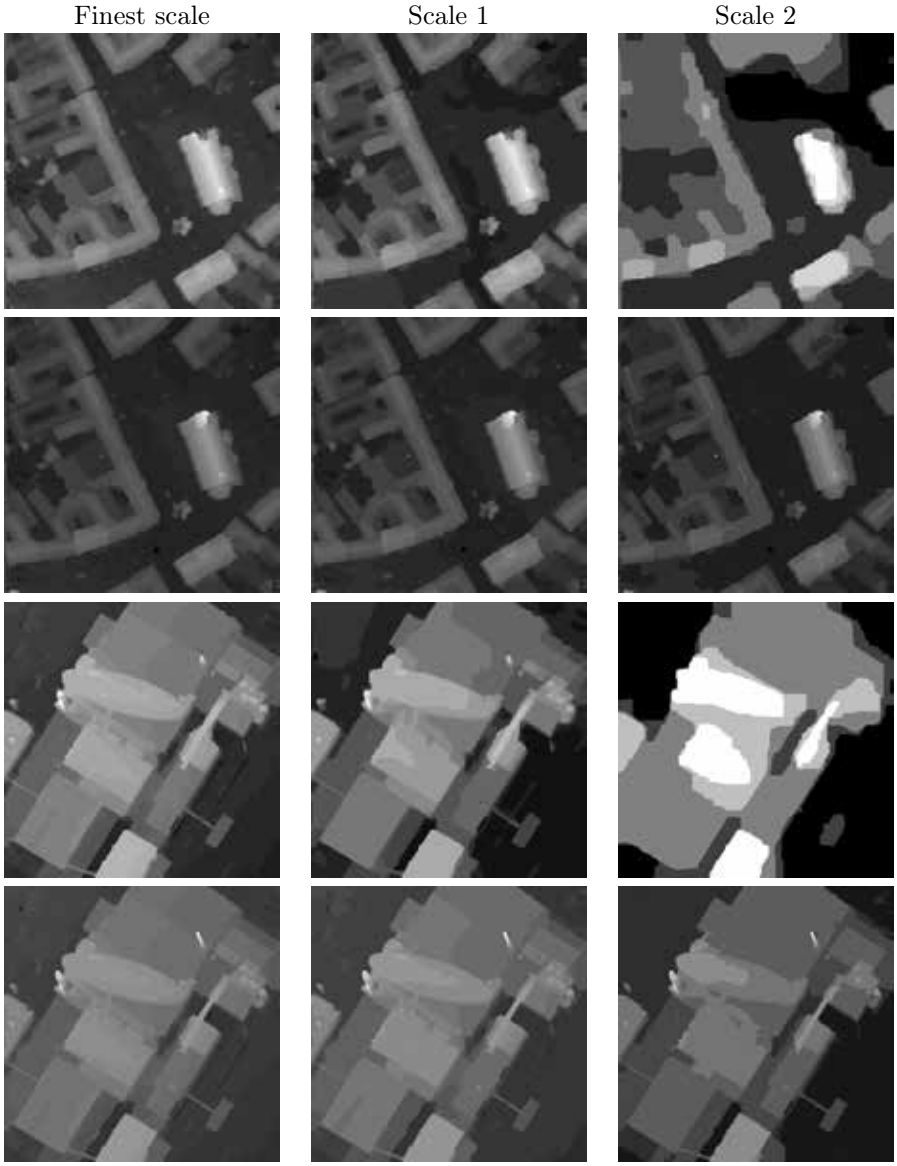


Figure 5.14 – Image coarsening results for odd rows and Energy coarsening for even rows. The energy coarsening scheme clearly outperforms the image coarsening scheme for coarse scales.

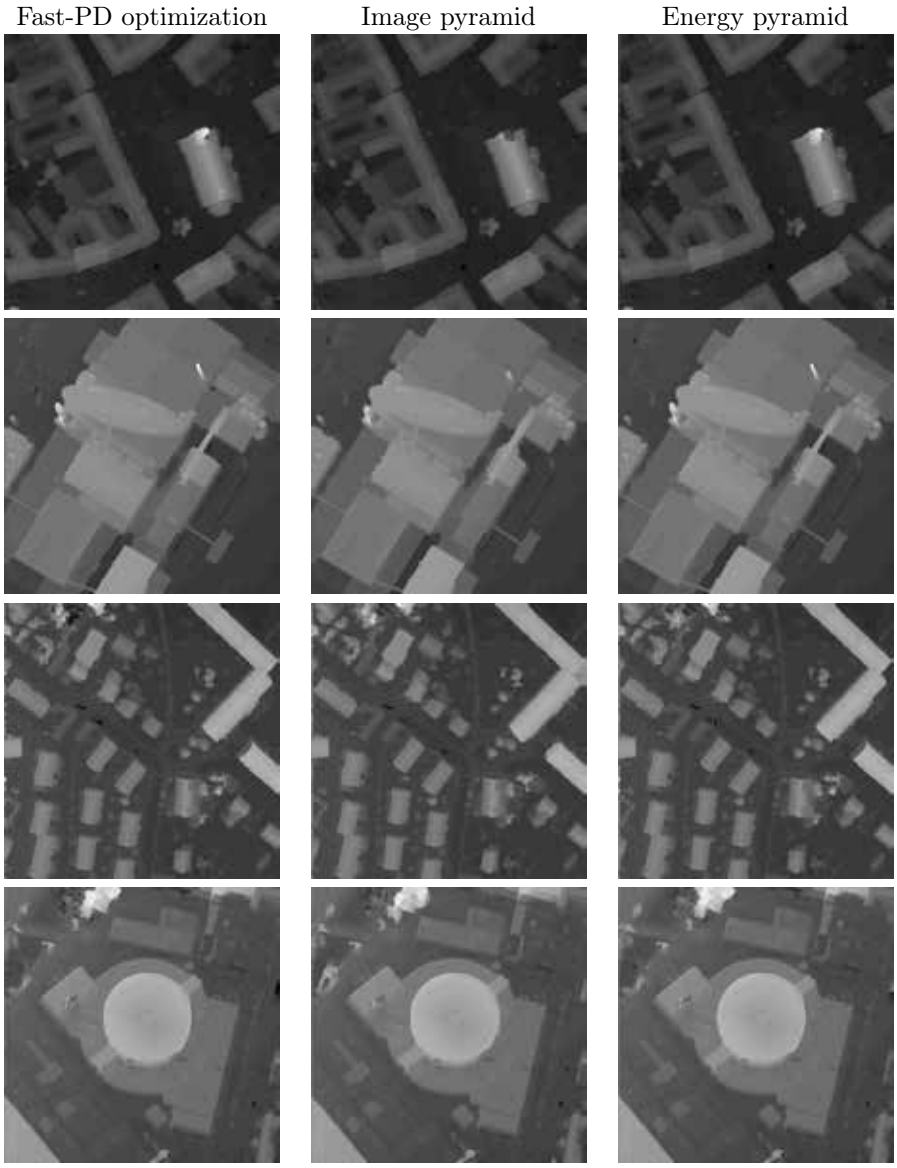


Figure 5.15 – Disparity maps for the baseline, the image and energy pyramid schemes. We see that the Image pyramid scheme has difficulty to retrieve small details like the bell tower of the church or the chimney of the factory. On the other hand, the Energy pyramid produces fine detailed disparity maps.

Inference by learning

For the evaluation of the inference by learning framework we make use of exactly the same coarsening functions as for the energy coarsening scheme. Hence, we group the nodes in a $[k \times k]$ fashion, with k being a positive integer in $\{1, 2, 3, 4\}$. We also group m labels together with m being an positive integer among $\{1, 2, 3, 4\}$.

During our experiments, we compare the baseline inference (optimizing the input MRF with Graph-Cuts), the multi-scale inference ($\lambda = 0$, i.e., this framework without any pruning), and our Inference by Learning method with different pruning aggressiveness factors λ that range between 0.001 and 1. We use 5 scales and we decrease at each scale the λ factor by an order of magnitude.

As for other experiments we use the energy ratio and the mean error to assess the performance of this framework. We also compute the *speed-up factor* that measure the ratio of computation time between the baseline optimization and the current optimization strategy. To verify the expected correlation between the speed-up and the number of active labels, we keep track of the *Active label ratio* that computes the percentage of active labels at the finest scale.

We report all results in figures 5.20, 5.21, 5.22 and 5.23. An analysis of these results reveals that by setting the node and label coarsening to 2 we achieve a best comprise between high quality solution and speed-up. Hence, we now focus only on this particular set of experiments.

For all experiments illustrated in figure 5.19, $\lambda = 0.1$ seems to be the sweet spot. This is also consistent for most of the other node and label coarsening values. For aggressiveness factors lower than $\lambda = 0.1$, the inference by learning scheme computes a lower energy solution than the baseline. Hence, we get a better solution for less computation. This happens because the approximation bounds of Fast-PD are reduced due the labels being pruned. In terms of speed, for $\lambda = 0.1$ we get consistent and large speed-up ranging from 6 to 12. For higher λ, the accelerations are even better. However, the quality of the solution deteriorates as pictured in figure 5.18.

As for the energy pyramid scheme, the inference by learning framework properly represents the energy even at coarse scales. The figure 5.17 illustrates the behavior. We see that the disparity maps estimated at coarse scales already contained the global structure. The details are then progressively resolved throughout the next fine scales.

As expected, the percentage of active labels strongly correlates with the speed-up factor. In fact, the speed-up is mainly caused by having a large number of labels being pruned. In figure 5.16, we see that a lot of labels are pruned early on and as predicted, less pruning happens near label discontinuities. This justifies the use of a dedicated linear classifier. Moreover, large homogeneously labeled regions are pruned earlier in the coarse to fine scale.

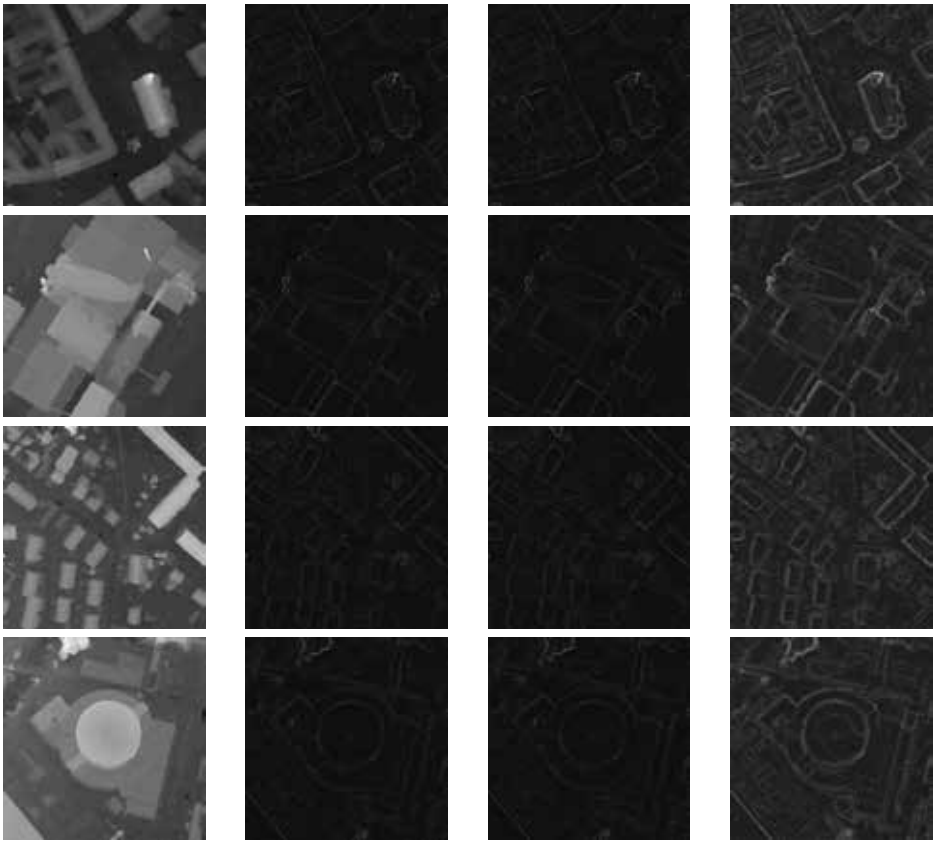


Figure 5.16 – Percentage of active labels per vertex (black 0%, white 100%) for Inference by Learning framework for $\lambda = 0.1$ with node and label coarsening both set to 2. Each row is a different subset. The framework maintains more active labels near discontinuities than in smooth ares. Many labels are pruned at coarse scale. At the finest scale, only a few active labels remain per node.

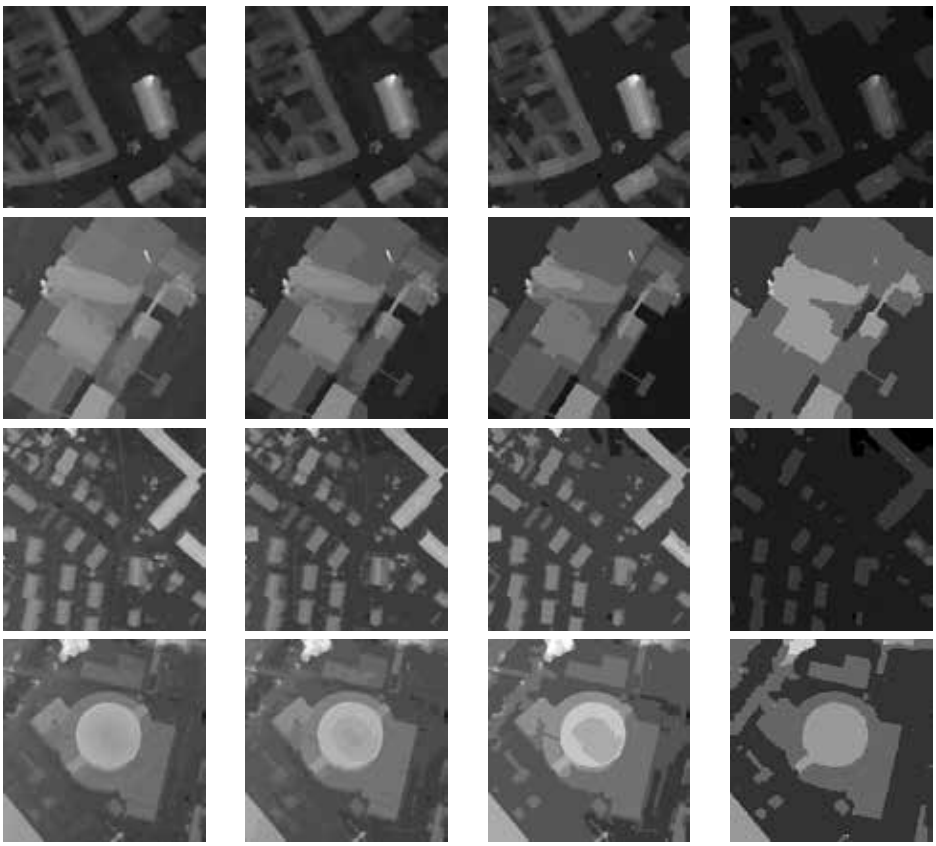


Figure 5.17 – Disparity maps produced at different scales by the Inference by Learning framework for $\lambda = 0.1$ with node and label coarsening both set to 2. The global structure of the disparity maps is estimated at coarse scales. The details of the disparity maps are then progressively resolved.

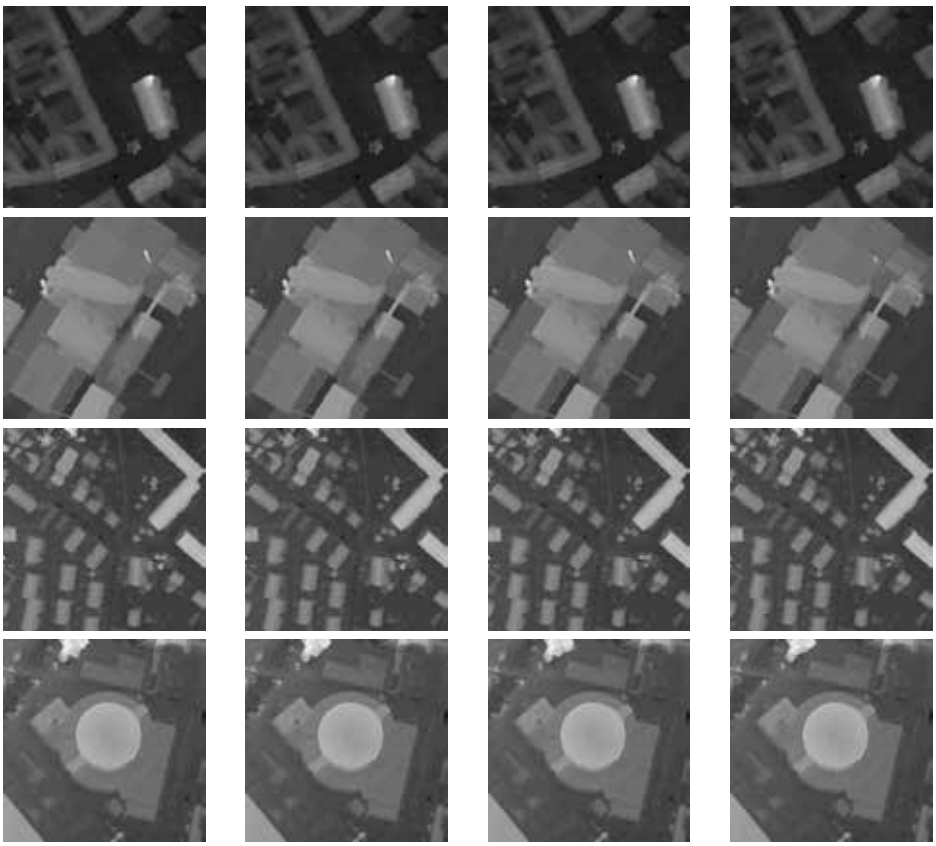
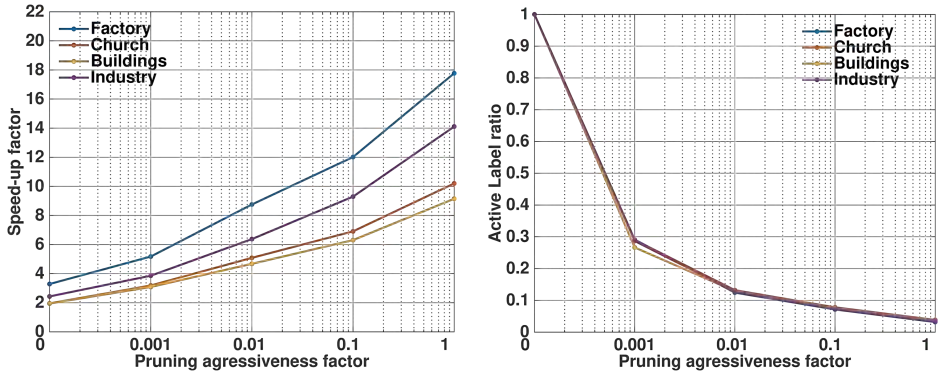
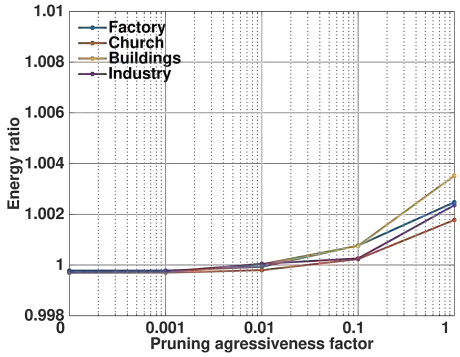


Figure 5.18 – Disparity maps produced by the Inference by Learning framework for different pruning factors λ with node and label coarsening both set to 2. Only the most aggressive pruning factor, $\lambda = 1$, is unable to resolve some fine details like the chimney of the factory subset.

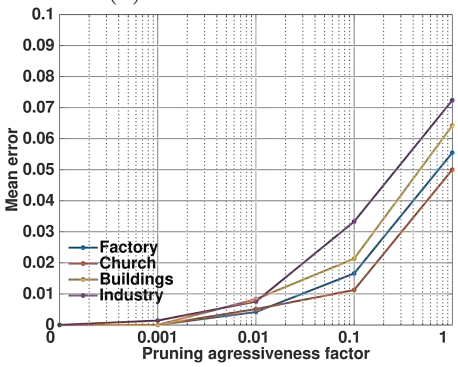


(a) Speed-up

(b) Active label ratio



(c) Energy ratio



(d) Label agreement

Figure 5.19 – Performance of the Inference by Learning framework for node and label corsening set to 2.

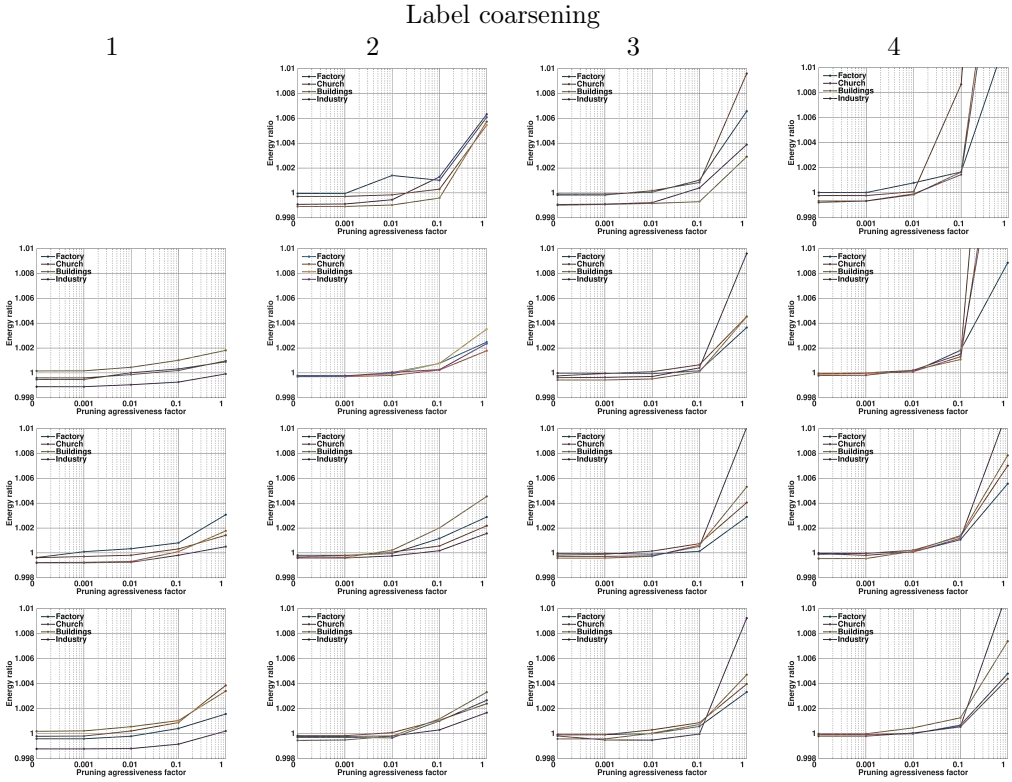


Figure 5.20 – Energy ratio results for the Inference by Learning experiments. From top row to bottom row node coarsening is set to 1, 2, 3 and 4.

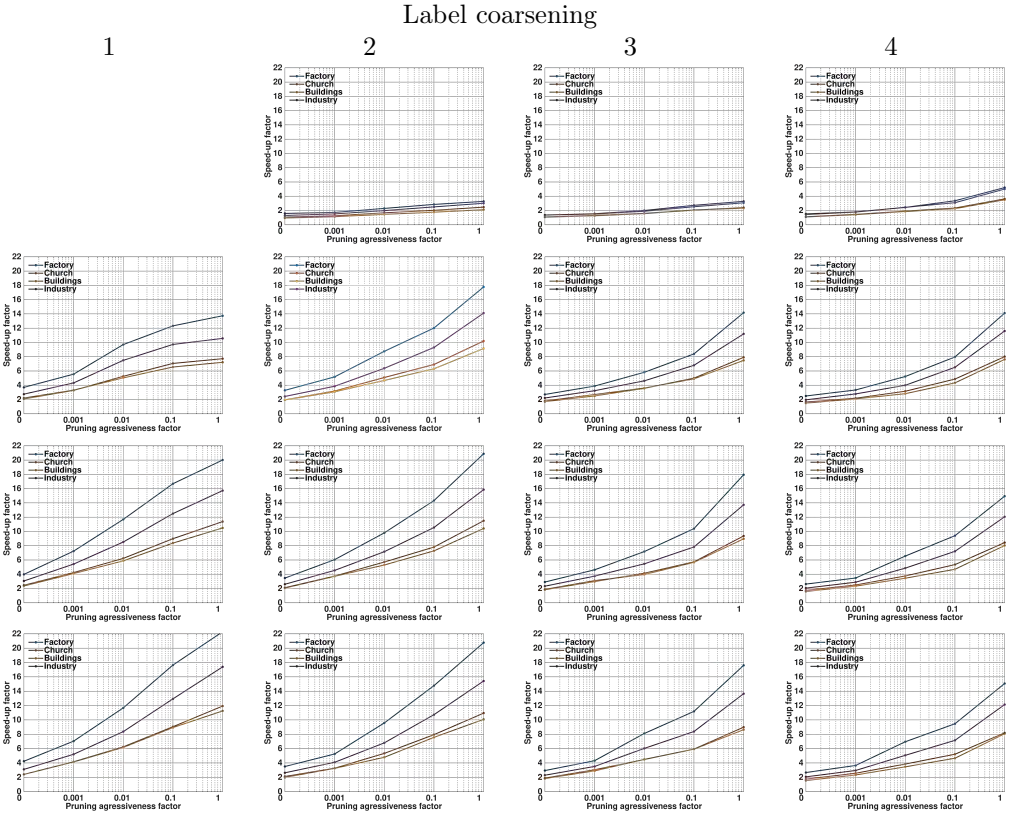


Figure 5.21 – Speed-up results for the Inference by Learning experiments. From top row to bottom row node coarsening is set to 1, 2, 3 and 4.

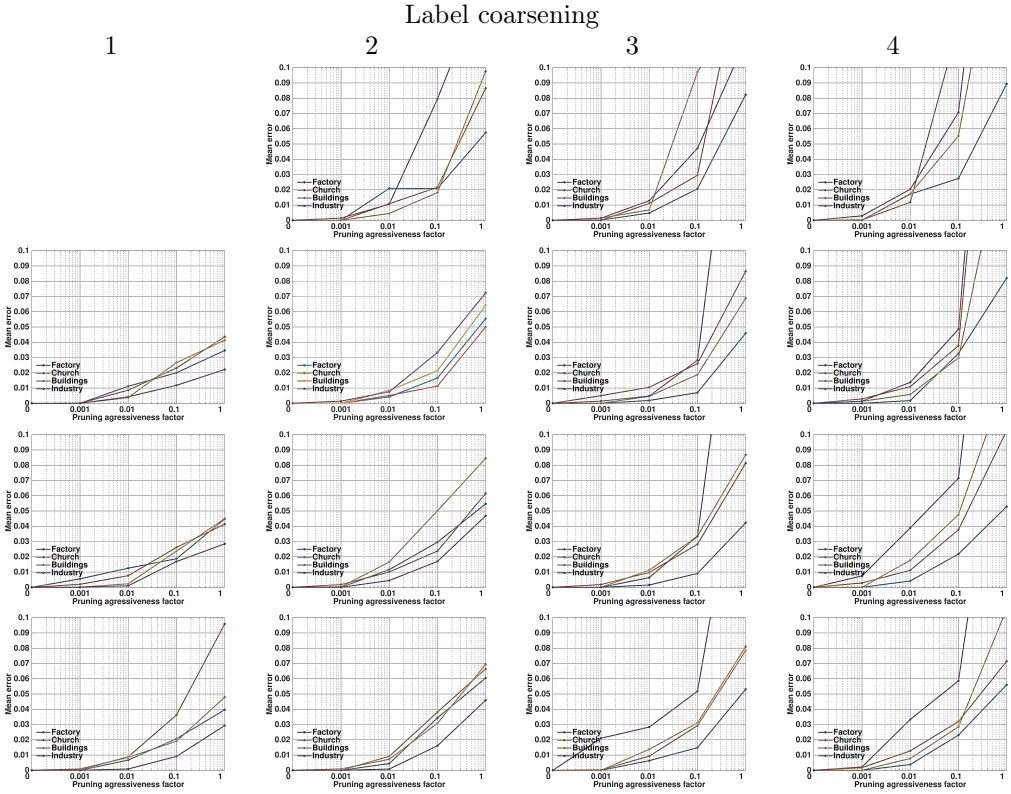


Figure 5.22 – Mean Error results for the Inference by Learning experiments. From top row to bottom row node coarsening is set to 1, 2, 3 and 4.

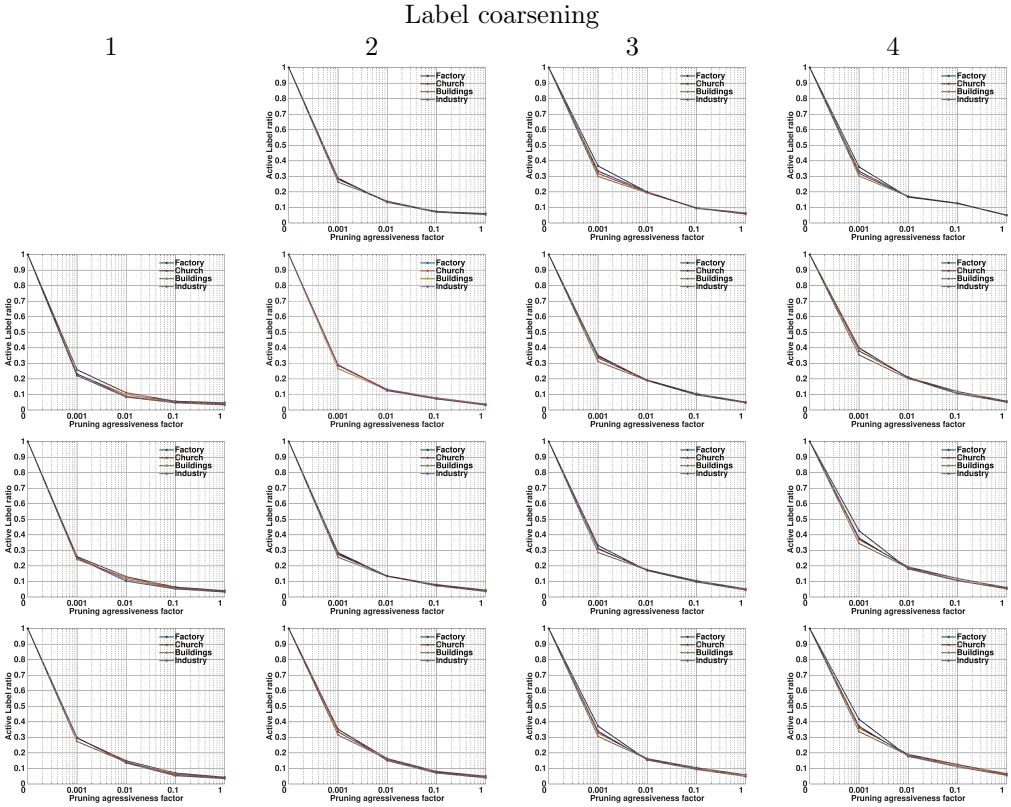


Figure 5.23 – Active Ratio results for the Inference by Learning experiments. From top row to bottom row node coarsening is set to 1, 2, 3 and 4.

5.4 Conclusion

This last technical chapter introduced smoothing and coarsening schemes for both First order Primal-Dual methods and non convex optimization techniques such as Fast-PD. For the latter one, we detailed a new framework, *Inference by Learning*, to drastically speed-up the optimization.

We now propose to apply some of the techniques presented to practical problems encountered in remote sensing tasks for Earth Sciences.

Chapter 6

Applications

6.1 Introduction and chapter organization

6.1.1 Introduction

For this final chapter, we display some applications of the techniques presented in chapters 3, 4 and 5. Our experiments mainly focus on remote sensing tasks performed in geological studies.

For each task, we introduce the context and the necessary background. We model the problem as an energy optimization task and then we perform experiments to illustrate the model's features. In any circumstances, we do not claim to get better results than other techniques. Indeed, we keep our models fairly generic and we leave for future work the derivation of a finely tuned version of our models.

6.1.2 Chapter organization

The section 6.2 introduces the notation and terminology used throughout the chapter. We present in section 6.3 the stereo-matching task with Earth and Mars bound acquisitions. The section 6.4 proposes a study of earth crust deformation from LiDAR acquisition with a simulated earthquake model. In the section 6.5 we apply our techniques to damage detection due to an earthquake from LiDAR acquisitions of Christchurch, New-Zealand.

6.2 Notations and Preliminaries

6.2.1 Images

We remind some useful notation for this final chapter. An image I is a collection of pixels taking value in $\mathbb{R}$ for gray images and $\mathbb{R}^3$ for color images. The pixels of I are organized on a rectangular grid Ω. A row and column (r_i, c_i) identifies

a unique pixel $p_i = (r_i, c_i) \in \Omega$. To ease the notations, we access the value of a pixel $p_i = (r_i, c_i)$ of an image I by either $I(p_i)$ or $I(r_i, c_i)$.

6.2.2 Graph

A neighborhood graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ is associated to the image I. The set of vertices $\mathcal{V}$ of $\mathcal{G}$ simply denotes the pixels of Ω. The set of edges $\mathcal{E}$ defines the canonical 4-neighborhood connectivity.

6.2.3 LiDAR as elevation maps

Light Detection and Ranging, LiDAR, is a class of instrument used in remote sensing to measure distances. A LiDAR is generally composed of a laser, a scanner, and a GPS receiver. Using light pulses in a form of a laser, the LiDAR measures the time to receive a reflection of the pulse in order to estimate a range of distance to an object of interest. This is illustrated by figure 6.1.

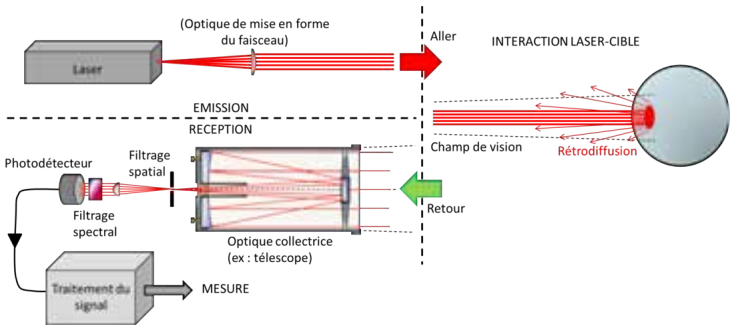


Figure 6.1 – Principle of LiDAR.

In our context LiDAR instruments are mainly used for topographic purposes [71]. They can be mounted on small aircraft during aerial surveys to estimate the topography. However, it is also common to mount LiDAR instruments on boat to estimate the bathymetry of the seafloor or a riverbed elevation [38] and [75]. In both settings, LiDAR are precious equipments that allow scientists and mapping professionals to build precise 3D model of their environment.

The LiDAR acquisitions create a 3D point cloud such as illustrated in figure 6.2. Since our acquisitions are mainly from zenith incidence with respect to the ground, we can transform the 3D point cloud into an elevation image. We set the elevation image grid to align with some portion of the ellipsoid of reference used to represent the Earth. Each pixel represents a square surface, for instance a square meter. The intensity of each pixel defines the elevation. Popular GIS, Geographic Information System, softwares like GRASS [132] have routines to convert a 3D point cloud to an elevation image. We note that the elevation image representation makes it very easy to apply the techniques presented in previous chapters.

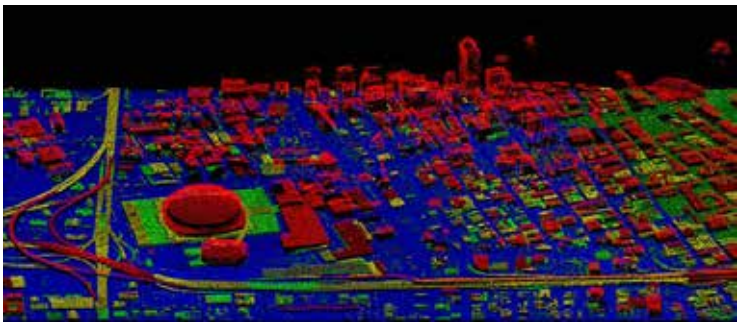


Figure 6.2 – LiDAR acquisition over New Orleans, LA, USA (Courtesy of USGS).

6.2.4 Matching criterion

We also remind two useful matching criteria based respectively on the ZNCC and the census coefficient. Both criteria compute a matching likeliness score between two patches P_1 and P_2 of the same size with spatial support $\mathcal{P}$.

Zero Normalized Cross Correlation: ZNCC

The ZNNC matching criterion derives from the Zero Normalized Cross Correlation coefficient that computes the angles between the normalized patches P_1 and P_2.

$$\text{ZNCC}(P_1, P_2) = \frac{1}{\text{card}(\mathcal{P})} \sum_{i \in \mathcal{P}} \frac{(P_1(i) - m_1)\,(P_2(i) - m_2)}{\sigma_1 \sigma_2} \tag{6.1}$$

where:

- m_1 and m_2 are the mean values of patches P_1 and P_2,
- σ_1 and σ_2 are the standard deviations of patches P_1 and P_2.

The ZNCC matching criterion ρ simply computes:

$$\rho(P_1, P_2) = 1 - \text{ZNCC}(P_1, P_2) \tag{6.2}$$

To simplify the notations, we define by

$$\text{ZNCC}_W(I_r, I_t, i, d) \tag{6.3}$$

the ZNCC coefficient computed on patches of size $W \times W$ from image I_r centered at pixel i and from image I_t centered at pixel $i + d$.

By construction the ZNCC coefficient and the ZNCC matching criterion are unaffected by global illumination and contrast changes. These invariance properties come extremely handy for image matching. We note that [172] describes an efficient implementation to compute the ZNCC coefficient in the context of overlapping patches.

Ternary Census

The Ternary Census matching criterion of [175] and relies on the Ternary Census code. The Ternary Census code of a patch P is defined by:

$$TC(P,\delta) = \otimes_{i\in\mathcal{P}} \;\; t(P(i_c), P(i), \delta) \tag{6.4}$$

Where:

- $\otimes$ is the concatenation operator,
- i_c is the central pixel of the patch P,
- $t : \mathbb{R}\times\mathbb{R}\times\mathbb{R}^+ \to \{-1, 0, 1\}$ is the ternary function parametrized by $\delta \in \mathbb{R}^+$:

$$t(x,y,\delta) = \begin{cases} -1 & \text{if} \quad x < y - \delta \\ 1 & \text{if} \quad x > y + \delta \\ 0 & \text{otherwise} \end{cases} \tag{6.5}$$

The Ternary Census matching criterion ρ simply computes:

$$\rho(P_1, P_2) = ||TC(P_1,\delta) - TC(P_2,\delta)||_1 \tag{6.6}$$

To simply the notations, we define by

$$\mathrm{TC}_{W,\delta}(I_r, I_t, i, d) \tag{6.7}$$

the Ternary Census coefficient computed on patches of size $W \times W$ from image I_r centered at pixel i and from image I_t centered at pixel $i + d$.

As for the ZNCC matching criterion, the Census matching criterion is unaffected by global illumination change. The Census matching criterion is also robust to mild contrast variation.

6.3 Stereo-matching

We briefly introduce the background of the stereo matching task. The figure 6.3 illustrates the key principle of stereo-matching: the apparent motions of an object between the two images is proportional to the object's depth. With additional information about the two cameras and their positions in space, one can recover the 3D position of each object imaged.

We explain in more detail how the problem is formulated. To this end, we first present the classic camera model and the epipolar geometry. Then, we formulate the stereo matching task as an optimization model which we evaluate with a series of experiments.

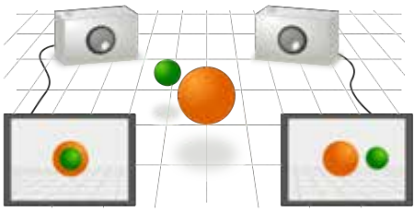


Figure 6.3 – Stereo-matching principle, *courtesy of Wikipedia*.

6.3.1 Camera models and Epipolar geometry

Pinhole camera model

We first need to present a camera model that describes the mathematical relationship between 2D coordinates $Q = (y_1, y_2)$ on the image plane and the 3D coordinates $P = (x_1, x_2, x_3)$ in the world. As for many modeling, we start with the pinhole camera model [160] which assumes that the camera aperture is a simple point with no optic as illustrated by figure 6.4. Hence, this model does not integrate neither geometrics distortions nor the discrete sampling done by the CCD sensor. However, since we process image from high quality camera we can compensate the geometrics distortions with simple coordinate transformations of the image coordinates. Using the notations of figure 6.4, one can easily demonstrate:

$$\begin{pmatrix} y_1 \\ y_2 \end{pmatrix} = -\frac{f}{x_3} \begin{pmatrix} x_1 \\ x_2 \end{pmatrix} \tag{6.8}$$

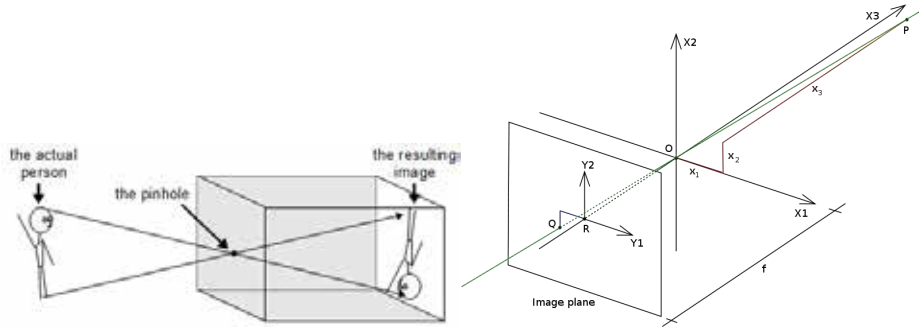


Figure 6.4 – Pinhole camera diagram

Camera parameters

To enhance the pinhole camera model, we need to introduce the intrinsic and extrinsic camera parameters.

Intrinsic parameters The intrinsic parameters describe features that are internal to the camera. These features are fixed for a given camera and digitalization setup. These parameters model:

- the focal length,
- the position of the optical center,
- the distortion introduced by the lenses,
- the size and shape of pixels.

Extrinsic parameters The extrinsic parameters define the location and orientation of the camera in the 3D world frame. Hence, these parameters include:

- a 3D translation to define the position,
- a 3D rotation to define the orientation.

Frame and Push-broom sensors

In our context we work with images acquired with frame and push-broom sensors. The frame sensors are the most common for aerial surveys while push-broom sensors are ubiquitous in satellite based acquisitions.

All pixels composing an image of a frame sensor are acquired at the same time by individual CCD (charge-coupled device) or CMOS (complementary metal-oxide semiconductor) sensors. In contrast, push-broom sensors only contain a single array of CCD or CMOS sensors. Using the motion of the satellite, one can construct an image by *stacking* the series of acquisition made by the array. The figure 6.5 illustrates the two technologies.

Epipolar geometry

Epipolar geometry refers to the particular geometry of the stereo-vision task. It describes the geometric relationship between a 3D point of the world frame and its projection onto 2D images acquired by two cameras in distinct positions. Both camera are assumed to be modeled by the pinhole camera model previously presented. While we do not present the detailed formulation of the epipolar geometry, we illustrate with figure 6.6 its concept. We refer to the seminal book of [76] for an extensive discussion around the epipolar geometry.

Interestingly for frame camera, in the epipolar geometry we know that a given object image at point X_L in the left camera lies on a line in the right camera named the epipolar line. The position on this epipolar line is proportional to the depth. However, for push-broom sensors the epipolar line is not straight, but hyperbola-like curve as explained in [135].

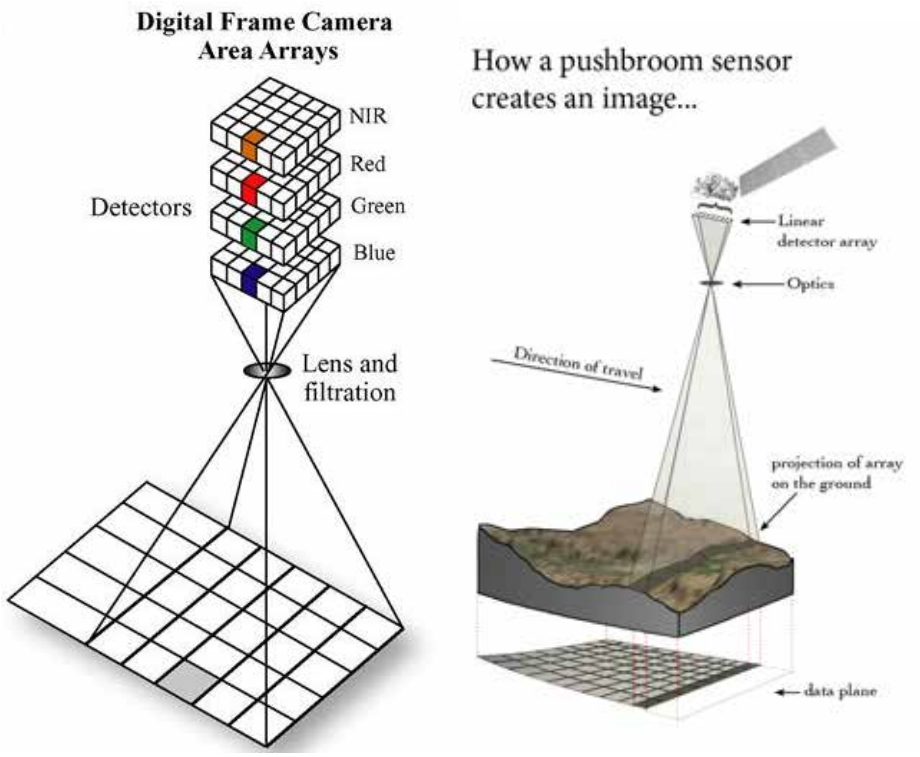


Figure 6.5 – Frame camera for a 4 band sensor and Push-broom camera for a panchromatic camera, *courtesy NASA*.

Image rectification

Finally, the rectification projects an image onto a reference image plane as illustrated by figure 6.7. With a judicious choice of the reference plane the rectification makes the epipolar lines horizontal for frame camera. This comes particularly handy for the stereo-matching task since one obtains a 1D registration problem. For push-broom, since the epipolar lines are curved one can make use of rectification but only to obtain locally horizontal line. We encourage the curious reader to study the work of [41] and [42] on this particular topic.

6.3.2 The stereo matching problem

We assume that we are given two rectified images, I_r and I_t. Let I_r be a reference image, Ω its spatial support and $G = (\mathcal{V}, \mathcal{E})$ its associated graph. The set of nodes $\mathcal{V}$ consists of the pixels of I_r and the set of edges $\mathcal{E}$ is defined by the 4 connectivity as illustrated in Fig. 6.8. Let I_t be the target image.

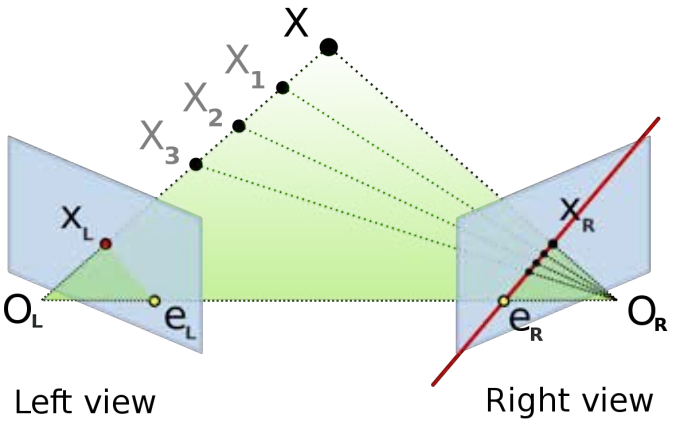


Figure 6.6 – Epipolar geometry for a frame sensor *courtesy of Wikipedia.*

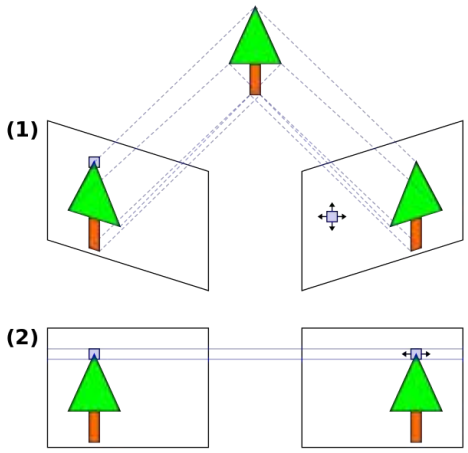


Figure 6.7 – Illustration of the image rectification transformation to obtain horizontal epipolar lines, *courtesy of Wikipedia.*

Probability formulation

Given I_r and I_t, we need to find the most probable 1D deformation d, that associates each pixel of I_r to a pixel of I_t with d being a function of $p \in \mathcal{V} \rightarrow d(p) \in \mathbb{R}$. Thus, d lives in $D = \mathbb{R}^{\Omega}$. We measure how a given d fits the data I_r and I_t by defining:

$$P(d|I_r, I_t). \tag{6.9}$$

The definition of P is context dependent, but most approaches enforce: (1) a

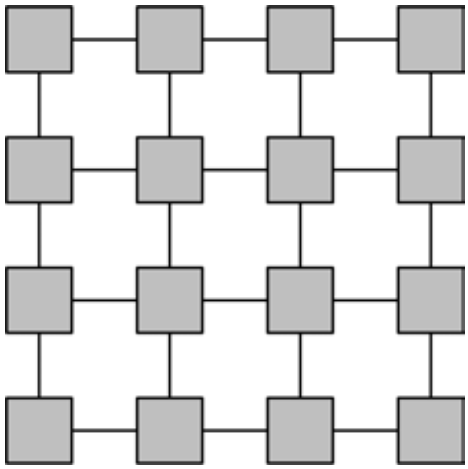


Figure 6.8 – Graph $G = (\mathcal{V}, \mathcal{E})$ of a 4 by 4 pixels image with a 4 connectivity.

notion of the similarity between I_r and $I_t \circ (id + d)$ (where id refers to the identity operator) by expressing $P_M(d|I_r, I_t)$ and; (2) a notion of regularity for d by expressing $P_R(d|I_r)$. If we suppose that these two probabilities are independent, we can write:

$$P(d|I_r, I_t) = P_M(d|I_r, I_t)P_R(d|I_r). \tag{6.10}$$

Instead of directly working with probabilities, we prefer using the energy domain as in [83] since it is easier to define measure on images. One can simply relate probability density function to energy through the Gibbs measure:

$$P(X = x) = \frac{1}{Z} \exp(-E(x)), \tag{6.11}$$

with Z being a normalization factor so that the integral of the probability function equal to 1.

Energy formulation

Through Eq. 6.11, we relate E, E_M, and E_R to P, P_M, and P_R respectively, which gives the following energy:

$$E(d) = \sum_{p \in \mathcal{V}} E_M(d, p) + \sum_{pq \in \mathcal{E}} E_R(d, p, q). \tag{6.12}$$

We define a pixel-wise similarity measure based on the similarity function ρ. Commonly used similarity functions are L1 or L2 norms [12], Census, Normalized Cross Correlation (NCC) or Zero Normalized Cross Correlation (ZNCC) [23], and the different versions of the Mutual Information [168, 91, 79]. In any case, the matching energy of a pixel p is defined as:

$$E_M(d, p) = \rho(I_r, I_t \circ (id + d))(p). \tag{6.13}$$

If the similarity measure is defined on a patch, we apply a rigid translation to the patch. Here, we use the ZNCC coefficient as it is robust to changes of illumination and contrast between I_r and I_t that appear due to specular objects or different acquisition times.

To enforce the regularity of d we choose to penalize the L1-norm of its discretized gradient, modulated by a weight function w. For each edge $pq \in \mathcal{E}$:

$$E_R(d, p, q)) = w(p, q)\|d(p) - d(q)\|_1, \tag{6.14}$$

ith :

$$w(p, q) = \lambda_1 + \lambda_2 \exp\left(-\frac{\|I_r(p) - I_r(q)\|^2}{\sigma^2}\right). \tag{6.15}$$

λ_1, λ_2, and σ are real positive scalar parameters. The L1-norm of the gradient naturally enforces piece-wise constant disparities. The weight function $w(p, q)$ relaxes the regularization on radiometric discontinuities of the reference image as in [58, 34]. This is an effective heuristic as most of the edges of the disparity maps are also edges of the image I_r.

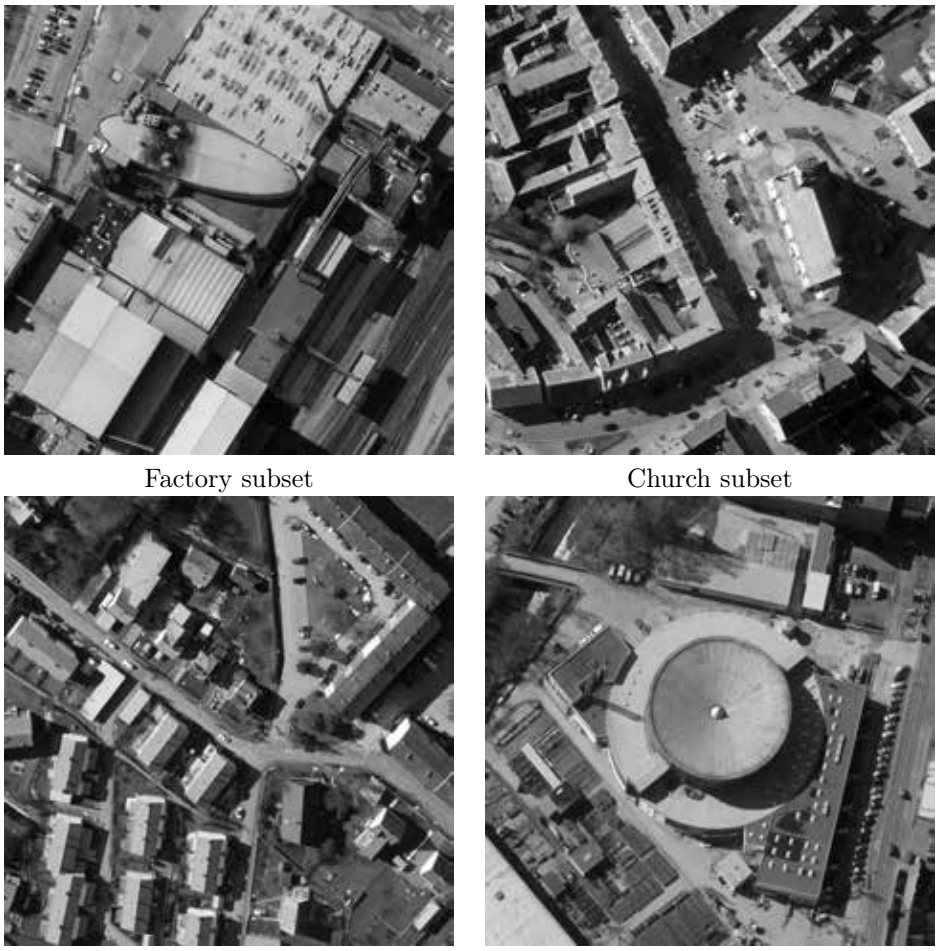
6.3.3 Experiments

We perform experiments with images acquired by the Ultra-cam and the Hirise cameras to illustrate the different components of our model. We start by displaying the images of the stereo-pairs. Then, we illustrate the unary terms issued from two different the matching criteria: the ZNCC and the Census measure. Finally, we study the influence of the regularization strength.

Images

Ultra-cam acquisitions The Ultra-cam cameras belong to the class of a frame sensor [109]. Hence, we can perform the calibration and rectification steps previously presented to end-up with a horizontal apparent motion to estimate.

From a large calibrated and rectified Ultra-cam stereo-pair we extract four subsets: *Factory*, *Church*, *Buildings* and *Industry*. The subsets, illustrated in figure 6.9, portray urban and industrial environments. Man-made structures present numerous challenges for the stereo-matching task. For instance, the tall chimney of the *Factory* subset creates a large occlusion zone. Some areas in the *Factory* and *Industry* subsets are texture-less or repetitive, making the matching very challenging. Finally, the *Church* and *Buildings* subsets have many fine-details due to their urban or suburban locations. This creates fine and sharp discontinuities in the disparity maps that are challenging to recover.



Factory subset | Church subset

Buildings subset | Industry subset

Figure 6.9 – Reference images of stereo pair subsets for the Ultra-cam experiments.

Hirise acquisitions The Hirise camera belongs to the class of a push-broom sensor [122]. Hence, we perform the calibration on the full image but the rectification step is applied onto each of the subsets. As for the Ultra-cam acquisitions we end-up with a near horizontal apparent motion to estimate.

Using two stereo-pairs from the Hirise website [134], we extract the four subsets pictured in figure 6.10: *Valley*, *Dunes*, *Crater* and *Channel*. Since the Hirise camera is mounted on a satellite orbiting the planet Mars, the subsets illustrate natural scenes of dunes, channels, valleys and carters. These natural scenes present different challenges compared to the urban environment of the UltraCam experiments. Indeed, most areas are heavily textured and non repetitive, which

greatly helps the matching. However, change in disparity is less likely to follow the radiometric variation of the reference image. This hinders the *a-priori* prior enforced by the regularization costs. Moreover, large change of elevation can be observed at the edges of craters, cliff, canyons and channels.

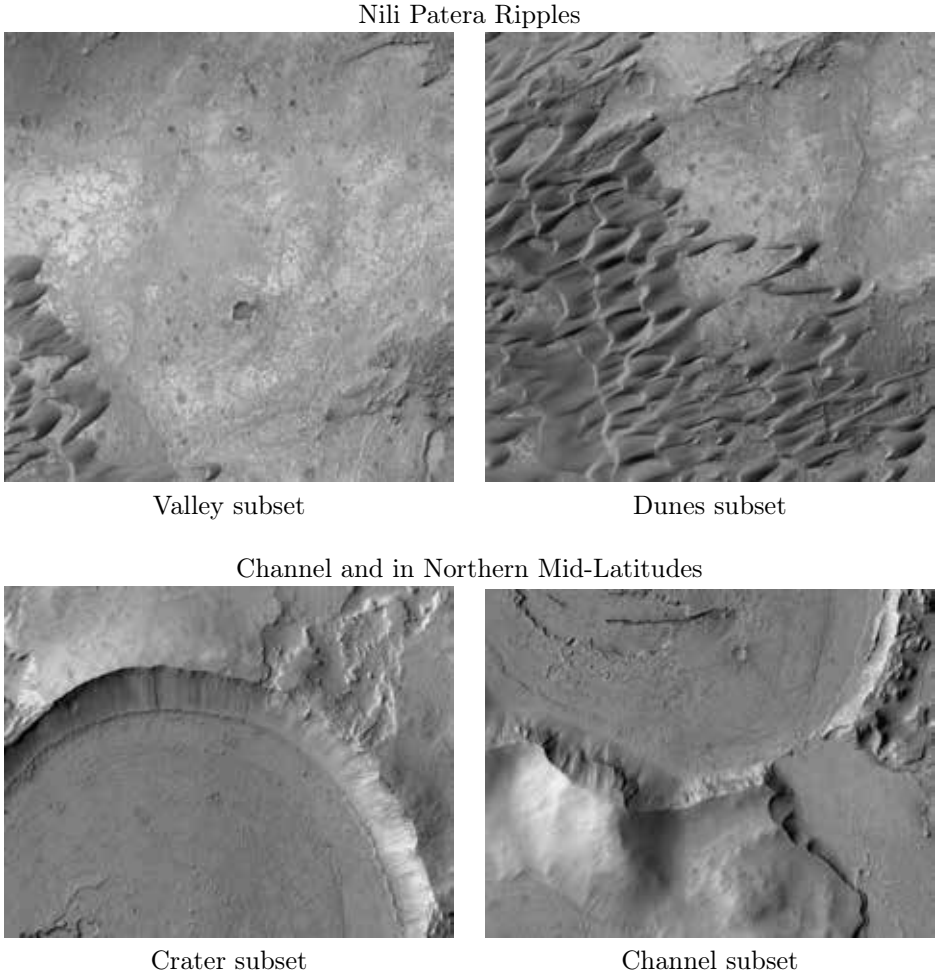


Figure 6.10 – Reference images of stereo pair subsets for the Hirise experiments.

Unary terms

In our context of stereo-matching, one can advocate that the unary terms are the most important part of the model. Indeed, their purpose is to measure the likelihood of one patch to match another. In a perfect world, we would like to rely only on the matching terms to estimate the disparity. Unfortunately, texture-less patches, noise, occlusions or change in illumination all hinder the

reliability of the matching terms. Hence, as an illustration we display in figure 6.11 and 6.12 the disparity maps obtained solely from the matching terms, i.e. we set all regularization terms to 0.

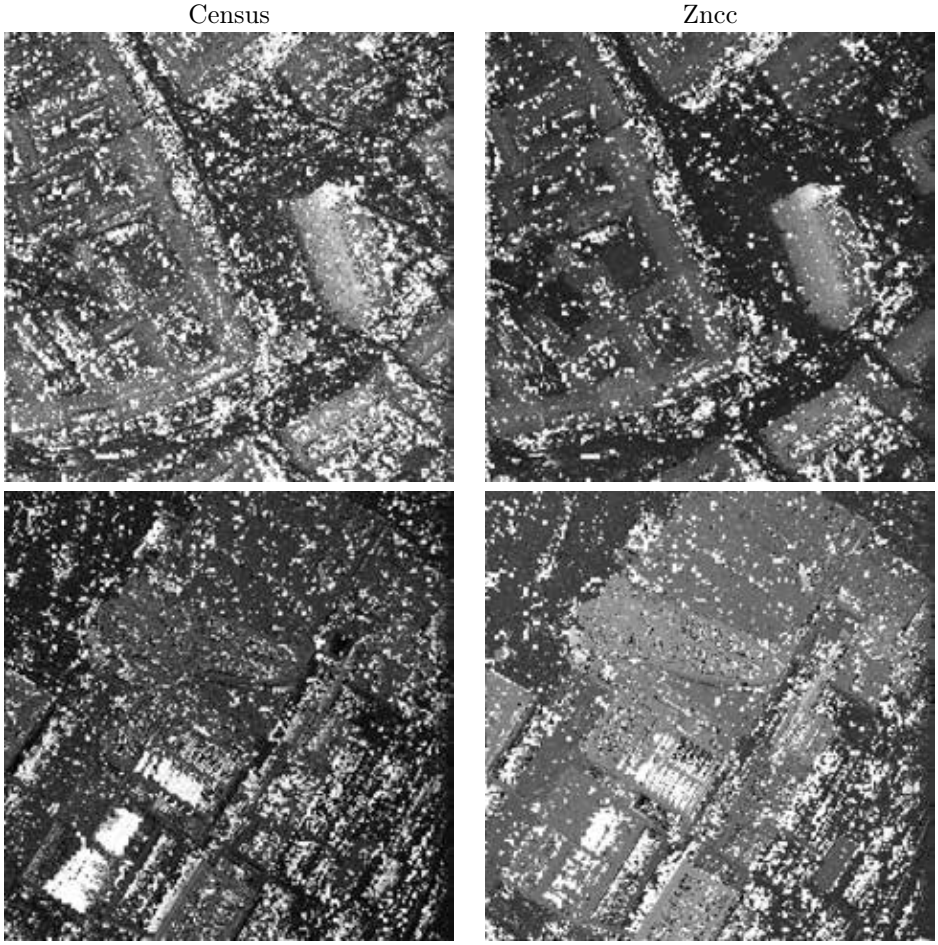


Figure 6.11 – Disparity maps obtained for the Ultra-cam stereo-pairs using exclusively the matching terms.

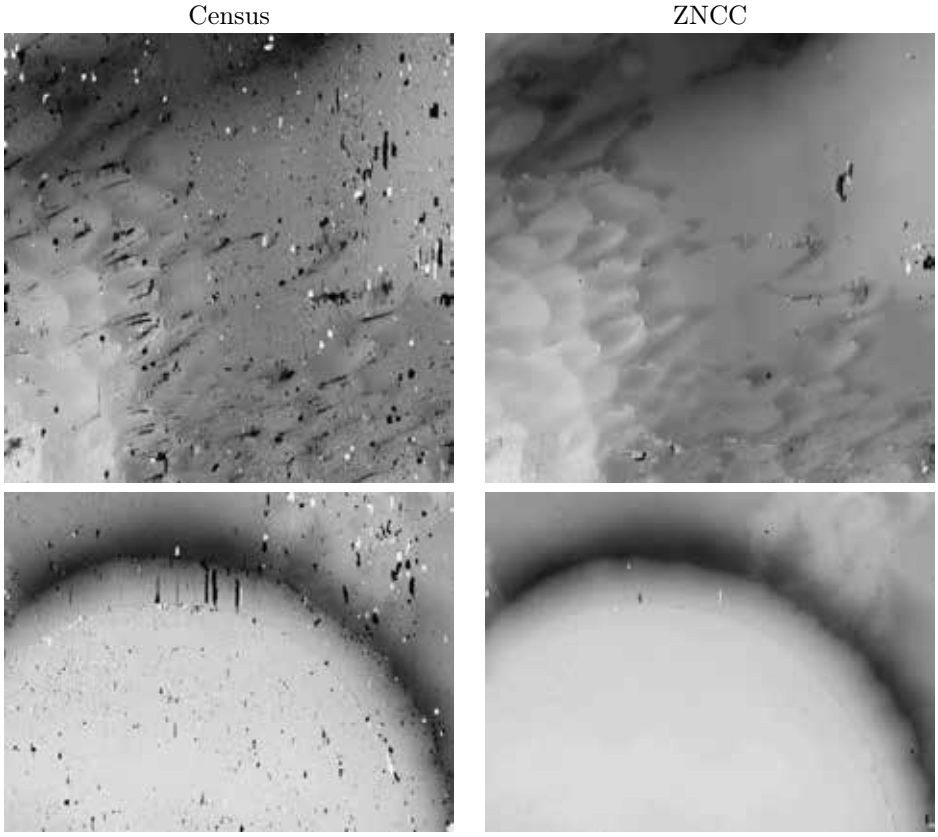


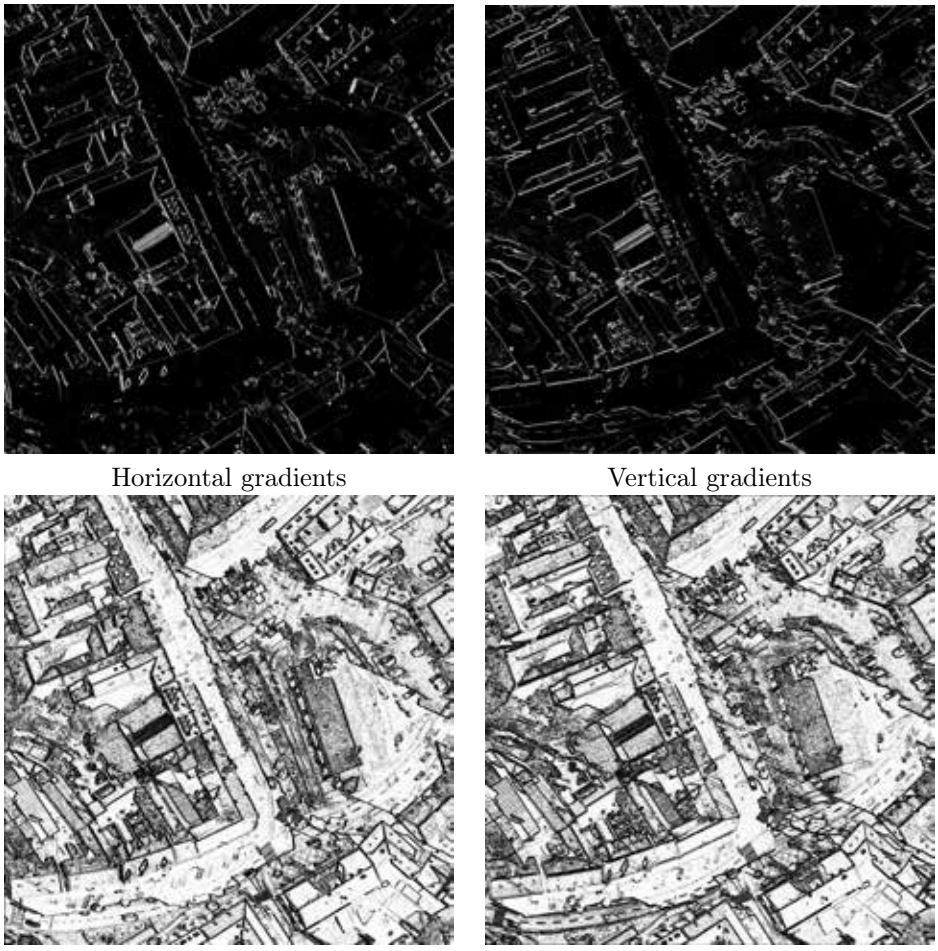
Figure 6.12 – Disparities maps obtained for the Hirise stereo-pairs using exclusively the matching terms.

Independently of the matching criterion, the disparity maps obtained for the Ultra-cam images are extremely noisy and entire areas are completely miss-estimated. The ZNCC criterion seems to perform a bit better than the Census. This can be attributed to the fact that the Census is more robust than the ZNCC. In our context, this extra-robustness seems to slightly deteriorate the matching performance. Interestingly, the disparity maps computed for the Hirise stereo-pairs are quite good. This is likely due to the heavily textured patches of natural scenes. However, we still observe numerous artifacts. As for the Ultra-cam results, the ZNCC outperforms the Census criterion.

Regularization weights

The regularization weights are an important part of the imposed prior on the disparity estimation. In our model they modulate the strength of the regularization according to the radiometric discontinuity of the reference image. We illustrate in figure 6.13 the horizontal and vertical gradients of the Church

subset's reference image and its derived regularization weights.



Horizontal gradients | Vertical gradients

Horizontal weights | Vertical weights

Figure 6.13 – Gradients and weights of the Church subset (brighter grays means higher values).

Impact of regularization

To study the regularization we modulate the strength of the pairwise terms by a global factor varying in $\{0.1, 0.5, 1, 2, 5\}$ where a factor of 1 sets the regularization of optimal hand-picked regularization parameters. This allows us to create models that are barely regularized (factor set to 0.1) to models that are heavily regularized (factor set to 5). We illustrate the results in figures 6.14, 6.15, 6.16 and 6.17.

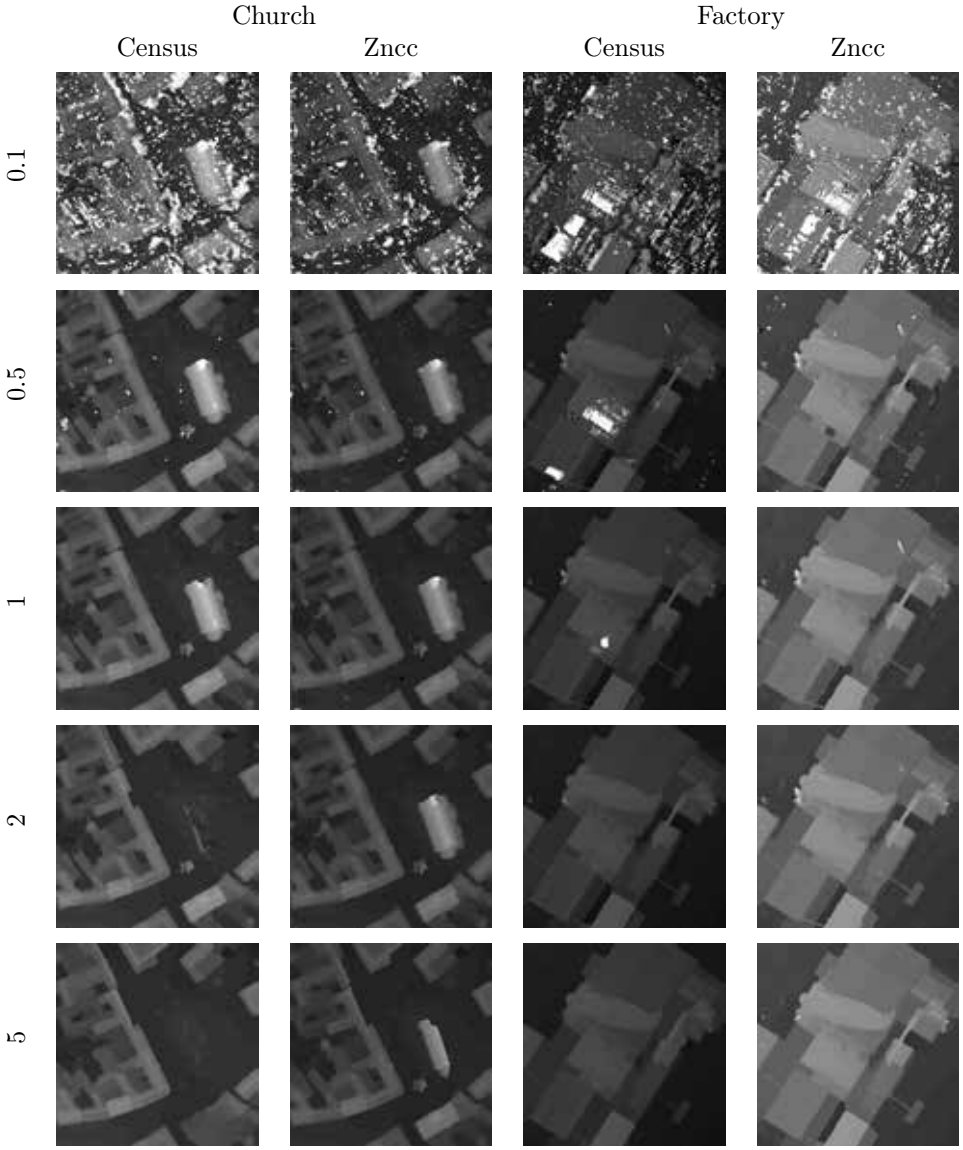


Figure 6.14 – Impact of the regularization for the Church and Factory subsets for the Ultra-Cam acquisition.

At low regularization strengths (factor set to 0.1 or 0.5), we still observe artifacts for the Ultra-cam images. The Census based experiments present more erroneous disparity estimations than their ZNCC counterparts. For larger regularization strengths (factor set to 2 or 5), and independently of the matching criterion, details such as the bell tower in the Church subset, the chimney in the

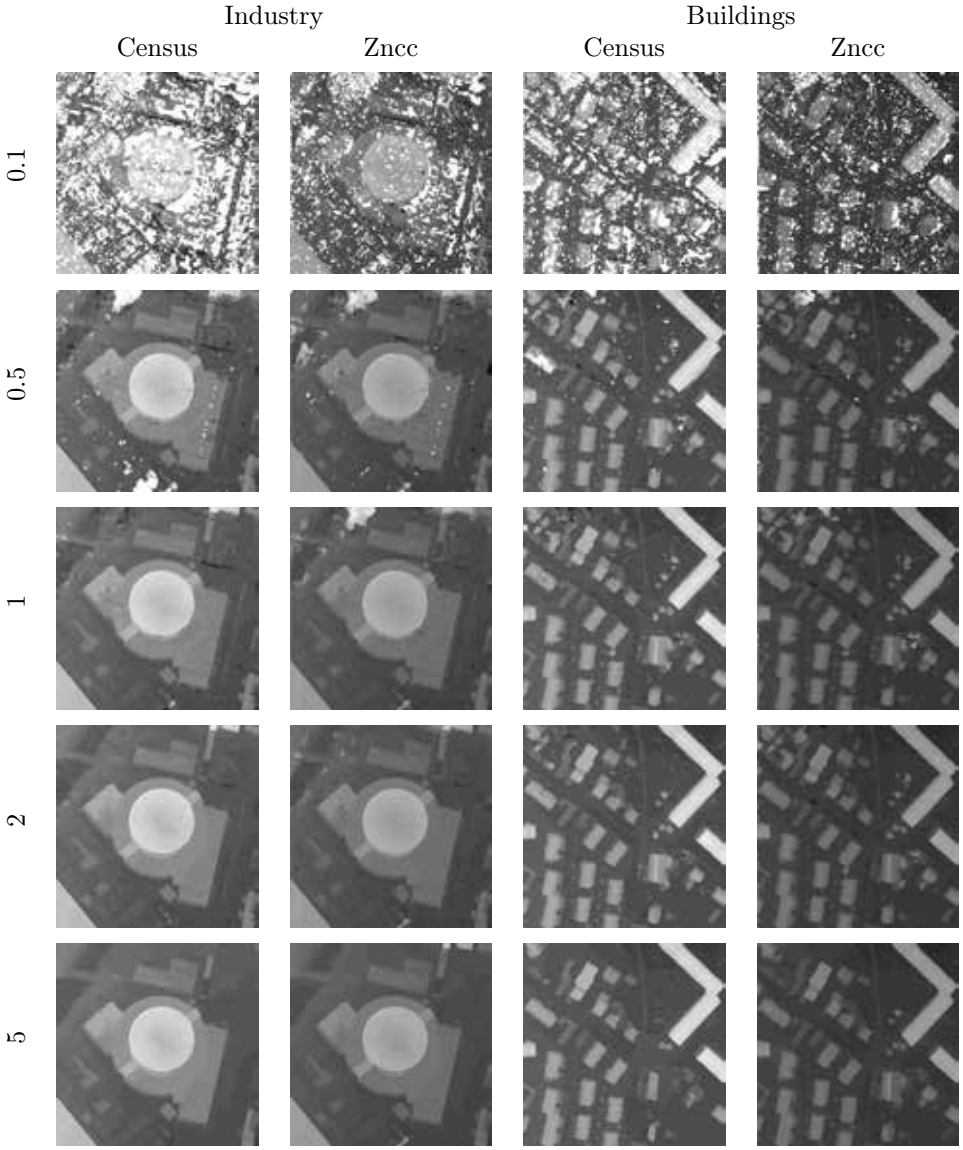


Figure 6.15 – Impact of the regularization for the Industry and Buildings subsets for the Ultra-Cam acquisition.

Factory subset and small structures of the Buildings subset start to disappear. The Census based experiment appears to be even more sensitive to the tuning of regularization since for a factor of 2 the church of the Church subset completely vanishes.

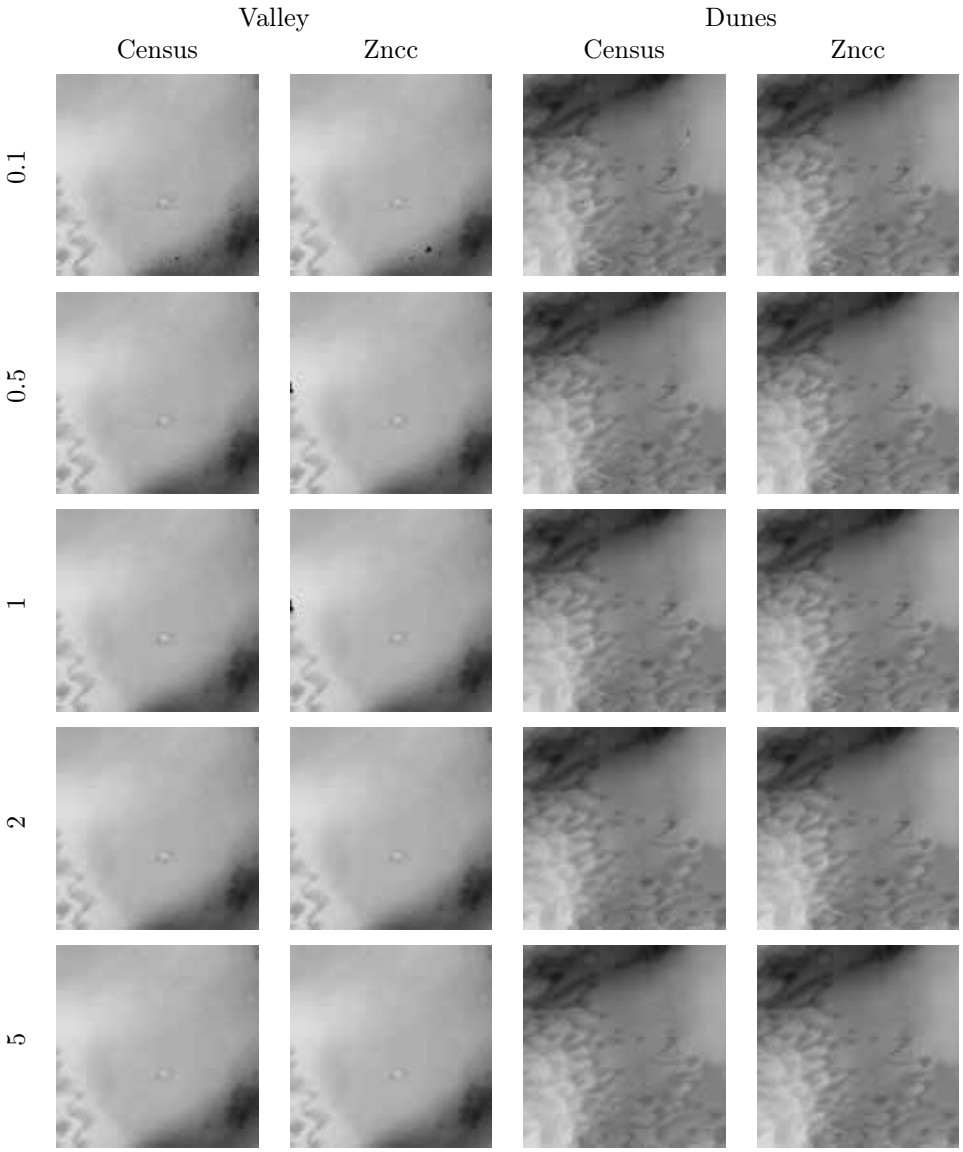


Figure 6.16 – Impact of the regularization for the Valley and Dunes subsets of the Hirise acquisition.

The results for the Hirise acquisitions exhibit the same tendency but to a lesser extent. We note that a regularization with a factor between 0.5 and 2 already gives good results for all subsets. This is because the matching terms become more discriminatory for images with lots of texture.

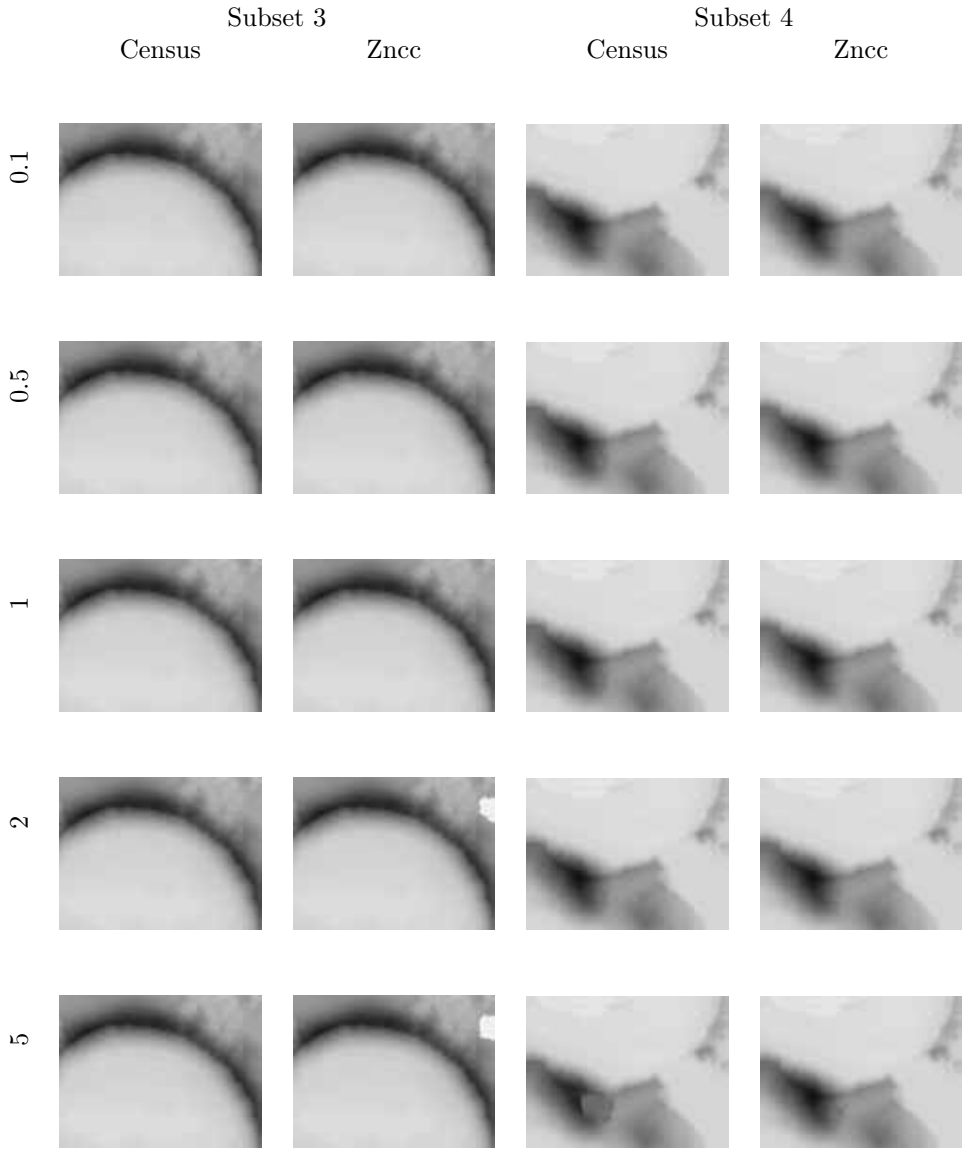


Figure 6.17 – Impact of the regularization for the Cater and Chanel subsets of the Hirise acquisition.

6.4 Simulated Earth crust deformation

We now move on to 3D registration of elevation maps. In this particular example we assume that an earthquake has occurred between the time of the acquisition of two elevation maps.

6.4.1 Context

Earthquakes are one of the most dangerous natural hazard on Earth. Contrary to volcano, hurricanes or tornado, it is as of today quite impossible to precisely predict when and where an earthquake will happen [60]. Earthquakes occur at a fault, i.e., a boundary between two or more tectonic plates, producing seismic waves that result from a sudden release of energy in the Earth's lithosphere. The figure 6.18 displays the epicenter of referenced earthquakes between 1963 to 1998. Earthquake are classified on a scale by their moment magnitude. The USGS maintains an active mapping of earthquakes thanks to an international network of seismometer [169].

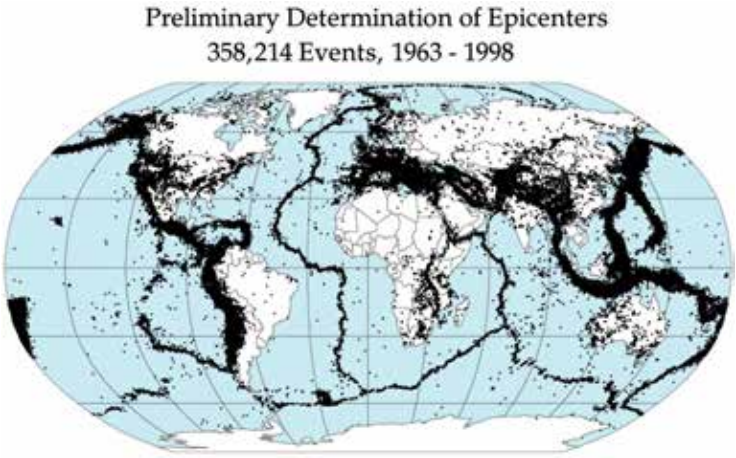


Figure 6.18 – Epicenter of earthquakes from 1963 to 1998.

The destruction and subsequent events caused by earthquakes are staggering: ground rupture and soil liquefaction, landslides and avalanches, fires, tsunami and flood. At multiple times in the past, a single quake and its aftermath have claimed more than ten of thousands lives as illustrated by figure 6.19.

Geologists have historically relied on seismometers to measure the motion of the ground. However, in the last decades they have started to heavily rely on GPS stations. The instantaneous position of civilian GPS is fairly inaccurate, within 10 meters. Nonetheless, using temporal aggregation of a GPS signal, one can retrieve precision up to a few millimeters. Some locations like the San Andreas fault in California, the Himalaya mountains or Japan are surveyed by networks of GPS stations. Unfortunately, many remote places do not benefit from this level of monitoring. Hence, for a large number of earthquake the scientific community does not have access to data acquired close to the epicenter. Another downside of the GPS monitoring is that even with large networks one can only obtain local and sparse measurements.

Hence, thanks to the progress of space imaging and aerial monitoring devices,

Figure 6.19 – Earthquakes of magnitude 8.0 and greater since 1900. The apparent 3D volumes of the bubbles are linearly proportional to their respective fatalities (Courtesy of Wikipedia).

geologists have started to take advantage of remote sensing acquisitions. In this context, they seek to obtain a more global view of the ground motion induced by an earthquake. Nevertheless, remote sensing techniques can not bring the level of accuracy obtained with GPS stations. Hence, it remains important to properly fuse the different modalities of information.

In this work, we only propose to retrieve the ground motion induced by a quake using two elevation maps: one acquired before and after the quake. To keep this experiment simple, we make use of a crop of an elevation map of the San Andreas fault in California, USA illustrated in figures 6.20 and 6.21. The crop serves as the pre-event elevation map. We simulate using an Okada model [137] the ground motion induced by a large earthquake. From this simulated ground motion we transform the pre-event elevation map to a post-event elevation map. We do not claim that the position of the simulated fault nor the generated earthquake are realistic. However, this experiment shows how to make use of our mathematical techniques in such context.

6.4.2 Model

We make use of notations I_{pre} and I_{post} for the pre and post geo-registered elevation images. Instead of directly looking for a full 3D deformation, we limit

Figure 6.20 – The San Andreas fault.

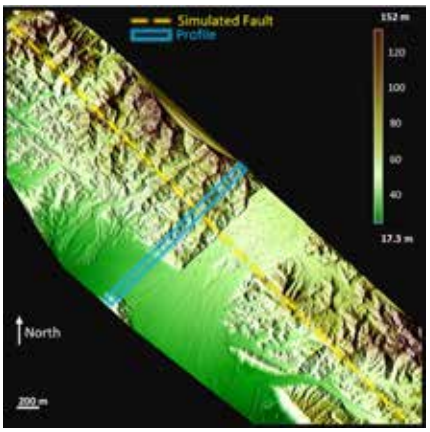


Figure 6.21 – Simulated fault.

our model to estimate a 2D motion and we automatically infer the change of elevation. This modeling reduces the number of unknowns and it performed better than others of our unreported attempts with full 3D motions. We elect to use the ZNCC matching criterion $\rho()$. The matching cost for pixel $(r_i, c_i) \in \Omega$ with displacement (u_i, v_i) is defined by:

$$\rho(r_i, c_i, v_i, u_i) \tag{6.16}$$

We formulate the registration problem as:

$$\begin{aligned} \underset{((u_i,v_i)\in\mathbb{R}\times\mathbb{R})_i}{\arg\min} & \sum_{(r_i,c_i)\in\Omega} \rho(r_i, c_i, v_i, u_i) \\ & + \sum_{(i,j)\in\mathcal{E}} \lambda^u |u_i - u_j| + \lambda^v |v_i - v_j| \end{aligned} \tag{6.17}$$

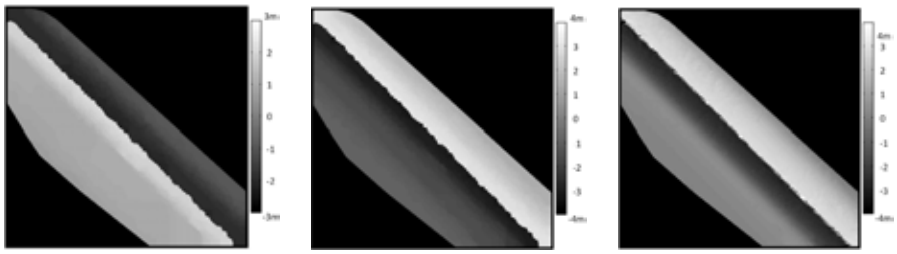
with λ^u and λ^v being real valued scalars.

We solve model 6.17 using the α-expansion technique embedded in a Image Pyramid. We elect for 4 scales with 49 labels per scale, i.e, 7 potential values for each u_i and v_i.

To recover the change of elevation w, we simply compute the change of elevation between the pre-event elevation map and the transformed post-event elevation map with the recovered deformation (u, v). We note that a further post processing smoothing step could also be performed if necessary.

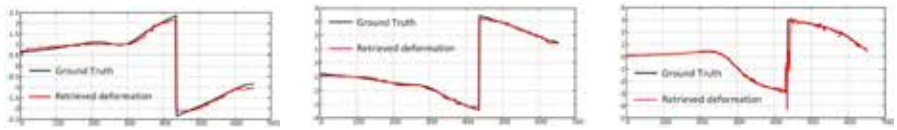
6.4.3 Experiments

We display in figure 6.22 the results of our method. In figure 6.23, we show a profile for each motion direction across the fault as illustrated in figure 6.21.



u-component v-component w-component

Figure 6.22 – Retrieved motion.



u-component v-component w-component

Figure 6.23 – Profile of the deformation in pixel across the simulated fault.

From the three profiles, we see that our method correctly recovers the ground motion for this toy example. We observe that the change of elevation w-component, is noisier. For real experiments, we would advocate to proceed with a smoothing post-processing step. One can for instance adapt the TV-L1 denoising framework to tailor an adequate smoothing algorithm.

6.5 Damage detection in New-Zealand

6.5.1 Context

For this final experiment, we study the application of damage detection induced by natural hazards from aerial LiDAR time series with one acquisition before and after the damaging event.

Natural disasters

In the context of natural disaster such as landslides, a tornadoes, hurricanes or earthquakes, a detailed cartography of the impacted area is of crucial importance.

In many cases, usual communication medium such as telephone or Internet are sometimes unavailable in some remote or less developed area or non-operational due damages induced by the natural disaster. Hence, gathering and centralizing information can be very challenging. First emergency responses face the challenge to adequately organize search and rescue operation while lacking a global view of the situation. Simple images from satellite and areal survey can provide crucial information of impacted areas.

Once the humanitarian situation has been resolved comes the time of reconstruction. In most modern countries, habitation and buildings are insured through various private or governmental based policies. Hence, one needs to provide a detail map of which building was affected by the natural hazard. This inspection is generally conducted on the ground. Depending on the size of the area impacted, this investigation can last quite a lot of time delaying families and business owners to obtain compensation from their insurance. Again, a detailed map of damages could fasten the economic recovery process.

Christchurch, New Zealand

Christchurch is located in the Canterbury Region in New Zealand's South Island as illustrated by figures 6.24 and 6.25. With a population of 389000 in 2016, Christchurch is one of the largest populated cities in New Zealand see 6.27 and 6.26.

Figure 6.24 – New Zealand on the planisphere.

Figure 6.25 – New Zealand map.

Figure 6.26 – Sattelite image of Christchurch.

Figure 6.27 – Christchurch skyline with Southern Alps in background.

On February 22, 2011 at 2:51 p.m. local time, Christchurch was struck by a 6.3 earthquake on the Richter scale [21]. This natural disaster claimed 185 lives while injuring a thousand individuals. The infrastructure and building were seriously damaged as illustrated by figures 6.28, 6.29, 6.30 and 6.31. For instance, the Canterbury Television (CTV) building nearly entirely collapsed leaving only its lift shaft standing. A series of thirty subsequent earthquakes of smaller magnitude, referred to as aftershocks, happened for approximatively a year.

Figure 6.28 – Collapsed bell tower.

Figure 6.29 – CTV building.

Figure 6.30 – Soil liquefaction caused by the earthquake.

Figure 6.31 – Lanslide triggered by the earthquake.

6.5.2 Model

We propose to establish a damage map from LiDAR aerial surveys pre and post event by using our registration technique. We first transform both LiDAR acquisitions to elevation images. We geo-register both elevation images such that corresponding pixels of both images represent the same portion of the land. Such registration can be handled within GIS softwares for instance.

We were provided directly with elevation images of the downtown of Christchurch gridded at 1 meter. The figures 6.32 and 6.33 display these elevation images.

Figure 6.32 – LiDAR pre earthquake.

Figure 6.33 – LiDAR post earthquake.

Brighter grays represent higher elevation (Images have been enhanced for display).

To detect the damaged building, we propose to create a map that monitors the change of elevation between the pre and post event. If we were to dispose of perfect elevation images then a simple difference between the pre and the post event elevation image should create a perfect map. Unfortunately, the LiDAR acquisitions, and hence the derived elevation images, suffer from noise and artifacts. For instance, LiDAR might have difficulties to resolve with accuracy the edges of tall buildings. Therefore, we propose to use a registration framework with regularization to obtain the damage map.

We make use of notations I_{pre} and I_{post} for the pre and post geo-registered elevation images. Our goal is to find a transformation that registers I_{pre} to I_{post}. The dominant part of the transformation belongs to the change of elevation caused by building collapsing. The horizontal and vertical component of the transformation attempt to account for acquisition artifacts created near building edges. Hence, we formulate the following optimization problem:

$$\begin{aligned}\underset{(u_i\in[-1,1])_i,(v_i\in[-1,1])_i,(w_i\in\mathbb{R})_i}{\arg\min} & \sum_{(r_i,c_i)\in\Omega} |I_{pre}(r_i,c_i) - (I_{post}(r_i-v_i,c_i-u_i)-w_i)| \\ & + \sum_{(i,j)\in\mathcal{E}} \lambda^u|u_i-u_j| + \lambda^v|v_i-v_j| \\ & + \sum_{(i,j)\in\mathcal{E}} \lambda^w_{ij}|w_i-w_j|\end{aligned} \tag{6.18}$$

We set λ^u and λ^v to a small positive real value and we define for $(i,j)\in\mathcal{E}$:

$$\lambda^w_{ij} = \lambda^w_0 + \exp(-\lambda^w_1||I_{pre}(r_i,c_i) - I_{pre}(r_j,c_j)||^2_2)) \tag{6.19}$$

with $(\lambda_0^w, \lambda_1^w) \in \mathbb{R}^+ \times \mathbb{R}^+$.

We make use of first order primal dual techniques embedded in image pyramid scheme to optimize equation (6.18).

6.5.3 Experiments

As a baseline, we propose to simply subtract the post event image to the pre event image. We display the results of the both the baseline and our approach in figures 6.34 and 6.35.

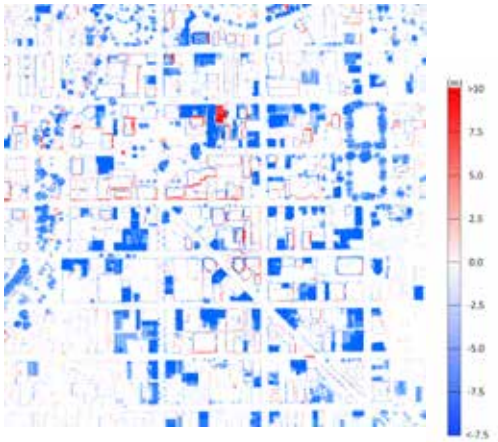


Figure 6.34 – Results from the baseline method.

Figure 6.35 – Results from our framework (elevation component only).

Blue color represents a negative change of elevation while the red color marks a positive change of elevation.

The result of the baseline technique in figure 6.34 highlights contour of each building, creating ghost contours. This is caused by the difficulties to resolve the edges of tall objects with aerial LiDAR acquisitions. However, our regularized framework produces a clearer damage map free of ghost contours while preserving fine details. We note since the LiDAR acquisitions are a few months apart, both methods capture vegetation changes. The disappearance of tree foliage in the post event acquisition creates a negative change of elevation. To enhance the quality of the damage map, one could classify building from vegetation by using an additional hyper-spectral information for instance.

6.6 Chapter conclusion

This final chapter illustrated the techniques presented in chapters 3, 4 and 5 with concrete applications ranging from stereo-matching to damage detection. We leave for future work the detailed study of each of these problems.

We strongly believe in the relevance of the presented techniques. Nevertheless, we are conscious that much work still remains to be done for achieving significant scientific contributions in tasks such as studying Earth crust deformation. However, we hope to have made accessible to other communities the mathematical concepts and methods to tackle those challenges.

Chapter 7

Conclusions, limits and future work

We conclude this thesis by summarizing each chapter and mentioning potential future research avenues.

First order Primal-Dual techniques for convex optimization This chapters introduced the basics of convex optimization and presented in detail the First order Primal-Dual techniques for convex optimization. A study of the dual solution space of TV regularized problems leads to the demonstration of various theorems. Using those theorems we were able to connect through their dual space several TV regularized models.

In future work one could attempt to extend the demonstrated theorems to other classes of regularized problems. Furthermore, one could investigate to decrease the computational cost of the first order Primal-Dual techniques by restricting the space of functions on which the convergence is ensured. Moreover, a detailed study of the implementation of such technique on GPU remains to be done.

Maxflow and graph cuts This chapter introduced the basics of non-convex optimization, the famous link between the maxflow and mincut problems, and presented two graph cuts techniques, namely the α expansion and Fast-PD. We proposed our own implementation of Fast-PD that drastically outperformed the original implementation.

In future work, one could investigate to extend graph-cut techniques to higher order terms as in [53], [85] or [92]. Furthermore, one could research on how to implement a CPU multi-threated code of Fast-PD. Moreover, efficient GPU based implementation of maxflow-minuct and graph cuts would also be greatly welcome by the computer vision community.

Coarsening schemes for optimization techniques In this chapter we studied coarsening frameworks for First order Primal-Dual techniques and for the graph cuts methods. We also introduced a new approach, the inference by learning framework, that drastically speeds-up the optimization of graph cuts methods.

In future work, one could research on novel ways to coarsen both nodes and labels. One could also investigate different coarsening approaches depending on how coarse the current scale is. Finally, a multi-grid approach could be used instead of the current pyramidal coarsening scheme.

Applications The final chapter of this thesis illustrated the technical methods presented in the previous chapters. In particular, we studied the stereo-matching problem to estimate depth from aerial and space surveys. We also illustrated how to retrieve from LiDAR acquisitions damages induced by earthquake. Finally, we showed with a simulated earthquake the potential of our approaches to track the ground deformation due to geological activities.

In future work, we hope that the geologists and remote sensing practitioners will build from our methods algorithms to extract pertinent information from aerial and space surveys. For instance, one could attempt to monitor the motion of glaciers and dunes, to estimate the ground deformation induced by real earthquakes, or monitor the change of volume created by landslides. All those measurements could help to design better physical model of those natural phenomena.

Bibliography

[1] Edward H Adelson, Charles H Anderson, James R Bergen, Peter J Burt, and Joan M Ogden. Pyramid methods in image processing. *RCA engineer*, 29(6):33–41, 1984.

[2] Andrés Almansa. *Echantillonnage, interpolation et détection: applications en imagerie satellitaire*. PhD thesis, Cachan, Ecole normale supérieure, 2002.

[3] Andrés Almansa, Coloma Ballester, Vicent Caselles, and Gloria Haro. A tv based restoration model with local constraints. *Journal of Scientific Computing*, 34(3):209–236, 2008.

[4] Francois Ayoub, Jean-Philippe Avouac, Antoine Lucas, Sebastien Leprince, and Nathan T Bridges. Measuring mars sand flux seasonality from a time series of hirise images. In *AGU Fall Meeting Abstracts*, volume 1, page 1873, 2012.

[5] Emanuel P Baltsavias. Airborne laser scanning: existing systems and firms and other resources. *ISPRS Journal of Photogrammetry and Remote sensing*, 54(2):164–198, 1999.

[6] Emmanuel P Baltsavias. Airborne laser scanning: basic relations and formulas. *ISPRS Journal of photogrammetry and remote sensing*, 54(2):199–214, 1999.

[7] Emmanuel P Baltsavias. A comparison between photogrammetry and laser scanning. *ISPRS Journal of photogrammetry and Remote Sensing*, 54(2):83–94, 1999.

[8] John L Barron, David J Fleet, Steven S Beauchemin, and TA Burkitt. Performance of optical flow techniques. In *Computer Vision and Pattern Recognition, 1992. Proceedings CVPR'92., 1992 IEEE Computer Society Conference on*, pages 236–242. IEEE, 1992.

[9] Paul L Basgall, Fred A Kruse, and Richard C Olsen. Comparison of lidar and stereo photogrammetric point clouds for change detection. In *SPIE Defense+ Security*, pages 90800R–90800R. International Society for Optics and Photonics, 2014.

[10] Joshua R Ben-Arie, Geoffrey J Hay, Ryan P Powers, Guillermo Castilla, and Benoît St-Onge. Development of a pit filling algorithm for lidar canopy height models. *Computers & Geosciences*, 35(9):1940–1949, 2009.

[11] Robert Bindschadler. Monitoring ice sheet behavior from space. *Reviews of Geophysics*, 36(1):79–104, 1998.

[12] Stan Birchfield and Carlo Tomasi. A pixel dissimilarity measure that is insensitive to image sampling. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 20(4):401–406, 1998.

[13] Herman Blinchikoff and Helen Krause. *Filtering in the time and frequency domains*. The Institution of Engineering and Technology, 2001.

[14] Stephen Boyd, Neal Parikh, Eric Chu, Borja Peleato, and Jonathan Eckstein. Distributed optimization and statistical learning via the alternating direction method of multipliers. *Found. Trends Mach. Learn.*, 3(1):1–122, January 2011.

[15] Stephen Boyd and Lieven Vandenberghe. *Convex Optimization*. Cambridge University Press, New York, NY, USA, 2004.

[16] Stephen Boyd and Lieven Vandenberghe. Localization and cutting-plane methods. *Stanford EE 364b lecture notes*, 2007.

[17] Yuri Boykov and Vladimir Kolmogorov. An experimental comparison of min-cut/max-flow algorithms for energy minimization in vision. *IEEE transactions on pattern analysis and machine intelligence*, 26(9):1124–1137, 2004.

[18] Yuri Boykov and Olga Veksler. Graph cuts in vision and graphics: Theories and applications. *Handbook of mathematical models in computer vision*, pages 79–96, 2006.

[19] Yuri Boykov, Olga Veksler, and Ramin Zabih. Markov random fields with efficient approximations. In *Computer vision and pattern recognition, 1998. Proceedings. 1998 IEEE computer society conference on*, pages 648–655. IEEE, 1998.

[20] Yuri Boykov, Olga Veksler, and Ramin Zabih. Fast approximate energy minimization via graph cuts. *IEEE Transactions on pattern analysis and machine intelligence*, 23(11):1222–1239, 2001.

[21] Brendon A Bradley and Misko Cubrinovski. Near-source strong ground motions observed in the 22 february 2011 christchurch earthquake. *Seismological Research Letters*, 82(6):853–865, 2011.

[22] Kristian Bredies, Karl Kunisch, and Thomas Pock. Total generalized variation. *SIAM J. Imag. Sci.*, 3(3):492–526, 2010.

[23] Lisa Gottesfeld Brown. A survey of image registration techniques. *ACM computing surveys (CSUR)*, 24(4):325–376, 1992.

[24] Antoni Buades, Bartomeu Coll, and Jean-Michel Morel. A review of image denoising algorithms, with a new one. *Multiscale Modeling & Simulation*, 4(2):490–530, 2005.

[25] Antonin Chambolle. An algorithm for total variation minimization and applications. *J. Math. Imaging Vis.*, 20(1-2):89–97, January 2004.

[26] Antonin Chambolle. Total variation minimization and a class of binary mrf models. In *International Workshop on Energy Minimization Methods in Computer Vision and Pattern Recognition*, pages 136–152. Springer, 2005.

[27] Antonin Chambolle and Thomas Pock. A first-order primal-dual algorithm for convex problems with applications to imaging. *J. Math. Imaging Vision*, 40(1):120–145, 2011.

[28] Tony F Chan, Selim Esedoglu, and Mila Nikolova. Algorithms for finding global minimizers of image segmentation and denoising models. *SIAM journal on applied mathematics*, 66(5):1632–1648, 2006.

[29] Chaur-Chin Chen and Hsueh-Ting Chu. Similarity measurement between images. In *Computer Software and Applications Conference, 2005. COMPSAC 2005. 29th Annual International*, volume 2, pages 41–42. IEEE, 2005.

[30] Gary E Christensen and Hans J Johnson. Consistent image registration. *IEEE transactions on medical imaging*, 20(7):568–582, 2001.

[31] John A Christian and Scott Cryan. A survey of lidar technology and its use in spacecraft relative navigation. In *AIAA Guidance, Navigation, and Control (GNC) Conference*, page 4641, 2013.

[32] Patrick L Combettes and Jean-Christophe Pesquet. A douglas–rachford splitting approach to nonsmooth convex variational signal recovery. *IEEE Journal of Selected Topics in Signal Processing*, 1(4):564–574, 2007.

[33] Patrick L Combettes and Jean-Christophe Pesquet. Proximal splitting methods in signal processing. In *Fixed-point algorithms for inverse problems in science and engineering*, pages 185–212. Springer, 2011.

[34] B Conejo, S Leprince, F Ayoub, and Jean-Philippe Avouac. Fast global stereo matching via energy pyramid minimization. *ISPRS Annals of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, 2(3):41, 2014.

[35] Bruno Conejo, Nikos Komodakis, Sebastien Leprince, and Jean Philippe Avouac. Inference by learning: Speeding-up graphical model optimization via a coarse-to-fine cascade of pruning classifiers. In *Advances in Neural Information Processing Systems*, pages 2105–2113, 2014.

[36] Thomas H.. Cormen, Charles Eric Leiserson, Ronald L Rivest, and Clifford Stein. *Introduction to algorithms*, volume 6. MIT press Cambridge, 2001.

[37] Corinna Cortes and Vladimir Vapnik. Support vector machine. *Machine learning*, 20(3):273–297, 1995.

[38] BM Costa, TA Battista, and SJ Pittman. Comparative evaluation of airborne lidar and ship-based multibeam sonar bathymetry and intensity for mapping coral reef ecosystems. *Remote Sensing of Environment*, 113(5):1082–1100, 2009.

[39] National Research Council et al. *Earth observations from space: The first 50 years of scientific achievements*. National Academies Press, 2008.

[40] George Dantzig. *Linear programming and extensions*. Princeton university press, 2016.

[41] Carlo de Franchis, Enric Meinhardt-Llopis, Julien Michel, J-M Morel, and Gabriele Facciolo. On stereo-rectification of pushbroom images. In *Image Processing (ICIP), 2014 IEEE International Conference on*, pages 5447–5451. IEEE, 2014.

[42] Carlo De Franchis, Enric Meinhardt-Llopis, Julien Michel, Jean-Michel Morel, and Gabriele Facciolo. An automatic and modular stereo pipeline for pushbroom images. *ISPRS Annals of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, 2(3):49, 2014.

[43] Andrea Donnellan, Bernard Hallet, and Sebastien Leprince. Gazing at the solar system: Capturing the evolution of dunes, faults, volcanoes, and ice from space. *Keck Institute journal*, 2014.

[44] Stuart E Dreyfus and Averill M Law. *Art and Theory of Dynamic Programming*. Academic Press, Inc., 1977.

[45] Roger D Eastman, Nathan S Netanyahu, and Jacqueline Le Moigne. Survey of image registration methods., 2011.

[46] Peter Elias, Amiel Feinstein, and Claude Shannon. A note on the maximum flow through a network. *IRE Transactions on Information Theory*, 2(4):117–119, 1956.

[47] Jeff Erickson, 2017.

[48] Pedro Felzenszwalb and Daniel Huttenlocher. Distance transforms of sampled functions. Technical report, Cornell University, 2004.

[49] Pedro F Felzenszwalb and Daniel P Huttenlocher. Efficient belief propagation for early vision. *International journal of computer vision*, 70(1):41–54, 2006.

[50] W. Fenchel, D.W. Blackett, and Princeton University. Dept. of Mathematics. Logistics Research Project. *Convex cones, sets, and functions*. Princeton University - Lecture Notes. Princeton University, Department of Mathematics, 1953.

[51] Werner Fenchel. On conjugate convex functions. *Canad. J. Math*, 1(73-77), 1949.

[52] Terrence J Finnegan. *Shooting the Front: Allied Aerial Reconnaissance and Photographic Interpretation on the Western Front–World War I*. Center for Strategic Intelligence Research National Defense, 2006.

[53] Alexander Fix, Aritanan Gruber, Endre Boros, and Ramin Zabih. A graph cut algorithm for higher-order markov random fields. In *Computer Vision (ICCV), 2011 IEEE International Conference on*, pages 1020–1027. IEEE, 2011.

[54] Lester R Ford and Delbert R Fulkerson. Maximal flow through a network. *Canadian journal of Mathematics*, 8(3):399–404, 1956.

[55] LR Ford Jr and DR Fulkerson. Maximal flow through a network. In *Classic papers in combinatorics*, pages 243–248. Springer, 2009.

[56] G David Forney. The viterbi algorithm. *Proceedings of the IEEE*, 61(3):268–278, 1973.

[57] Jerome Friedman, Trevor Hastie, Holger Höfling, Robert Tibshirani, et al. Pathwise coordinate optimization. *The Annals of Applied Statistics*, 1(2):302–332, 2007.

[58] Ed Gamble and Tomaso Poggio. Visual integration and detection of discontinuities: The key role of intensity edges. Technical report, MASSACHUSETTS INST OF TECH CAMBRIDGE ARTIFICIAL INTELLIGENCE LAB, 1987.

[59] Louis Geli, Pierre-Yves Bard, and Béatrice Jullien. The effect of topography on earthquake ground motion: a review and new results. *Bulletin of the Seismological Society of America*, 78(1):42–63, 1988.

[60] Robert J Geller. Earthquake prediction: a critical review. *Geophysical Journal International*, 131(3):425–450, 1997.

[61] DD Gilbertson, M Kent, and FB Pyatt. Aerial photography and satellite imagery. In *Practical Ecology for Geography and Biology*, pages 176–193. Springer, 1985.

[62] Ben Glocker, Nikos Komodakis, Nikos Paragios, and Nassir Navab. Non-rigid registration using discrete mrfs: Application to thoracic ct images. In *Workshop Evaluation of Methods for Pulmonary Image Registration, MICCAI 2010*, 2010.

[63] Ben Glocker, Nikos Komodakis, Georgios Tziritas, Nassir Navab, and Nikos Paragios. Dense image registration through mrfs and efficient linear programming. *Medical image analysis*, 12(6):731–741, 2008.

[64] Michel X Goemans and David P Williamson. The primal-dual method for approximation algorithms and its application to network design problems. *Approximation algorithms for NP-hard problems*, pages 144–191, 1997.

[65] Tom Goldstein, Min Li, and Xiaoming Yuan. Adaptive primal-dual splitting methods for statistical learning and image processing. In C. Cortes, N. D. Lawrence, D. D. Lee, M. Sugiyama, and R. Garnett, editors, *Advances in Neural Information Processing Systems 28*, pages 2089–2097. Curran Associates, Inc., 2015.

[66] S Gosh. History of photogrammetry. *Laval University, Canada*, 1981.

[67] A Ardeshir Goshtasby. *Image registration: Principles, tools and methods*. Springer Science & Business Media, 2012.

[68] Frédéric Guichard and François Malgouyres. Total variation based interpolation. In *Signal Processing Conference (EUSIPCO 1998), 9th European*, pages 1–4. IEEE, 1998.

[69] Ravi P Gupta. *Remote sensing geology*. Springer Science & Business Media, 2013.

[70] Shweta Gupta et al. A review and comprehensive comparison of image denoising techniques. In *Computing for Sustainable Global Development (INDIACom), 2014 International Conference on*, pages 972–976. IEEE, 2014.

[71] Roberto Gutierrez, Amy Neuenschwander, and Melba M Crawford. Development of laser waveform digitization for airborne lidar topographic mapping instrumentation. In *Geoscience and Remote Sensing Symposium, 2005. IGARSS'05. Proceedings. 2005 IEEE International*, volume 2, pages 1154–1157. IEEE, 2005.

[72] Joachim Hagenauer and Peter Hoeher. A viterbi algorithm with soft-decision outputs and its applications. In *Global Telecommunications Conference and Exhibition'Communications Technology for the 1990s and Beyond'(GLOBECOM), 1989. IEEE*, pages 1680–1686. IEEE, 1989.

[73] Bertil Hallert. Photogrammetry, basic principles and general survey. *TRIS*, 1960.

[74] TE Harris and FS Ross. Fundamentals of a method for evaluating rail net capacities. Technical report, DTIC Document, 1955.

[75] Stefan Harsdorf, Manfred Janssen, Rainer Reuter, Stefan Toeneboen, Berhard Wachowicz, and Rainer Willkomm. Submarine lidar for seafloor inspection. *Measurement Science and Technology*, 10(12):1178, 1999.

[76] Richard Hartley and Andrew Zisserman. *Multiple view geometry in computer vision*. Cambridge university press, 2003.

[77] Javier Hervás, José I Barredo, Paul L Rosin, Alessandro Pasuto, Franco Mantovani, and Sandro Silvano. Monitoring landslides from optical remotely sensed imagery: the case history of tessina landslide, italy. *Geomorphology*, 54(1):63–75, 2003.

[78] J.B. Hiriart-Urruty and C. Lemaréchal. *Fundamentals of Convex Analysis*. Grundlehren Text Editions. Springer Berlin Heidelberg, 2004.

[79] Heiko Hirschmuller. Accurate and efficient stereo processing by semi-global matching and mutual information. In *Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer Society Conference on*, volume 2, pages 807–814. IEEE, 2005.

[80] Eija Honkavaara, Roman Arbiol, Lauri Markelin, Lucas Martinez, Michael Cramer, Stéphane Bovet, Laure Chandelier, Risto Ilves, Sascha Klonus, Paul Marshal, et al. Digital airborne photogrammetry—a new tool for quantitative remote sensing?—a state-of-the-art review on radiometric aspects of digital photogrammetric images. *Remote Sensing*, 1(3):577–605, 2009.

[81] Yue Hu and Mathews Jacob. Higher degree total variation (hdtv) regularization for image recovery. *IEEE Trans. Image Process.*, 21(5):2559–2571, 2012.

[82] David R Hunter and Kenneth Lange. A tutorial on mm algorithms. *The American Statistician*, 58(1):30–37, 2004.

[83] Laura Igual, Javier Preciozzi, Luís Garrido, Andrés Almansa, Vicente Caselles, and Bernard Rougé. Automatic low baseline stereo in urban areas. *Inverse Problems and Imaging. 2007; 1 (2)*, 2007.

[84] Meilano Irwan, Fumiaki Kimata, Kazuro Hirahara, Takeshi Sagiya, and Atsushi Yamagiwa. Measuring ground deformations with 1-hz gps data: the 2003 tokachi-oki earthquake (preliminary report). *Earth, planets and space*, 56(3):389–393, 2004.

[85] Hiroshi Ishikawa. Higher-order gradient descent by fusion-move graph cut. In *Computer Vision, 2009 IEEE 12th International Conference on*, pages 568–574. IEEE, 2009.

[86] Michel Jaboyedoff, Pascal Horton, Marc-Henri Derron, Céline Longchamp, and Clément Michoud. Monitoring natural hazards. In *Encyclopedia of Natural Hazards*, pages 686–696. Springer, 2013.

[87] Ondřej Jamriška, Daniel Sỳkora, and Alexander Hornung. Grid cut code.

[88] Ondřej Jamriška, Daniel Sỳkora, and Alexander Hornung. Cache-efficient graph cuts on structured grids. In *Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference on*, pages 3673–3680. IEEE, 2012.

[89] LD Jones. Monitoring landslides in hazardous terrain using terrestrial lidar: an example from montserrat. *Quarterly journal of engineering geology and hydrogeology*, 39(4):371–373, 2006.

[90] Joerg Kappes, Bjoern Andres, Fred Hamprecht, Christoph Schnorr, Sebastian Nowozin, Dhruv Batra, Sungwoong Kim, Bernhard Kausler, Jan Lellmann, Nikos Komodakis, et al. A comparative study of modern inference techniques for discrete energy minimization problems. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1328–1335, 2013.

[91] Junhwan Kim et al. Visual correspondence using energy minimization and mutual information. In *Computer Vision, 2003. Proceedings. Ninth IEEE International Conference on*, pages 1033–1040. IEEE, 2003.

[92] Pushmeet Kohli, M Pawan Kumar, and Philip HS Torr. P3 & beyond: Solving energies with higher order cliques. In *Computer Vision and Pattern Recognition, 2007. CVPR'07. IEEE Conference on*, pages 1–8. IEEE, 2007.

[93] Vladimir Kolmogorov. Convergent tree-reweighted message passing for energy minimization. *IEEE transactions on pattern analysis and machine intelligence*, 28(10):1568–1583, 2006.

[94] Vladimir Kolmogorov and Yuri Boykov. Bk-maxflow code.

[95] Vladimir Kolmogorov and Ramin Zabin. What energy functions can be minimized via graph cuts? *IEEE transactions on pattern analysis and machine intelligence*, 26(2):147–159, 2004.

[96] Nikos Komodakis. Fastpd mrf optimization code.

[97] Nikos Komodakis. Towards more efficient and effective lp-based algorithms for mrf optimization. *Computer Vision–ECCV 2010*, pages 520–534, 2010.

[98] Nikos Komodakis, Nikos Paragios, and Georgios Tziritas. Mrf optimization via dual decomposition: Message-passing revisited. In *Computer Vision, 2007. ICCV 2007. IEEE 11th International Conference on*, pages 1–8. IEEE, 2007.

[99] Nikos Komodakis, Nikos Paragios, and Georgios Tziritas. Mrf energy minimization and beyond via dual decomposition. *IEEE transactions on pattern analysis and machine intelligence*, 33(3):531–552, 2011.

[100] Nikos Komodakis and Georgios Tziritas. A new framework for approximate labeling via graph cuts. In *Computer Vision, 2005. ICCV 2005. Tenth IEEE International Conference on*, volume 2, pages 1018–1025. IEEE, 2005.

[101] Nikos Komodakis and Georgios Tziritas. Approximate labeling via graph cuts based on linear programming. *IEEE transactions on pattern analysis and machine intelligence*, 29(8):1436–1453, 2007.

[102] Nikos Komodakis, Georgios Tziritas, and Nikos Paragios. Performance vs computational efficiency for optimizing single and dynamic mrfs: Setting the state of the art with primal-dual strategies. *Computer Vision and Image Understanding*, 112(1):14–29, 2008.

[103] K Kovari et al. General report: methods of monitoring landslides. In *Proceedings of the Fifth International Symposium on Landslides, 10-15 July 1988, Volume 3.*, pages 1421–1433. AA Balkema, 1990.

[104] John F Kreis, Alexander S Cochran Jr, Robert C Ehrhart, Thomas A Fabyanic, Robert F Futrell, and Williamson Murray. Piercing the fog: Intelligence and army air forces operations in world war 2. Technical report, DTIC Document, 1996.

[105] Sriram Krishnan, Chaitanya Baru, and Christopher Crosby. Evaluation of mapreduce for gridding lidar data. In *Cloud Computing Technology and Science (CloudCom), 2010 IEEE Second International Conference on*, pages 33–40. IEEE, 2010.

[106] NASA Landsat. Science data users handbook, 7.

[107] Donald T Lauer, Stanley A Morain, and Vincent V Salomonson. The landsat program: Its origins, evolution, and impacts. *Photogrammetric Engineering and Remote Sensing*, 63(7):831–838, 1997.

[108] Eugene L Lawler. *Combinatorial optimization: networks and matroids*. Courier Corporation, 2001.

[109] F Leberl, M Gruber, M Ponticelli, S Bernoegger, and R Perko. The ultracam large format aerial digital camera system. In *Proceedings of the American Society For Photogrammetry & Remote Sensing*, pages 5–9. sl]:[sn], 2003.

[110] Claude Lemaréchal and Claudia Sagastizábal. Practical aspects of the moreau–yosida regularization: Theoretical preliminaries. *SIAM J. Optim.*, 7(2):367–385, 1997.

[111] MJPM Lemmens. A survey on stereo matching techniques. *International Archives of Photogrammetry and Remote Sensing*, 27(B8):11–23, 1988.

[112] Sébastien Leprince, François Ayoub, Yann Klinger, and Jean-Philippe Avouac. Co-registration of optically sensed images and correlation (cosi-corr): An operational methodology for ground deformation measurements. In *Geoscience and Remote Sensing Symposium, 2007. IGARSS 2007. IEEE International*, pages 1943–1946. IEEE, 2007.

[113] John P Lewis. Fast normalized cross-correlation. In *Vision interface*, volume 10, pages 120–123, 1995.

[114] Jiangyu Liu and Jian Sun. Parallel graph-cuts by adaptive bottom-up merging. In *Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference on*, pages 2181–2188. IEEE, 2010.

[115] Thomas Luhmann. A historical review on panorama photogrammetry. *International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, 34(5/W16):8, 2004.

[116] François Malgouyres. Combining total variation and wavelet packet approaches for image deblurring. In *Variational and Level Set Methods in Computer Vision, 2001. Proceedings. IEEE Workshop on*, pages 57–64. IEEE, 2001.

[117] François Malgouyres. Mathematical analysis of a model which combines total variation and wavelet for image restoration. *Journal of information processes*, 2(1):1–10, 2002.

[118] François Malgouyres. Minimizing the total variation under a general convex constraint for image restoration. *IEEE transactions on image processing*, 11(12):1450–1456, 2002.

[119] André FT Martins, Mário AT Figueiredo, Pedro MQ Aguiar, Noah A Smith, and Eric P Xing. Ad3: alternating directions dual decomposition for map inference in graphical models. *Journal of Machine Learning Research*, 16:495–545, 2015.

[120] David Francis Maune. *Digital elevation model technologies and applications: the DEM users manual*. Asprs Publications, 2007.

[121] Scott McDougall and Oldrich Hungr. Dynamic modelling of entrainment in rapid landslides. *Canadian Geotechnical Journal*, 42(5):1437–1448, 2005.

[122] Alfred S McEwen, Eric M Eliason, James W Bergstrom, Nathan T Bridges, Candice J Hansen, W Alan Delamere, John A Grant, Virginia C Gulick, Kenneth E Herkenhoff, Laszlo Keszthelyi, et al. Mars reconnaissance orbiter's high resolution imaging science experiment (hirise). *Journal of Geophysical Research: Planets*, 112(E5), 2007.

[123] P Meixner and M Eckstein. Multi-temporal analysis of wwii reconnaissance photos. *ISPRS-International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, pages 973–978, 2016.

[124] Louis M Milne-Thomson. The calculus of finite differences. 1933.

[125] Renato DC Monteiro and Ilan Adler. Interior path following primal-dual algorithms. part i: Linear programming. *Mathematical programming*, 44(1):27–41, 1989.

[126] Jean-Jacques Moreau. Propriétés des applications "prox". *CR Acad. Sci. Paris*, 256:1069–1071, 1963.

[127] Jean-Jacques Moreau. Proximité et dualité dans un espace hilbertien. *Bull. Soc. Math. France*, 93(2):273–299, 1965.

[128] Mukesh C Motwani, Mukesh C Gadiya, and Rakhi C Motwani. Survey of image denoising techniques. In *Proceedings of GSPX*, 2004.

[129] Kevin P Murphy, Yair Weiss, and Michael I Jordan. Loopy belief propagation for approximate inference: An empirical study. In *Proceedings of the Fifteenth conference on Uncertainty in artificial intelligence*, pages 467–475. Morgan Kaufmann Publishers Inc., 1999.

[130] George L Nemhauser and Laurence A Wolsey. Integer programming and combinatorial optimization. *Wiley, Chichester. GL Nemhauser, MWP Savelsbergh, GS Sigismondi (1992). Constraint Classification for Mixed Integer Programming Formulations. COAL Bulletin*, 20:8–12, 1988.

[131] Yu. Nesterov. Smooth minimization of nonsmooth functions. *Math. Programming*, pages 127–152, 2005.

[132] Markus Neteler and Helena Mitasova. *Open source GIS: a GRASS GIS approach*, volume 689. Springer Science & Business Media, 2013.

[133] Mila Nikolova. *Optimization. Applications in image processing.* 2016.

[134] NASA/JPL/University of Arizona. High resolution imaging science experiment.

[135] Jaehong Oh. *Novel Approach to Epipolar Resampling of HRSI and Satellite Stereo Imagery-based Georeferencing of Aerial Images.* PhD thesis, The Ohio State University, 2011.

[136] Seungmi Oh, Hyenkyun Woo, Sangwoon Yun, and Myungjoo Kang. Non-convex hybrid total variation for image denoising. *J. Visual Commun. Image Represent.*, 24(3):332–344, 2013.

[137] Yoshimitsu Okada. Surface deformation due to shear and tensile faults in a half-space. *Bulletin of the seismological society of America*, 75(4):1135–1154, 1985.

[138] Francisco PM Oliveira and Joao Manuel RS Tavares. Medical image registration: a review. *Computer methods in biomechanics and biomedical engineering*, 17(2):73–93, 2014.

[139] National Research Council (US). Committee on Earthquake Engineering and National Research Council (US). Committee on Earthquake Engineering Research. *Liquefaction of soils during earthquakes*, volume 1. National Academies, 1985.

[140] Christos H Papadimitriou and Kenneth Steiglitz. *Combinatorial optimization: algorithms and complexity*. Courier Corporation, 1982.

[141] Neal Parikh and Stephen Boyd. Proximal algorithms. *Found. Trends Optim.*, 1(3):127–239, January 2014.

[142] R Gary Parker and Ronald L Rardin. *Discrete optimization*. Elsevier, 2014.

[143] Thomas Pock and Antonin Chambolle. Diagonal preconditioning for first order primal-dual algorithms in convex optimization. In *Computer Vision (ICCV), 2011 IEEE International Conference on*, pages 1762–1769. IEEE, 2011.

[144] Sorin C Popescu. Lidar remote sensing. *Advances in environmental remote sensing: Sensors, algorithms, and applications*, pages 57–84, 2011.

[145] R.T. Rockafellar. *Convex Analysis*. Princeton landmarks in mathematics and physics. Princeton University Press, 1970.

[146] Ana-Maria Rosu, Marc Pierrot-Deseilligny, Arthur Delorme, Renaud Binet, and Yann Klinger. Measurement of ground displacement from optical satellite image correlation using the free open-source software micmac. *ISPRS Journal of Photogrammetry and Remote Sensing*, 100:48–59, 2015.

[147] Tim Roughgarden. Lecture 2: Augmenting path algorithms for maximum flow, 2016.

[148] Leonid I Rudin, Stanley Osher, and Emad Fatemi. Nonlinear total variation based noise removal algorithms. *Physica D*, 60(1-4):259–268, 1992.

[149] Neus Sabater, Andrés Almansa, and Jean-Michel Morel. Meaningful matches in stereovision. *IEEE transactions on pattern analysis and machine intelligence*, 34(5):930–942, 2012.

[150] Neus Sabater, J-M Morel, and Andrés Almansa. How accurate can block matches be in stereo vision? *SIAM Journal on Imaging Sciences*, 4(1):472–500, 2011.

[151] Dirk Scherler, Sébastien Leprince, and Manfred R Strecker. Glacier-surface velocities in alpine terrain from optical satellite imagery—accuracy improvement and quality assessment. *Remote Sensing of Environment*, 112(10):3806–3819, 2008.

[152] Mark Schmidt and Karteek Alahari. Generalized fast approximate energy minimization via graph cuts: Alpha-expansion beta-shrink moves. *arXiv preprint arXiv:1108.5710*, 2011.

[153] Alexander Schrijver. *Combinatorial optimization: polyhedra and efficiency*, volume 24. Springer Science & Business Media, 2003.

[154] Arthur Schwalb. The tiros-n/noaa ag satellite series. *NASA STI/Recon Technical Report N*, 79, 1978.

[155] S Silvestro, LK Fenton, DA Vaz, NT Bridges, and GG Ori. Ripple migration and dune activity on mars: Evidence for dynamic wind processes. *Geophysical Research Letters*, 37(20), 2010.

[156] Aristeidis Sotiras, Christos Davatzikos, and Nikos Paragios. Deformable medical image registration: A survey. *IEEE transactions on medical imaging*, 32(7):1153–1190, 2013.

[157] Amelia Carolina Sparavigna. A study of moving sand dunes by means of satellite images. *International Journal of Sciences*, 2013.

[158] B St-Onge, C Vega, RA Fournier, and Y Hu. Mapping canopy height using a combination of digital stereo-photogrammetry and lidar. *International Journal of Remote Sensing*, 29(11):3343–3364, 2008.

[159] Sidney Sternberg and William G Stroud. Tiros i-meteorological satellite. *(NASA Goddard Space Flight Center Technical report*, 1960.

[160] Peter Sturm. Pinhole camera model. In *Computer Vision*, pages 610–613. Springer, 2014.

[161] Richard Szeliski, Ramin Zabih, Daniel Scharstein, Olga Veksler, Vladimir Kolmogorov, Aseem Agarwala, Marshall Tappen, and Carsten Rother. A comparative study of energy minimization methods for markov random fields with smoothness-based priors. *IEEE transactions on pattern analysis and machine intelligence*, 30(6):1068–1080, 2008.

[162] R. Takapoui and H. Javadi. Preconditioning via Diagonal Scaling. *ArXiv e-prints*, October 2016.

[163] Charles C Taylor. Measures of similarity between two images. *Lecture Notes-Monograph Series*, pages 382–391, 1991.

[164] J Thurston. How does satellite imagery compare with aerial photography. *The European Association of*, 2010.

[165] Olga Veksler and Andrew Delong. Alpha expansion code.

[166] Tanmay Verma and Dhruv Batra. Maxflow revisited: An empirical comparison of maxflow algorithms for dense vision problems. In *BMVC*, pages 1–12, 2012.

[167] Robert K Vincent. *Fundamentals of geological and environmental remote sensing*, volume 366. Prentice Hall Upper Saddle River, NJ, 1997.

[168] Paul Viola and William M Wells. Alignment by maximization of mutual information. In *Computer Vision, 1995. Proceedings., Fifth International Conference on*, pages 16–23. IEEE, 1995.

[169] David J Wald, Bruce C Worden, Vincent Quitoriano, and Kris L Pankow. Shakemap manual: technical manual, user's guide, and software guide. Technical report, 2005.

[170] Manuel Werlberger, Thomas Pock, and Horst Bischof. Motion estimation with non-local total variation regularization. In *Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference on*, pages 2464–2471. IEEE, 2010.

[171] Jonathan S Yedidia, William T Freeman, and Yair Weiss. Generalized belief propagation. In *Advances in neural information processing systems*, pages 689–695, 2001.

[172] Jae-Chern Yoo and Tae Hee Han. Fast normalized cross-correlation. *Circuits, Systems, and Signal Processing*, 28(6):819–843, 2009.

[173] Jing Yuan, Egil Bae, and Xue-Cheng Tai. A study on continuous max-flow and min-cut approaches. In *Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference on*, pages 2217–2224. IEEE, 2010.

[174] Jing Yuan, Egil Bae, and Xue-Cheng Tai. A study on continuous max-flow and min-cut approaches. In *Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference on*, pages 2217–2224. IEEE, 2010.

[175] Ramin Zabih and John Woodfill. Non-parametric local transforms for computing visual correspondence. In *European conference on computer vision*, pages 151–158. Springer, 1994.

[176] Sergey Zagoruyko and Nikos Komodakis. Learning to compare image patches via convolutional neural networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4353–4361, 2015.

[177] Mingqiang Zhu and Tony Chan. An efficient primal-dual hybrid gradient algorithm for total variation image restoration. *UCLA CAM Report*, 34, 2008.

[178] Barbara Zitova and Jan Flusser. Image registration methods: a survey. *Image and vision computing*, 21(11):977–1000, 2003.

Résumé

Dans le context de la vision par ordinateur cette thèse étudie le problème d'appariement d'images dans le cadre de la télédétection pour la géologie. Plus précisément, nous disposons dans ce travail de deux images de la même scène géographique, mais acquises à partir de deux points de vue différents et éventuellement à un autre moment. La tâche d'appariement est d'associer à chaque pixel de la première image un pixel de la seconde image.

Bien que ce problème soit relativement facile pour les êtres humains, il reste difficile à résoudre par un ordinateur. De nombreuses approches pour traiter cette tâche ont été proposées. Les techniques les plus prometteuses formulent la tâche comme un problème d'optimisation numérique. Malheureusement, le nombre d'inconnues ainsi que la nature de la fonction à optimiser rendent ce problème extrêmement difficile à résoudre. Cette thèse étudie deux approches avec un schéma multi-échelle pour résoudre le problème numérique sous-jacent.

Abstract

This thesis studies the computer vision problem of image registration in the context of geological remote sensing surveys. More precisely we dispose in this work of two images picturing the same geographical scene but acquired from two different view points and possibly at a different time. The task of registration is to associate to each pixel of the first image its counterpart in the second image.

While this problem is relatively easy for human-beings, it remains an open problem to solve it with a computer. Numerous approaches to address this task have been proposed. The most promising techniques formulate the task as a numerical optimization problem. Unfortunately, the number of unknowns along with the nature of the objective function make the optimization problem extremely difficult to solve. This thesis investigates two approaches along with a coarsening scheme to solve the underlying numerical problem.