HAL
open science

Video inpainting techniques : application to object removal and error concealment

Mounira Ebdelli

To cite this version:

Mounira Ebdelli. Video inpainting techniques : application to object removal and error concealment. Image Processing [eess.IV]. Université de Rennes 1, 2014. English. NNT : . tel-01093202

HAL Id: tel-01093202

https://inria.hal.science/tel-01093202v1

Submitted on 10 Dec 2014

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

N° d'ordre : 00000 ANNÉE 2014

THÈSE / UNIVERSITÉ DE RENNES 1
sous le sceau de l'Université Européenne de Bretagne

pour le grade de

DOCTEUR DE L'UNIVERSITÉ DE RENNES 1

Mention : Informatique

École doctorale Matisse

présentée par

Mounira EBDELLI

préparée à l' l'INRIA Rennes - Bretagne Atlantique
Institut National de Recherche en Informatique et en Automatique

---

Video inpainting techniques : application to object removal and error concealment

Thèse soutenue à Rennes le 20 Juin 2014

devant le jury composé de :

Frédéric DUFAUX
Directeur de recherche, LTCI Paris / *Rapporteur*

Philippe SALEMBIER
Professeur, Université Polytechnique de Catalogne / *Rapporteur*

Jean-Luc DUGELAY
Professeur, Eurécom Sophia Antipolis / *Examinateur*

Patrick PÉREZ
Chercheur, Technicolor R&D / *Examinateur*

Christine GUILLEMOT
Directeur de recherche INRIA Rennes / *Directrice de thèse*

Olivier LE MEUR
Maître de conférence, IRISA Rennes / *Co-directeur de thèse*

To the memory of my dad
To my family

Acknowledgements

I would like to thank Prof. Frédéric Dufaux, Prof. Philippe Salembier, Prof. Jean-Luc Dugelay, and Dr. Patrick Pérez for accepting to be members of the Ph.D. jury and for taking time to read and review this manuscript.

I would like to express my deep gratitude to my supervisor, Dr. Christine Guillemot, for her guidance through the work on this thesis and for the opportunity of working on this subject.
I also owe my gratitude to Dr. Olivier Le Meur for accepting to suppervise this work. His helpful and enthusiasm on research made this thesis more valuable.

I would also thank my labmates in SIROCCO group especially Marco Bevilacqua and Ronan Le Boulch. Finally, many thanks to my friends Safa Chérigui and Ahmed Jhinaoui for their support in difficult moments.

Contents

List of Figures v

List of Tables viii

Glossary ix

Résumé en Français :
Techniques d'inpainting vidéo : application à la suppression des objets et à la dissimulation des erreurs 1
1 Introduction 1
2 Contributions et Organisation du Manuscrit 4
2.1 Chapitre 2 : État de l'art des méthodes d'inpainting vidéo et de la dissimulation des erreurs 4
2.2 Chapitre 3 : Inpainting vidéo basé sur les méthodes neighbor embedding 4
2.3 Chapitre 4 : Étude et analyse des mesures de similarité pour les méthodes basées sur l'exemple 5
2.4 Chapitre 5 : Inpainting vidéo par optimisation globale . . . 6
2.5 Chapitre 6 : Recalage d'images pour l'inpainting de séquences à mouvement de la caméra 7
2.6 Chapitre 7 : Conclusion et perspectives 8
3 Conclusion 8

1 General introduction 9
1.1 Motivation 10
1.2 Contributions 11
1.3 Thesis organization 12

2 Background and Related Work 15
2.1 Video inpainting 15
2.1.1 What is inpainting? 15
2.1.2 Image inpainting techniques 17
2.1.2.1 Diffusion-based inpainting 17

2.1.2.2 Patch-based inpainting 18
2.1.2.3 Globally optimized approaches 22
2.1.2.4 Hybrid inpainting approaches 23
2.1.3 Video inpainting approaches 23
2.1.3.1 Globally optimized approaches 24
2.1.3.2 Locally optimized approaches 24
2.1.4 Video inpainting approaches comparison 25
2.2 Error Concealment 26
2.2.1 Video codec structure 27
2.2.2 Error detection 29
2.2.3 Error concealment 29
2.2.3.1 Forward Error Concealment 29
2.2.3.2 Post processing Error Concealment 32
2.2.3.3 Interactive Error Concealment 33
2.2.4 Model-based EC methods 33
2.3 Conclusion 34

3 Examplar-based video inpainting using neighbor embedding methods 35
3.1 Algorithm overview: background work 35
3.1.1 Inpainting the moving objects 36
3.1.2 Inpainting the stationary background 38
3.2 Patches sampling 39
3.2.1 Patches sampling based on Template Matching (TM) . . . 39
3.2.2 Patches sampling based on neighbor embedding techniques 39
3.2.2.1 Locally Linear Embedding (LLE) 40
3.2.2.2 Nonnegative Matrix Factorization (NMF) 40
3.2.3 Adaptive selection of the neighboring patches number . . . 41
3.2.4 Proposed patch sampling approach: competition between neighbor embedding techniques 42
3.3 Multi-resolution approach 42
3.3.1 Approximate NNF estimation 43
3.3.2 Iterated NNF 45
3.4 Application to error concealment 45
3.4.1 HEVC-based communication chain 46
3.4.1.1 Image partitions 46
3.4.1.2 Prediction modes 47
3.4.1.3 Information used by the loss concealment method 48
3.4.2 Motion recovery using space-time interpolation 48
3.5 Experimental results 49
3.5.1 Motion vectors recovery 49

3.5.2 Neighbor embedding techniques 49
3.5.2.1 Object removal 50
3.5.2.2 Error concealment 50
3.5.3 Multi-resolution approach 52
3.5.3.1 Quality-complexity trade-off of the multi-resolution approach 54
3.5.3.2 HEVC hierarchical temporal structure 56
3.6 Conclusion 59

4 Patches matching metrics 61
4.1 Similarity metrics 61
4.1.1 Pixel-based similarity metric 62
4.1.1.1 L_p-norm 62
4.1.1.2 Structural Similarity Information Measure (SSIM) 62
4.1.2 Statistic-based similarity metrics 63
4.1.2.1 Normalized Mutual Information (NMI) 63
4.1.2.2 Kullback-Leibler Divergence (KL-Div) 64
4.1.3 Hybrid distances: Bhattacharyya probability density based metric 64
4.1.4 Dominant orientation Template 65
4.1.5 Similarity metrics characteristics 65
4.2 Subjective similarity decisions 66
4.3 Experimental Results 66
4.3.1 Subjective test 66
4.3.2 Impact of similarity metrics on examplar-based approaches 68
4.3.2.1 Non Local means (NL-means) denoising 69
4.3.2.2 Inpainting application 70
4.4 Conclusion 72

5 Graph-based background inpainting 75
5.1 Graph-based inpainting approaches: state-of-the-art 76
5.1.1 Graph-based energy functions 76
5.1.1.1 Graph building 76
5.1.1.2 Energy functions 77
5.1.2 Graph-based background inpainting approaches 78
5.2 Proposed graph-based inpainting approach 79
5.2.1 Overview 79
5.2.2 Graph-based energy for background estimation 81
5.2.3 Impact of energy terms 82
5.2.3.1 Data term 82
5.2.3.2 Guiding term 83

5.2.3.3 Smoothness term 84
5.2.3.4 Energy convergence 85
5.3 Experiments results 86
5.3.1 Object removal 86
5.3.2 Error concealment 87
5.3.3 Background estimation 90
5.4 Conclusion 93

6 Image registration for inpainting free moving camera videos 97
6.1 Camera model 98
6.2 Epipolar geometry notions 99
6.2.1 Parallax effect 100
6.2.2 Epipolar constraint: fundamental matrix 100
6.2.3 Homography matrix 101
6.3 Motion model estimation for image registration 103
6.3.1 Feature-based registration 103
6.3.1.1 Feature detection 104
6.3.1.2 Features matching 105
6.3.2 Warping 105
6.3.3 Pixel-based method 106
6.4 Image registration for video editing applications 107
6.4.1 Analysis of multi-label homography-based registration . . . 107
6.4.2 Proposed region-based registration approach 111
6.5 Experiments results 114
6.5.1 Object removal 114
6.5.1.1 Results obtained from Change Detection dataset 114
6.5.1.2 Comparison with state-of-the art method 114
6.5.2 Error concealment 118
6.5.3 Running time 120
6.6 Conclusion 121

7 Conclusion and Perspectives 123
7.1 Summary of thesis contributions 123
7.2 Perspectives 124

Bibliography 135

Publications 137

List of Figures

2.1 Inpainting applications. 16
2.2 Inpainting problem. 16
2.3 Propagation direction using PDE. 17
2.4 The texture synthesis approach. 19
2.5 Examplar-Based inpainting approach. 20
2.6 Block diagram of a typical video communication system. 27
2.7 Encoding structure in H264/AVC. 28
2.8 Error propagation in frames sequence. 28
2.9 Slices coding types. 31

3.1 Overview of the video inpainting algorithm. 36
3.2 Foreground inpainting. 37
3.3 Illustration of the proposed approach. 42
3.4 Two-steps multi-resolution extension 43
3.5 Approximate NNF search 44
3.6 Evolution of the objective gain between two successive iterations . 45
3.7 Iteration number impact 46
3.8 Overview of the communication chain with the HEVC coder/decoder followed by the loss concealment. 46
3.9 Bilinear Motion Field Interpolation estimation (BMFI). 49
3.10 Performances of the BMFI method 50
3.11 Samples of patch estimation using TM and neighbor embedding techniques. 51
3.12 Impact of the neighbor embedding technique on inpainting results. 52
3.13 Comparison of different methods for recovering corrupted video. . 53
3.14 Performance comparison of the proposed approach with the state-of-the-art method in a context of error concealment. 54
3.15 Performances of the proposed approach in the error concealment application. 54
3.16 Test video sequences 55
3.17 Zoom on 4 blocks inpainted with the different methods. 56

4.1 Performances of the probability-based metrics. 64

4.2 Image test for subjective analysis. 66
4.3 Part of the web page used for gathering observers similarity decisions. 67
4.4 Agreement of observers' similarity decisions. 68
4.5 Agreement between subjective-based and metrics-based similarity decisions. 69
4.6 Image tests used for image denoising application. 70
4.7 PSNR values of NL-means denoising results using different similarity metrics and the best value of h. 70
4.8 NL-means denoising results using different similarity metrics. . . . 73
4.9 Image tests used for image inpainting application. 74

5.1 Smoothness constraint 77
5.2 Overview of the proposed approach 80
5.3 Background estimation: hole estimation step 81
5.4 Impact of the stationary term. 82
5.5 Guiding term using several approaches 84
5.6 Impact of smoothness term: inpainting labels for several value of α 85
5.7 Energy convergence 85
5.8 Inpainted results of change detection video sequence with dynamic background: fountain02 (432×288) 88
5.9 Inpainted results of change detection video sequence with dynamic background: fall(720×480) 89
5.10 Inpainted results of change detection video sequence with dynamic background: boats(320×240) 90
5.11 Inpainted results of change detection video sequence with shadow: peopleInShade (380×244) 91
5.12 Inpainted results of change detection video sequence with intermittent object motion: winter driver 91
5.13 Error concealment results of the sequence fountain01 92
5.14 Background estimation: sequence terrace 93
5.15 Background estimation 94
5.16 Background inpainting comparison parametric vs. non-parametric approaches 95

6.1 Projection model. 99
6.2 Parallax effect 100
6.3 Epipolar constraint 101
6.4 Homography projection and epipolar constraints. 102
6.5 Feature-based registration approach. 103
6.6 Features detection. 104
6.7 Features matching. 104
6.8 Percentage of matching features for 10 video sequences 109

6.9 Alignment results using MRF-based optimization. 110
6.10 Overview of the proposed registration approach. 111
6.11 Samples of region segmentation results using Mean Shift 112
6.12 Comparison of registration quality with scenes having different types of camera motion. 113
6.13 Inpainting result of badmindton video sequence (camera jitter motion). 115
6.14 Test frames of sequences proposed by Granados et al. [1]. 115
6.15 Comparison of our inpainting results with Granados's method [1]. 116
6.16 Comparison of our inpainting results with Granados's method [1]. 117
6.17 Test frames from $Scene11$ video sequences with varying lost rate. 118
6.18 Test frames from $Scene12$ video sequences with varying lost rate . 119
6.19 Test frames from $Scene13$ video sequences with variying lost rate. 120

List of Tables

2.1 Taxonomy of video inpainting methods. 26

3.1 PSNR values of concealed videos presented in Figure 3.13 and 3.15. 55
3.2 PSNR and computational times for BasketBallDrill sequence using a BMFI-based motion estimation and a scale factor of 0.25 56
3.3 PSNR and computational times for BasketBallDrill sequence using a BMFI-based motion estimation and a scale factor of 0.5 57
3.4 PSNR and computational times for BQMall sequence using a BMFI-based motion estimation and a scale factor of 0.25 57
3.5 PSNR and computational times for BQMall sequence using a BMFI-based motion estimation and a scale factor of 0.5 58
3.6 PSNR and computational times of inpainting input video at native high-resolution with the two-steps approach: interpolation and NNF-based super-resolution method. The scale factor is 0.25. . . 58
3.7 PSNR and computational times of inpainting input video at native high-resolution with the two-steps approach: interpolation and NNF-based super-resolution method. The scale factor is 0.5. . . . 59

4.1 Similarity metrics sensitivity. 65
4.2 PSNR values of denoising results of noisy images with $\sigma = 25$. . . 71
4.3 PSNR values of denoising results of noisy images with $\sigma = 40$. . . 71
4.4 PSNR values of inpainted images using the method in [2] (TM, searching window= 50×50). 72

6.1 PSNR values of inpainted videos in function of region numbers. . 121
6.2 Running time in function of the number of missing pixels. 121

Glossary

ATM Average Template Matching

AVC Advanced Video Coding

BMA Boundary Matching Algorithm

BMFI Bilinear Motion Field Interpolation

DLT Direct Linear Transform

FEC Forward Error Concealment

HEVC High Efficiency Video Coding

KL-Div Kullback-Leibler Divergence

LLE Locally Linear Embedding

LMS Least Median of Squares

LSE Least Square Estimation

MB Macro Block

MRF Markov Random Field

MSE Mean Square Error

MV Motion Vector

NLM Non-local Means

NMI Normalized Mutual Information

NMF Non-negative Matrix Factorization

OBMA Outer Boundary Matching Algorithm

PCA Principle Component Analysis

PDE Partial Differential Equation

PSNR Peak Signal-to-Noise Ratio

RANSAC RANdom SAmple Consensus

SIFT Scale Invariant Feature Transform

SSIM Structural Similarity Information Measure

SURF Speeded Up Robust Features

SSD Sum of Squared Differences

TM Template Matching

VLC Variable Length Coding

Résumé en Français : Techniques d'inpainting vidéo : application à la suppression des objets et à la dissimulation des erreurs

1 Introduction

L'inpainting vidéo est l'application qui vise à reconstruire des zones inconnues dans une séquence vidéo en utilisant l'information connue de son voisinage. Les regions à reconstruire peuvent correspondre à des zones endommagées ou des objets supprimés de la scène (logos, texte, etc.). Différentes méthodes d'inpainting vidéo ont été proposées dans la litérature. L'objectif principal de ces méthodes est d'estimer les pixels inconnus (de la zone cible) de manière à produire le résultat le plus plausible possible qu'il soit indétectable au sens perceptuel. La plupart des techniques d'inpainting vidéo sont l'extension de celles utilisées pour les images fixes. Ces méthodes peuvent être des approches d'optimisation locale généralement basée sur la similarité des exemples de texture (patches) ainsi que les vecteurs de mouvement [3–6] ou une optimisation globale d'une fonction d'énergie assurant la cohérence spatio-temporelle du résultat construit [1, 7].

Les techniques d'inpainting peuvent être utilisées dans plusieurs applications de traitement vidéo où une région inconnue de la scène doit être estimée telles que par exemple la suppression des objets ou la dissimulation des erreurs de transmission. Cette dernière technique est un post-traitement généralement effectué au décodeur pour récupérer des blocs corrompus ou perdus du flux décodé. Les méthodes classiques de dissimulation d'erreurs font usage de la corrélation spatio-temporelle dans une séquence vidéo pour reconstruire des blocs erronés ou perdus après transmission. Une méthode classique de dissimulation des erreurs basée sur une interpolation temporelle compensée en mouvement est généralement utilisée

par les standards de compression. Si le vecteur de mouvement correspondant à un block perdu est également perdu, une première étape d'estimation du vecteur de mouvement est nécessaire. Cependant, la plupart des méthodes d'estimation des vecteurs de mouvement dépendent de la disponibilité et la fiabilité des vecteurs des blocks voisins. Par conséquent, dans le cas d'un taux de perte élevé les méthodes de compensation de mouvement ne sont pas généralement efficaces.

Une utilisation des techniques d'inpainting vidéo, qui sont généralement basées sur la similarité de texture ainsi que l'information de mouvement peuvent permettre une meilleur estimation des blocks perdus. Dans cette thèse nous proposons une approche d'inpainting vidéo permettant de reconstruire de manière efficace les zones perdues dans différents types de séquence vidéos. L'approche proposée doit être aussi facilement adaptée à la dissimulation des erreurs. Dans la première partie du manuscrit nous nous focalisons sur l'inpainting des séquences à fond fixe (capturées par des caméras fixes). Ensuite, le cas des séquences capturées par une caméra en mouvement est considéré.

Deux catégories d'approches d'inpainting vidéo sont particulièrement étudiées. La première catégorie concerne une méthode basée sur l'exemple. Cette approche est constituée de trois étapes. Les zones inconnues des objets en mouvement dans la scène sont d'abord reconstruites. Ensuite les zones correspondant au fond statique sont estimées par l'information des images voisines dans un premier temps. Enfin, le reste des pixels inconnus est estimé en utilisant un inpainting spatial. A chaque itération, une recherche du patch (exemple) le plus prioritaire (patch cible) centré sur le bord de la zone à reconstruire est effectuée sous la base de la structure ainsi que le nombre de pixels inconnus dans le patch. Le patch le plus prioritaire est ensuite estimé par le patch source le plus similaire dans la partie connue de la vidéo. La similarité entre les patches est calculée en utilisant la distance euclidienne entre les vecteurs de mouvement ainsi que la valeur des pixels connus dans le patch cible (template) et ceux des pixels colocalisés dans le patch source.

Afin d'améliorer les résultats de cette approche, nous proposons dans le Chapitre 3 son extension sur plusieurs directions. Deux paramètres de l'algorithme sont d'abord étudiés. Le premier paramètre concerne l'approximation des pixels inconnus du patch à reconstruire. Au lieu d'estimer les pixels inconnus en utilisant un seul candidat qui est le patch le plus similaire par rapport au template, nous proposons de combiner linéairement plusieurs candidats. Les coefficients de pondération des candidats sont calculées en utilisant les pixels connus du template et en formulant le problème sous forme d'un problème aux moindres carrés avec différents types de contraintes. Les mêmes coefficients sont par la suite utilisés pour estimer les pixels inconnus. L'idée ici consiste à résoudre le problème des moindres carrées par deux techniques de neighbors embedding différentes.

Ensuite la meilleure estimation permettant d'approximer les pixels connus dans le template est considérée pour approximer les pixels inconnus par combinaison linéaire de leur correspondants dans les patches candidats.

La méthode d'inpainting est par la suite adaptée au contexte de dissimulation des erreurs. Pour cela, une étape de prétraitement qui consiste à estimer les vecteurs de mouvement des blocs perdus/erronés dans la vidéo est proposée. Cette étape permet de restreindre la région de recherche des meilleurs patches à une fenêtre compensée en mouvement. Une architecture en multi-résolution est également proposée pour réduire le temps de calcul.

Enfin, le deuxième paramètre de l'algorithme d'inpainting qui consiste à calculer la similarité des patches est étudié dans le Chapitre 4. Cette étude est basée sur des tests subjectifs et objectifs de comparaison des exemples en utilisant différentes mesures de similarité. L'impact de ces mesures sur la qualité des résultats de différents algorithmes de traitement d'image basée sur l'exemple est par la suite présenté.

Ainsi, l'approche considérée dans cette première partie de la thèse fournit un algorithme efficace et rapide capable de reconstruire correctement des petits trous correspondant à des blocks perdus ou la suppression des objets dans des séquences vidéos de petite résolution. Cependant, cette approche ne permet pas de résultats de haute qualité quand la région à estimer est large.

La deuxième partie de la thèse traite ce cas en utilisant une approche d'inpainting vidéo basée sur l'optimisation d'une fonction d'énergie permettant d'assurer la cohérence visuelle des régions reconstruites. En effet, ces méthodes d'inpainting permettent généralement une qualité de reconstruction meilleure que les méthodes basées sur l'exemple grâce à la contrainte de cohérence globale. Pour chaque pixel dans la zone à reconstruire, le coût de chaque candidat possible est calculé en utilisant deux termes : le terme de donnée et le terme de distorsion. Le terme de distorsion a pour but de forcer la cohérence entre chaque pair de pixels voisins de la zone à reconstruire ce qui permet des résultats avec une plus grande cohérence visuelle. Nous proposons dans le Chapitre 5 une fonction d'énergie robuste qui permet de reconstruire des zones larges dans des séquences vidéos de grande résolution. L'approche proposée utilise une fenêtre temporelle réduite ce qui rend l'algorithme plus rapide et adapté pour différentes applications.

Enfin, dans le Chapitre 6, nous considérons le cas des séquences à caméra en mouvement. Nous proposons un recalage des images voisines dans la séquence en utilisant une matrice d'homographie par région homogène.

2 Contributions et Organisation du Manuscrit

2.1 Chapitre 2 : État de l'art des méthodes d'inpainting vidéo et de la dissimulation des erreurs

Dans le chapitre 2, nous introduisons un état de l'art des méthodes d'inpainting vidéo ainsi que des techniques de dissimulation des erreurs de transmission. Deux catégories de méthodes d'inpainting vidéo sont principalement présentées. Les méthodes basées sur l'exemple ainsi que celles basées sur une optimisation globale, lesquelles sont résolues en utilisant les champs aléatoires de Markov. Une présentation brève des méthodes classiques de dissimulation des erreurs est par la suite effectuée.

2.2 Chapitre 3 : Inpainting vidéo basé sur les méthodes neighbor embedding

Dans ce chapitre, nous nous intéressons à la méthode d'inpainting vidéo basée sur l'exemple présentée dans [8]. Nous proposons d'améliorer la qualité et la complexité de cette approche par différentes contributions. D'abord, nous proposons une approximation des pixels inconnus de chaque patch à reconstruire en utilisant des méthodes neighbor embedding. Ces méthodes cherchent à combiner linéairement plusieurs candidats similaires au template. Le calcul des coefficients de pondération des candidats retenus est effectué sur les valeurs connues du template en formulant l'approximation par un problème aux moindres carrés sous la forme suivante :

$$\arg\min_{\omega} \|b - A\omega\|_2^2 \tag{1}$$

avec b est le vecteur colonne contenant les pixels connus du template, la matrice A est composée de colonnes dont chacune est formée par les pixels colocalisés dans les patches candidats et ω le vecteur des coefficients de pondération des candidats.

La résolution de ce problème est effectuée en utilisant deux techniques de réduction de dimensionnalités : locally linear embedding (LLE) et factorisation de matrice non négative (FMN). Ces deux techniques présentent différentes contraintes d'approximation. L'approche FMN, calcule la combinaison sous contrainte de positivité des pondérations de la combinaison ($\omega \geq 0$). Une approche itérative permettant de calculer les coefficients de pondération en utilisant la formule suivante :

$$\omega_k \leftarrow \omega_k \frac{\left(A^T b\right)_k}{\left(A^T A\omega\right)_k + \epsilon} \quad k = 1 \ldots K \tag{2}$$

avec X_k représente le k^{me} élément du vecteur X et ϵ une constante fixée à $\epsilon = 10^{-9}$.

D'autre part, la technique LLE calcule les pondérations avec la contrainte que leur somme est égale à 1 ($\sum_k \omega_k = 1$). Les coefficients $\omega_k \in \omega$ sont calculés par :

$$\omega = \frac{D^{-1}1}{1^T D^{-1} 1}$$

avec D est la matrice de covariance de la matrice A.

Ensuite, les deux approches utilisent les mêmes coefficients de pondération calculés par rapport aux pixels connus du template pour estimer les valeurs des pixels inconnues. Nous proposons dans ce chapitre une méthode d'approximation des pixels inconnus du template utilisant les avantages de trois différentes méthodes. Pour chaque patch à reconstruire, trois approximations des pixels inconnus sont calculées en utilisant les techniques TM, FMN et LLE. L'estimation la plus similaire au template par rapport aux pixels connus est enfin retenue pour estimer les pixels inconnus.

La deuxième contribution dans ce chapitre consiste à adapter l'approche d'inpainting vidéo à l'application de dissimulation des erreurs. Les vecteurs de mouvement des blocs perdus/erronés dans la vidéo sont alors estimés avant d'appliquer la méthode d'inpainting. Cette information permet aussi de restreindre la zone de recherche des meilleurs patches à une fenêtre compensée en mouvement et par conséquent réduit le temp de calcul et limite les artefacts dus à des erreurs de similarité des patches. Enfin, pour réduire d'avantage la complexité, une architecture en multi-résolution inspirée des méthodes de superresolution est proposée.

Les idées proposées dans ce chapitre ont été évaluées pour deux types d'applications : la suppression d'objets dans la scène ainsi que la dissimulation d'erreurs. Les expérimentations montrent l'efficacité de l'approche proposée dans les deux applications. Elle permet en effet de reconstruire correctement les zones avec moins d'artefacts visuels comparé aux méthodes classiques d'inpainting utilisant un simple template matching.

2.3 Chapitre 4 : Étude et analyse des mesures de similarité pour les méthodes basées sur l'exemple

Nous étudions dans ce chapitre un autre paramètre des algorithmes basés sur l'exemple qui est le calcul de la similarité des exemples (patches). L'idée principale de ces algorithmes est que chaque patch dans l'image peut être approché par une combinaison linéaire des pixels correspondants dans les exemples les plus similaires. La mesure de similarité a par conséquent un impact important sur les résultats de ces algorithmes.

Ce chapitre présente une analyse des performances des mesures de similarité les plus utilisées dans la littérature. Cette analyse est basée sur des tests subjectifs effectués par un ensemble d'observateurs. Ensuite, les scores de qualité objectifs sont calculés en utilisant différentes mesures de similarité afin de trouver la métrique la plus corrélée avec les tests perceptuels. Ces résultats sont enfin confirmés en étudiant l'impact des différentes métriques sur la qualité des résultats de deux algorithmes basés sur l'exemple.

2.4 Chapitre 5 : Inpainting vidéo par optimisation globale

Les algorithmes d'inpainting vidéo basés sur l'exemple ne permettent pas généralement une bonne qualité de reconstruction des zones larges. Ceci est dû principalement au fait que la contrainte d'optimisation de ces méthodes est calculée uniquement localement en fonction du patch à reconstruire à chaque itération. Ainsi, l'absence de contrainte globale de cohérence sur tous les pixels de la zone à estimer peut conduire à des artefacts qui sont plus visibles avec les larges régions.

Les méthodes d'inpainting vidéo basées sur l'optimisation globale d'une fonction d'énergie permettent une meilleure qualité de reconstruction dans le cas de grande zones. Ces méthodes appelées, méthodes basées graph-cuts ou approches multi-labels cherchent l'approximation des pixels perdus de la séquence vidéo en utilisant les champs aléatoires de Markov (MRF). Chaque pixel dans la région à reconstruire est estimé avec la valeur de pixel qui minimise une fonction d'énergie exprimant la cohérence spatio-temporelle du résultat. Ces méthodes sont caractérisées d'une complexité élevée rendant difficile leur utilisation dans des applications de traitement vidéo. Le but de ce chapitre est de proposer un algorithme d'inpainting vidéo basé sur l'optimisation globale d'une fonction d'énergie avec une complexité réduite. Pour simplifier le problème, nous considérons dans ce chapitre l'inpainting des régions perdues correspondant au fond statique dans les vidéos. Dans ce cas, l'objet à supprimer de la scène ne couvre aucun autre objet en mouvement. Ainsi, les pixels perdus doivent être estimés avec des valeurs de pixel d'arrière-plan statique. Une autre application de ce cas est l'estimation d'arrière-plan. Cette application vise à supprimer tous les objets en mouvement dans la scène. Elle diffère de la suppression d'objets dans une étape de pré-traitement où un masque binaire correspondant aux objets en mouvement doit être estimé au préalable.

Nous commençons par introduire une description brève des fonctions d'énergie proposées dans la littérature pour l'estimation de l'arrière-plan (background). Ensuite, nous présentons une description de la fonction d'énergie et de l'approche proposée. Les expérimentations avec plusieurs séquences vidéos de textures complexes illustrent l'efficacité de la méthode proposée dans plusieurs applications telles que la suppression d'objets, la dissimulation des erreurs et l'estimation du

backgound.

2.5 Chapitre 6 : Recalage d'images pour l'inpainting de séquences à mouvement de la caméra

Dans le cas d'une séquence vidéo capturée avec une caméra en mouvement, les images voisines de la scène ne présentent pas le même changement de point de vue. Ainsi, les algorithmes d'inpainting vidéo doivent d'abord aligner les images afin de compenser le mouvement de la caméra avant de recontruire les régions manquantes. Cette étape est très importante dans l'algorithme d'inpainting vidéo car les erreurs d'alignement peuvent conduire à des artefacts dans les résultats estimés. Dans ce chapitre, nous nous intéressons uniquement à cette étape d'alignement d'images appelée aussi recalage.

Les méthodes d'alignement doivent être robuste aux différentes caractéristiques des séquences vidéos : flou, changement d'illumination, mouvement des objets dans la scène etc. La majorité des approches d'alignement sont basées sur la correspondance de points entre les images de la même scène. Cette correspondance peut être calculée à l'aide d'une transformation géométrique basée sur la similarité entre les pixels traduisant ainsi le mouvement de la caméra entre les images. Le mouvement de la caméra peut être exprimé à l'aide d'un modèle de mouvement en trois dimensions calculé sur toute la séquence. Ce modèle est généralement d'une complexité elevée. Une approche plus rapide est de calculer le mouvement de la caméra à l'aide d'un modèle affine à deux dimensions décrivant le mouvement entre chaque paire d'images consécutives dans la séquence. Cette approche est souvent basée soit sur une minimisation des différences entre tous les pixels de l'image (modèle dense) ou un sous-ensemble de points caractéristiques extrait de chaque image [9, 10]. Cette dernière catégorie d'approche est très utilisée dans les applications de traitement des séquences capturées par une caméra en mouvement étand donné sa facilité de mise en oeuvre et sa complexité réduite comparée aux approches tridimensionnelles.

Les méthodes d'alignement bidimensionnelles peuvent être basées sur la correspondance du flux optique dense des points ou l'appariement d'un sous ensemble de points dans la paire d'images. La première méthode calcule le mouvement de la caméra en fonction des vecteurs de mouvement de chaque pixel ou bloc dans les images, alors que la deuxième approche utilise l'appariement d'un ensemble restreint de points à l'aide d'une matrice d'homographie exprimant la transformation géométrique entre les points considérés.

Dans cette thèse, nous avons choisi de résoudre le problème de recalage en utilisant une approche bidimensionnelle basée sur le calcul d'une matrice d'homographie. Nous présentons, dans une première partie de ce chapitre, les princi-

pales approches de recalage d'images décrites dans la littérature. Ces méthodes cherchent à calculer pour chaque pixel de l'image la matrice d'homographie optimale à partir d'un ensemble de matrices calculées au préalable entre les deux images. Cette approche n'est pas adaptée à la structure des différents plans dans la scène. Nous proposons ainsi une méthode d'alignement utilisant une matrice d'homographie par région. Une comparaison des performances de cette approche avec les méthodes récentes de l'état de l'art montre son efficacité permettant une meilleure qualité d'alignement.

2.6 Chapitre 7 : Conclusion et perspectives

Dans le chapitre 7, nous présentons une conclusion du travail réalisé en résumant les idées principales étudiées dans cette thèse et en proposant des perspectives sur les méthodes d'inpainting des séquences vidéos.

3 Conclusion

Le problème d'inpainting vidéo est traité dans cette thèse en utilisant deux catégories d'approches. Deux principales contributions ont été proposées pour les méthodes basées sur l'exemple. Une application des méthodes de neighbor embedding est d'abord proposée pour l'approximation des pixels perdus dans un patch. Deux méthodes de réduction de dimensionnalité FMN et LLE sont utilisées à chaque itération. L'estimation la plus similaire au template est par la suite choisie. Cette approche permet une meilleure qualité de reconstruction qu'une méthode classique de template matching. Une analyse des métriques de similarité des exemples a été aussi proposée en utilisant des tests subjectifs et objectifs afin de déterminer la meilleure métrique adaptée à la sensibilité perceptuelle. La méthode d'inpainting proposée a été appliquée pour la dissimulation des erreurs en utilisant une étape de prétraitement d'estimation de l'information de mouvement corrompue. Cette méthode montre de meilleures performances que les approches classiques de dissimulation des erreurs basée sur la compensation de mouvement. Une deuxième catégorie d'approches de vidéo inpaintinting a été aussi étudiée. Cette approche est basée sur l'optimisation d'une fonction de la cohérence spatio temporelle de la région reconstruite. Enfin, le problème d'inpainting des séquences vidéo capturée par des caméras en mouvement est résolu à l'aide d'une méthode d'alignement des images utilisant une homographie par région. Nous pensons que les méthodes proposées dans cette thèse pour résoudre le problème d'inpainting vidéo ouvrent de nouvelles perspectives permettant son utilisation dans différentes applications ainsi que le traitement de différents types de séquences.

Chapter 1

General introduction

Video inpainting refers to the field of computer vision which aims at filling-in holes in a video sequence using spatial and temporal information from neighboring regions. The holes may correspond to missing parts or removed objects from the scenes (logos, text, etc.). The main objective of video inpainting approaches is to complete the hole in a way which would be as visually plausible as possible both in space and time. To that end, most video inpainting techniques extend those used for still images, either by extending locally optimized approaches usually based on similarities between texture patches with the help of the motion information [3–6] or by ensuring spatio-temporal consistency with optimizing a global cost function [1, 7, 11].

Video inpainting techniques find a wide range of applications in video processing where an unknown region in the scene has to be estimated in a way providing visually coherent results. For instance, error/loss concealment which is a post-processing technique usually performed at the decoder side to recover missing blocks of decoded streams may represent an interesting application of video inpainting techniques.

Classical error concealment methods make use of the high spatial and temporal correlation of video content [12] to reconstruct errors and lost blocks caused by transmission over error-prone channels (e.g. IP and wireless networks) or limited-bandwidth networks based on the correctly received data. The spatial interpolation consists in estimating the missing pixels by smoothly interpolating surrounding pixels. While the temporal interpolation repeats the co-located pixels in previously correctly decoded frames. This method is only efficient on the static parts of images. In presence of motion, spatial and temporal interpolation can be combined [13] in a motion-compensated temporal interpolation to provide better concealment. Missing blocks are estimated by motion-compensating blocks from neighboring frames. If the Motion Vector (MV) is also lost, one has to estimate the MV first, typically by copying the MV of the block above or by interpolat-

ing the MV of previously decoded frames. Several approaches of motion vector estimation have been proposed in the literature. A well-known method used in H.264/AVC is the Boundary Matching Algorithm (BMA) [14]. This approach consists in recovering the lost motion vectors starting from the leftmost image block and then proceeding to the right and then downwards (raster scan order). If the motion vector of the leftmost image block is lost, the algorithm finds the first image block that has a correct motion vector and proceeds backwards. Another well used approach for motion vectors estimation is the Bilinear Motion Field Interpolation (BMFI) proposed in [15, 16], where the missing MVs are estimated using bilinear interpolation of those in the neighboring available Macro Blocks (MBs). Other hybrid extensions have also been proposed for efficiently estimating missing MVs. However, most of these methods usually depend on the availability and reliability of the neighboring MVs or pixels. Therefore, they can only be efficient for low MB loss rate and especially when there are sufficient available neighbors for missing MBs. In the case of heavy corruption, they may lead to severe artefacts. In this case, video inpainting techniques which are usually based on both texture and motion redundancies in the video content, may provide more visually consistent results than classical loss concealment techniques.

1.1 Motivation

In summary, this thesis is motivated by two major limitations of existing error concealment and video inpainting approaches. Loss concealment methods take into account the texture similarity between blocks only in motion vectors computation. The estimated motion vectors are then used to find and copy the compensated blocks. Thus, errors in motion analysis may lead to severe artifacts in the recovered regions. On the other hand, video inpainting approaches are usually time consuming and sometimes need human assistance to improve inpainting quality. Therefore, a direct application of existing video inpainting algorithms to correct losses in a decoded video may provide low quality of concealment results especially for high loss rates and introduces latency due to the high computation time. These limitations motivated for the development of a fast and unsupervised video inpainting technique capable of dealing with several kinds of videos captured either with static or moving cameras. To be suitable for applications such as object removal and loss concealment, the proposed approach should yield high visual quality while having a reduced complexity.

1.2 Contributions

To achieve the main objective in this thesis which is efficiently recovering space-time holes in different kinds of video scenes, two categories of video inpainting approaches are particularly studied. The first part of the thesis considers a fast and greedy optimized video inpainting method namely an exemplar-based approach. At each iteration, the approach finds the highest priority patch (target patch) in the missing region to be first inpainted. Then, the target patch is inpainted using the most similar patch in the known part of the video (source region). The similarity between patches is computed using the Euclidean distance between the known pixels in the target patch and the collocated pixels in the patch candidate (Template Matching(TM)). The priority is computed giving the amount of known pixels and the structure inside the patch to be inpainted. In order to improve the inpainting results of this approach, two important parameters of the algorithm are studied. The first parameter concerns the patch sampling step used to fill in the hole patch. Hence, instead of recovering each target patch using the most similar one according to the template, we consider instead neighbor embedding techniques which have been shown efficient in a context of still image inpainting [17]. Instead of using a single best matching patch these methods estimate each taget patch using a linear combination of several patches, taken from a source region in the image. Our contribution here consists in computing several estimations of the unknown pixels in the patch to be filled in using different neighbor embedding techniques and select the best estimation within the known pixels in the target to inpaint the unknown pixels.

Then, a pre-processing step which consists in estimating the motion vectors of the lost/erroneous blocks in the video is proposed to make the exemplar-based video inpainting approach suitable for loss concealment application. We propose to limit the search region of the best matching patches to a motion compensated window. This step helps to drastically reduce the computation time and limit artefacts due to errors in similarities between patches. Then, the computation time is further reduced by using a multi-resolution approach.

The second parameter concerns the similarity measure used to finding best matching patches. For this purpose, an analysis based on both subjective and objective tests has been performed. To confirm the results of this analysis, we then compare the performances of several examplar-based image processing algorithms using different similarity metrics to find the best correlated similarity metric with the perceptual results.

In summary, the approach presented in the first part of the document provides an efficient and fast algorithm able to deal with small holes corresponding to losses or object removal in low resolution video sequences. However, the proposed approach does not provide high quality results for large holes.

The second part of the thesis addresses this case by considering a video inpainting approach based on the optimization of a predefined cost function expressing the global consistency of the recovered regions. Globally optimized inpainting methods known as MRF-based approaches usually provide better quality than locally optimized methods thanks to the global constraint. Classical MRF-based inpainting algorithms compute, for each pixel in the hole, the energy cost of each possible known pixel candidate using two terms: the data term and smoothness term. How to efficiently design the energy terms is still a research topic. Usually, the data term computes the unary cost for each candidate pixel value to inpaint a given missing pixel. On the other hand, the smoothness term helps to compute the distortion or consistency induced by inpainting two neighboring pixels of the hole using two known pixels from different positions outside of the target region. In other words, this energy term favors that two inpainted neighboring pixels in the hole should be similar to a pair of adjacent pixels outside the hole. Using this constraint for all pixels of the hole helps to copy coherent segment from the known part of the image or video and therefore to provide results with increased visual coherence in the recovered areas.

Our first contribution in this part consists in proposing a robust energy function that helps recovering large holes even for high resolution videos sequences. Furthermore, the proposed approach uses a sliding temporal window to process overlapping group of frames, which makes the algorithm faster and suitable for video editing and loss concealment applications.

Finally, the third part of the thesis describes our contributions on compensating the camera motion to be capable to deal with free moving camera sequences. For this purpose we propose a registration of neighboring frames based on segmented regions in the source frames representing planar regions in the images.

1.3 Thesis organization

The thesis starts with a global introduction to state-of-the-art video inpainting and loss concealment approaches described in Chapter 2. This chapter first describes existing image and video inpainting algorithms. Various methods used for recovering error/loss blocks in a context of video transmission over networks with no guarantee in terms of quality of service are later detailed. Then, the rest of the manuscript is divided in three parts presenting the descriptions of our contributions for video inpainting approaches.

The first part addresses video inpainting using exemplar-based approaches, providing a fast and efficient solution for recovering errors and lost blocks in low resolution video sequences with static camera motion. In Chapter 3, a comparison of video inpainting quality using different neighbor embedding techniques to

compute the best matching patches is introduced. This algorithm has also been validated in a context of error concealment. For this purpose, the motion recovery is presented as a preprocessing step to limit artifacts in the texture of the inpainted region. A motion compensation window is also introduced to reduce computation time of searching the best matching patches in the known region of the video. A multi-resolution approach is later proposed to further reduce the complexity of the algorithm and enhance the recovery results. Finally, Chapter 4 presents and studies the impact of different similarity metrics for finding the best matching patches. Both subjective and objective tests are performed to compare the performances of the most used similarity measures.

The second part of the thesis addresses the question of efficiently recovering more complex scenes with large holes. We achieve this objective using a video inpainting algorithm based on the optimization of a well-defined energy function detailed in Chapter 5.

The last part of the thesis presented in Chapter 6 concerns video inpainting of free-moving camera sequences. A global introduction to existing state-of-the-art methods of registering images and compensating the camera motion is first presented. Then, our contribution for image registration using region-based homography is described.

Finally, Chapter 7 concludes on the contributions of the dissertation and discusses some perspectives.

Chapter 2

Background and Related Work

This chapter gives overview of the main state-of-the-art image and video inpainting methods as well as loss concealment approaches. In Section 2.1, classical image inpainting approaches are first described. Then, the main used techniques for inpainting holes in video sequences are addressed.

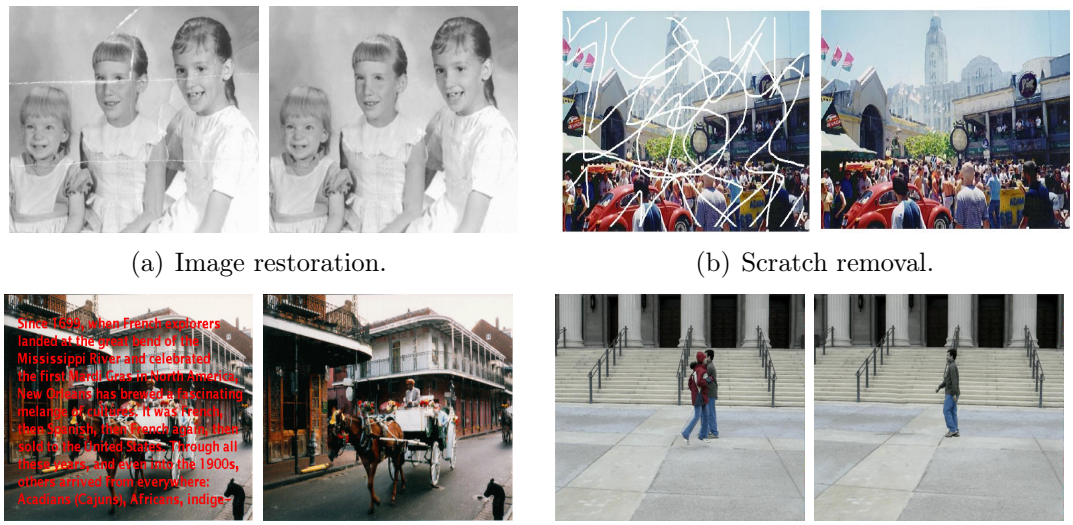
In Section 2.2 we focus on error/loss concealment which is a post-processing technique usually performed at the decoder side to estimate missing or erroneous blocks of decoded streams. This application can represent an interesting use case of video inpainting techniques. The target region to be inpainted is here the lost/erroneous blocks. We describe in this section existing error concealment techniques and the application's constraints to be able to provide an efficient inpainting method suitable for this application.

2.1 Video inpainting

2.1.1 What is inpainting?

Formally, inpainting is the way of extending paintings to automatically filling-in holes in image or video using information from surrounding area [18,19] in such a way that it is difficult to the observer to find the modified region within the image. Thereby, inpainting techniques also named image/video completion may find application in any image and video processing problems where a target region has to be estimated given its surrounding regions. As illustrated in Figure 2.1, regions to be inpainted may correspond to a damaged regions e.g. image gaps (Figure 2.1(a)) in old photos left by folds and scratches (Figure 2.1(b)) or holes corresponding to removing of unwanted objects from the scene (Figure 2.1(d)).

Mathematically, inpainting is an ill-posed inverse problem since there is no well-defined unique solution. Let's Ω the hole in an image I, the inpainting



(a) Image restoration. (b) Scratch removal.

(c) Text removal. (d) Object removal.

Figure 2.1: **Inpainting applications.** (Left) Input image. (Right) Inpainted result. (These images are taken from [3, 18])

problem can be defined as: $\forall p \in I$

$$\chi : \left| \begin{array}{l} \Omega \subset \mathbb{R}^n \rightarrow \mathbb{R}^m \\ p \mapsto I(p) \end{array} \right. \tag{2.1}$$

where $m = 3$ for RGB color images and n refers to the data dimension i.e. $n = 2$ ($p = (x, y)$) for image pixels (respectively $n = 3$ and $p = (x, y, t)$ for videos). The known region named source region is then defined as $\Phi = I \backslash \Omega$.

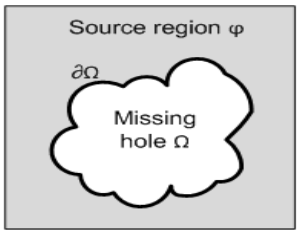


Figure 2.2: **Inpainting problem.**

To achieve their objective, almost all inpainting algorithms make use of the high degree of redundancy in the image and video content to extent data from outside the target region to inside. Samples from the source region are copied or diffused inside the hole based on given image local or global priors [20].

Since video inpainting approaches are often extensions of those used for image completion we start here by describing the most used image inpainting approaches before dealing with video inpainting methods.

2.1.2 Image inpainting techniques

Existing image inpainting methods can be broadly classified into three main categories: diffusion-based, patch-based and global optimizing approaches.

2.1.2.1 Diffusion-based inpainting

The pioneer inpainting approaches apply the same principle of the heat propagation in physical structures to the image context by smoothly propagating the local source information inside the hole to be filled. The propagation process is iteratively regularized using a given constraint model which may be linear, nonlinear, isotropic or anisotropic.

For instance, Bertalmio et al. [18, 21, 22] consider an anisotropic model that iteratively prolongs lines of constant intensity, known as isophotes or edges directions, at the border of the hole Ω (see Figure 2.3). The isophote direction is

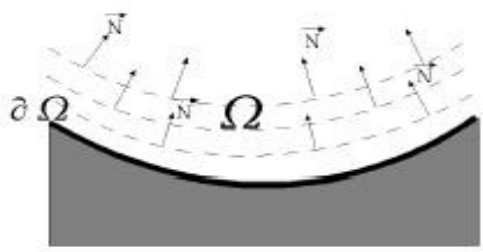


Figure 2.3: **Propagation direction using PDE.** (This image is taken from [18])

given by the normal to the discretized gradient vector ($\nabla I^{\perp}$). The propagation is then performed in a way preserving the edges directions by solving the equation: $\frac{\partial I}{\partial t} = \nabla(\Delta I)\nabla I^{\perp}$. Where $\nabla(\Delta I)$ is the Laplacian smoothness operation on the gradient image. In other terms, for each iteration n, the smoothed image I^{n+1} is given by:

$$I^{n+1}(i,j) = I^{n}(i,j) + \Delta_t {I_t}^{n}(i,j), \quad \forall (i,j) \in \Omega \tag{2.2}$$

where Δ_t is the weighting scale of the smoothed image ${I_t}^{n}(i,j)$ given by applying a Laplacian operation in the isophote direction: $I_t = \nabla(\Delta I)\nabla I^{\perp}$. The propagation iterates until a steady state solution:

$$I^{n+1}(i,j) = I^{n}(i,j) \quad \Rightarrow \quad \frac{\partial I}{\partial t} = \nabla(\Delta I)\nabla I^{\perp} = 0 \tag{2.3}$$

Several variants of this approach have then been proposed with the goal of providing better inpainting quality.

The diffusion-based inpainting problem may also be based on the Total Variational (TV) model [23] where the image is a function of bounded variation (BV). This approach firstly applied for denoising and deblurring applications [24] and extended for image inpainting in [25, 26] is based on the idea of searching for a function of BV defined on the input image I and which minimizes a TV energy inside the hole based on some fidelity constraints. The minimization problem is given by:

$$J_{TV}(\hat{I}) = \int_{\Phi \cup \Omega} \left| \nabla \hat{I}(x) \right| \, \mathrm{d}x + \lambda \int_{\Phi} (I(x) - \hat{I}(x))^2 \, \mathrm{d}x, \tag{2.4}$$

where the first integral term represents the regularization term, whereas the second term is a data term measuring the fidelity of the reconstructed image to the input image for the known samples. To better handle the curved structures, this approach has been extended to Curvature Driven Diffusion model (CDD) [27] by including the curvature information of the isophotes.

Almost all the diffusion-based inpainting approaches were focused on prolonging the structure at the border to inside the inpainted region. These approaches perform well for filling small gaps. However, they cannot deal with complex textures and large holes where they introduce blurring artifacts.

2.1.2.2 Patch-based inpainting

Another category of image inpainting methods solves the completion problem by using texture synthesis techniques which are sometimes referred to as region-growing methods. Texture synthesis approaches, illustrated in Figure 2.4, create new image pixels from an initial sample in a way which preserves the visual appearance of the sample [28–30]. Several synthesizing methods have been proposed in the literature in order to create textures with various statistical characteristics, gradient, color or intensity variations. A survey of the most used techniques can be found in [31]. The main difference between the texture synthesis and inpainting problems is that in texture synthesis applications, the hole is a square or rectangle corresponding to the extended region which will be often filled in a raster scan order. Whereas in the inpainting problem the hole may have an arbitrary form which may be filled in with textures or structures data depending on the surrounding information. Applying texture synthesis techniques to solve the inpainting problem in images, as proposed in [32], performs well in the case of images with regular textures even under varying brightness conditions. The computation time of this approach has been later drastically reduced in [33] by using a fast multi-resolution approach. For this purpose, the dimensions of the data representation are reduced using the Principal Component Analysis (PCA) and

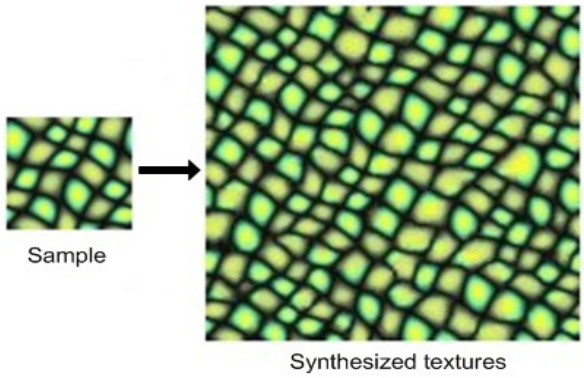


Figure 2.4: **The texture synthesis approach.** A large texture is produced from the input sample.

a Vector Quantization (VQ) is considered to speed up the matching comparison.

However, texture synthesis based inpainting approaches are not suitable for images with a lot of structures corresponding to object contours as most of the natural images. In this case, combining textures and structures data in the image helps to provide better results. Examplar-based approaches [2, 29] use this idea to iteratively sample and copy best matching patches from the source region into the hole. As illustrated in Figure 2.5, in each iteration the filling order of each patch centered on a pixel in the front line ($\partial\Omega$) is first computed based on a combination of the amount of known texture and the isophote direction ($\nabla I^{\perp}$). This combination is given by: $P(p) = C(p)D(p)$ where $C(p)$ is the confidence term expressing the amount of known pixels in the input patch and $D(p)$ the data term defined as:

$$D(p) = \frac{\left|\nabla {I_p}^{\perp} n_p\right|}{\alpha} \tag{2.5}$$

This term describes the structures (edges, corners) in the patch which helps to favor patches in which the isophote is perpendicular to the front line at p. Therefore, patches on the hole border with less unknown pixels and at the edge of an object have to be first inpainted.

Finally, the highest priority patch (Ψ_p) is filled in with the most similar patch in the source region (Φ). The similarity between patches is computed with respect to the known pixels in Ψ_p, so-called template. Several variants of this approach have then been proposed in the literature to improve the inpainting quality and the computation time. The main modifications of these variants concern the filling order and the patches matching.

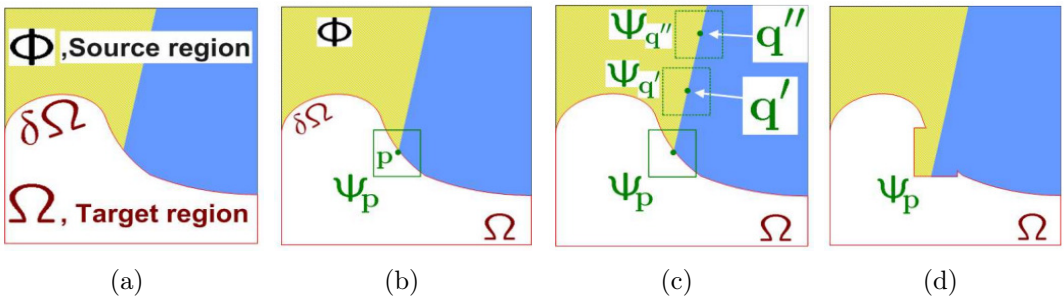


Figure 2.5: **Examplar-Based inpainting approach [2].** (a) Input image with the missing region (Ω). (b) Computation of the highest priority patch (Ψ_p). (c) Search of the best matching patch (Ψ_q) to Ψ_p in the source region (Φ). (d) Input image after inpainting a patch of Ψ_p using Ψ_q. (Illustration is taken from [2])

Filling order

To start by prolonging object edges at the border inside the hole region to avoid structures discontinuity in the inpainted image, the data term in equation 2.5 gives higher priority to patch where the isophotes is perpendicular to the front line. However, the gradient operator is sensitive to the image noise and being combined with the confidence term may provide wrong results leading to start by filling in flat image areas. Authors in [34] propose to enforce structure direction priors based on the structure tensor introduced in [35]. This approach computes the local geometry based on the discrepancy of the structure tensor eigenvalues (λ_1 and λ_2) and therefore evaluates the anisotropy of a local region. The data term is then given by:

$$D(p) = \alpha + (1-\alpha)exp\left(-\frac{\eta}{(\lambda_1 - \lambda_2)^2}\right) \tag{2.6}$$

where $\eta \geq 0$ and $\alpha \in [0,1]$. On flat regions ($\lambda_1 \approx \lambda_2$), no direction is favored for the propagation (isotropic filling order). Whereas in the presence of a structure ($\lambda_1 \succ\succ \lambda_2$), the data term $D(p)$ tends to be 1.

Another structure prior assumes that structural patches are, in general, sparsely distributed in the image and therefore have sparser nonzero similarities with their neighboring patches compared to textured patches. The structure priors can then be defined based on the sparse image representations. This idea introduced by Xu et al. in [36] shows that sparsity-based data term helps to better distinguish structural from textural patches compared to the isophotes-based method. For each pair of patches, a weight proportional to the similarity between them is computed. A large value of the structure sparsity term means sparse similarity with neighboring patches i.e. the input patch is probably on some structure, whereas

a small value means that the patch can be approximated by many candidates and therefore is likely to be a textural patch.

Patches matching

Once the highest priority patch is selected, founding its best matching patch in the source region is guided by two main parameters: the similarity metric and the searching algorithm.

Similarity metrics: the sum of squared differences (SSD) is the most widely used similarity metric for patches matching. However, as shown in [37], the SSD favors the copy of pixels from uniform regions. The weighted SSD metric based on Bhattacharya distance proposed in [37] helps to overcome this limitation. Given two patches Ψ_p and Ψ_q this metric is defined as: $d(\Psi_p, \Psi_q) = d_{SSD}(\Psi_p, \Psi_q) \times d_{BC}(\Psi_p, \Psi_q)$, where the term $d_{SSD}(\Psi_p, \Psi_q)$ is the SSD between the two patches and $d_{BC}(\Psi_p, \Psi_q)$ is the Bhattacharya distance given by $\sqrt{1 - \sum_k \sqrt{h_1 h_2}}$. $h1$ and $h2$ represent the histograms of the patches Ψ_p and Ψ_q, respectively. This metric provides wrong results when two patches have the same distribution since their Bhattacharya distance is zero and therefore a distance of zero even if one patch is a geometrically transformed version of the other one. A variant of the weighted Bhattacharya distance addressing this problem is proposed in [38] where $d(\Psi_p, \Psi_q) = d_{SSD}(\Psi_p, \Psi_q) \times (1 + d_{BC}(\Psi_p, \Psi_q))$. When two patches have the same distribution, the distance value is equal to the SSD between the two patches.

Nearest neighbors algorithms: regarding the search of the best matching patches also referred as nearest neighbors (NN), a naive solution consists in computing the distance between the patch to be filled in and all possible candidates in the source region which is time consuming. Variants of faster and approximate NN search methods have been proposed.

The k-dimensional trees (kd-trees) [39] and the vantage point trees (vp-trees) [40], organize the candidates in specific space-partitioning data structures according to their distribution in the search space to make faster the searching process based on the tree properties. However, the number of searched nodes increases exponentially with the space dimension.

A faster approximate Nearest Neighbors field (NNF) computation is proposed in [41]. This algorithm named PatchMatch computes the dense NNF based on the collaborative search. After initializing the NNF with a random guess or prior information, the algorithm iterates to refine the NNF results by using a propagation and random search at the patch level. The propagation step updates a patch offset with the known offsets of up and

left patches at even iterations and right and bottom patches at odd iterations. The second operation performs a local random search to refine the patch matches. A faster version of this algorithm has been proposed in [42] replacing the random search step with a hashing scheme. Good matches are propagated to nearby patches as well as to similar patches that were hashed to the same value. A general PatchMatch algorithm has also been proposed in [43] to find K-NNs instead of a single NN with extending the search space with transformations e.g. rotations, scaling.

2.1.2.3 Globally optimized approaches

Examplar-based approaches are usually greedily optimized methods which iteratively progress patch per patch into the hole. Each time, the optimization is performed locally i.e. the best matching patch to copy is only selected according to the template of the target to be filled in. For this reason, the result of these methods does not ensure a global image coherence. The visual quality of the result can be improved by maximizing the coherence over all the hole pixels. Globally optimized approaches try to achieve this goal by optimizing a cost function, also called energy function, computed over all missing pixels. Each pixel within the hole is inpainted with the unoccluded pixel value which globally minimizes the predefined energy cost. The energy cost expresses the visual coherence of each possible solution. The inpainting problem is then formulated, using a Markov Random Field (MRF) [44–48], as a 3-dimensional discrete offset field expressing for each missing pixel (p) the cost of each unoccluded values ($p+d_i \in \Phi$) that can be copied from the source region. This MRF can be optimized using belief propagation [47] or graph cuts [49]. The optimal offset field is computed as a graph labeling optimization minimizing the predefined cost function. A node in the graph corresponds to a pixel in the output image labeled by a shift (translational displacement). The energy cost should express the spatial coherence between the inpainted pixel using a given unoccluded pixel candidate (label) and its neighbors which enforces the global coherence in the inpainted hole. Several ways to search for candidates of each missing pixel can be used. In [48] a candidate (or label) for a missing pixel is simply the distance or translational displacement (offset) which maps it to a known pixel in the source region. Authors in [50] reduce the computational time of this approach by using a multiscale graph cut. Another label definition proposed in [51] is based on the sparsity of the offset's statistics of matching patches in the source region. The dominant offsets are used to form a stack of shifted versions of the input image which will be combined by optimizing a global MRF energy function ensuring neighborhood consistency. Therefore, the candidates of each missing pixel are here the colocated pixels in each one of the shifted images. Authors in [52] define the candidates as the collocated pixels in

similar images selected from internet.

Usually, energy-based methods produce visually pleasing and coherent inpainted images compared to the examplar-based approaches, in particular when the source region has homogeneous texture and few structures, however they suffer from a high computational cost.

2.1.2.4 Hybrid inpainting approaches

Inpainting approaches can also be a combination of different methods to make advantages of each of them. For instance, a completion approach based on both texture synthesis and PDE has been proposed in [53, 54]. First, the image is decomposed into structure and textures regions that will be filled in by edge propagating and texture synthesis techniques respectively. Authors in [54] start with a structure completion step using a segmentation based on the geometry, color and texture information in the input image. Then, the partitioning boundaries are extrapolated to generate a complete segmentation for the input using tensor voting [55]. The second step consists in synthesizing texture and color information in each segment, again using tensor voting. Whereas, Bugeau et al. [37] combine three approaches via an energy function expressing the texture synthesis, the propagation/diffusion and the coherence among neighboring pixels which helps to perform better than taking each term separately.

However, almost all hybrid approaches are computationally intensive especially for large holes since they also combine complexity of several methods.

2.1.3 Video inpainting approaches

Although, the image inpainting problem has been extensively studied in the literature, there are much less works for video content. This can be explained by the fact that when dealing with inpainting problem in video, not only the spatial coherence should be taken into account but also the temporal continuity among video frames needs to be ensured which introduces high time complexity. Thus, simply applying image inpainting approaches is not sufficient to provide pleasing results. Nevertheless, video inpainting methods are often the extensions of image completion algorithms by using additional constraints that aim to provide both space and temporal consistency. The temporal information is either considered by using a segmentation of video objects (e.g. tracking) or a global coherency in space-time patches. Besides, the filling of the hole can be performed based on greedy or globally optimized method. In the following subsections we describe the widely cited approaches in the literature broadly classified into local and global optimizing methods.

2.1.3.1 Globally optimized approaches

This category solves the inpainting problem by using a single energy function whose minima provides globally consistent results. Similarly to the image, each pixel or patch in the hole is associated to the optimal unoccluded values based on the similarity between 2D or 3D patches in videos data.

A well-known method of this category has been proposed by Wexler et al. [7]. This approach solves the video inpainting problem of static camera videos based on the optimization of a well-defined energy function. In a first step, space-time source patches are sampled to provide a database of patches statistics in the video. Then, the completion is performed by optimizing the cost function expressing the local consistencies by using a weight of the global completion quality provided by each possible pixel value. Each missing pixel is then assigned the most likely color among the patches database that optimizes the global coherence function. This approach can handle both structured and textured scenes and provide high inpainting results without any segmentation. However, the processing time is high even for low resolutions videos. The complexity of the search for the best matching patches has been reduced in [56] by using an extension of the patch matching algorithm in [41] to the video.

In [57], Shen et al. proposed also a global optimizing inpainting approach with low computational complexity by tracking every pixel in the occluded area through the video. The searching window for each missing pixel is then reduced from 3D to a 2D manifold provided by the tracking step. Unfortunately, this method can handle only translational or periodic objects motions.

Hu et al. proposed in [58] another global optimizing approach applying the image retargeting shift-map method in [48] to the video sequences. The approach searches the optimal displacements so-called shift map which is a vector fields of the correspondences between missing pixels and their corresponding unoccluded values to be used for the inpainting. The optimal shift map is computed based on a hierarchical framework. A 3D shift map in low resolution is first performed. Then, an incremental refinement using 2D shift map (refinement is only allowed spatially) is considered. This approach helps to provide high quality results with reduced complexity compared to the case of 3D shift map computed on the original resolution. However, spurious artifacts may be produced in the inpainted results due to the lack of temporal consistency in the refinement steps.

2.1.3.2 Locally optimized approaches

The Locally or greedy optimized approaches are usually based on video segmentation into moving foreground objects and background to be inpainted using separate algorithms.

This idea has been introduced by Patwardhan et al. in [8] by extending

examplar-based image inpainting approach in [2] to solve the video completion problem. The inpainting of the hole is performed with a pre-processing stage followed by three inpainting steps. In the pre-processing step the frames are segmented into moving objects (foreground) and static background. Then, the inpainting algorithm starts by filling in the missing regions corresponding to moving objects. The remaining holes which mainly correspond to the static background are first recovered by exploiting similarity between frames (temporal redundancy). Finally, the spatial inpainting in [2] is used for remaining holes. In each step, the priority of every patch located on the front line is computed based on the local amount of undamaged pixels and on the motion direction. Proceeding by highest priority, the method copies those patches that best match with the target patch with respect to the known pixels. The similarity between patches is here the distance between the patches textures and motion vectors. This method has been later extended to handle translational camera motions in [3] and more general camera motions in [6].

Another form of extending [2] to the video has been also introduced in [4] to modify the motion of people in videos. The boundary of the hole is also filled in based on the patches similarity in terms of texture and motion. The time complexity of the patches matching search is reduced here thanks to the building of a 2D skeleton model of each tracked person in the video. However, to be able to provide correct results, the object motions should be cyclic.

Venkatesh et al. [59] propose an object-based hole completion by tracking the occluded object and building a database of segmented frames where the occluded object is fully visible. The missing regions are then completed by aligning frames in the database to the partially or fully occluded frames in the hole. The contour of the occluded object is then introduced by Ling et al. [60] to retrieve the object frames from the database. Query postures are synthesized based on local segments of the object to find database frames. Similarly, Jia et al. [61] inpaint the missing holes of the occluded objects by aligning the object's visible trajectory.

Other approaches based on transfer of motion fields into the hole either by gradually propagating motion vectors [5] in the border area or by using motion similarities between patches [62]. These methods are likely to suffer from smoothing artifacts after few frames making the approach not well suited for completion over a high number of frames.

2.1.4 Video inpainting approaches comparison

In summary, locally optimized video inpainting approaches are often one-pass greedily propagating information from outside the hole into the missing region. These approaches are reasonably fast and enable fast completion and plausible results when the hole is small. However, similarly to fixe image, the methods

suffer from the lack of global visual coherence especially for large holes. Additionally, both periodic motion for each occluded object and accurate segmentation of moving objects and static background are often necessary to provide pleasing results. Otherwise, segmentation errors lead to severe artefacts. Furthermore, if the inpainting of moving objects is performed independently of the background, the blending between the completed foreground and the background can look unnatural.

In contrast, inpainting methods based on global optimization do not suffer from these limitations but cannot take advantage of model-based priors. Moreover, the computation time of these approaches is usually very high making difficult their use for high definition sequences. Table 2.1 summarizes the main characteristics of the well-known state-of-the-art video inpainting approaches of the two categories. One can mention that a fast method providing global coherent results for different types of sequences is not yet available.

Table 2.1: Taxonomy of video inpainting methods.

Methods	Bertalmio et al. [21]	Patwardhan et al. [3]	Venkatesh et al. [59]	Wexler et al. [7]	Granados et al. [1]	Granados et al. [11]
Optimization	Local	Local	Local	Global	Global	Global
Sensitivity to setting	No	Yes	Yes	No	Yes	Yes
Types of videos	Low resolution	Low resolution	Low resolution	Low resolution	High resolution	High resolution
Camera motions	Static	Slow motion	Slow motion	Static	Slow motion	Static
Object occluding	Small	Small to medium	Small to medium	Small	Large	Large
Complexity	High	Medium	Medium	High	High	High
Human interactions	No	No	No	No	No	Yes

2.2 Error Concealment

A possible application of video inpainting techniques may be the concealment of errors and lost blocks in decoded bit streams. However, several error concealment approaches have been already developed to ensure acceptable video quality. The main objective of these approaches is to reduce the impact of transmission errors by limiting error propagation due to the predictive coding while recovering

as possible erroneous blocks. Although being usually performed as a post processing at the decoder (Figure 2.6), error concealment techniques may also make use of the interaction between the encoder and decoder. Thus, the most used concealment techniques can be classified into three main categories namely forward, post processing and interactive approaches. In the following we describe the main used techniques in each one of these categories. In this section, we briefly review the classical error concealment approaches to better understand the constraints of the application and provide an idea of how to apply video inpainting techniques in this context.

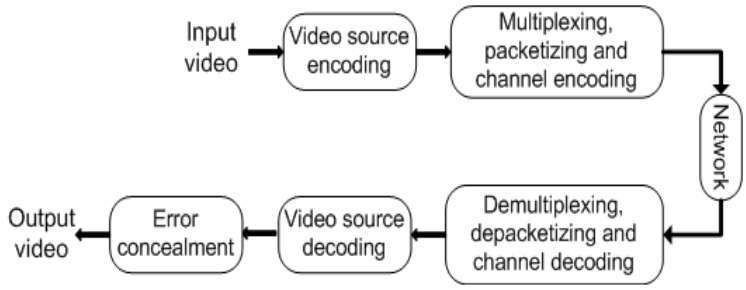


Figure 2.6: **Block diagram of a typical video communication system.**

2.2.1 Video codec structure

To be able to understand the details of error concealment techniques we start by describing the structure of the video codec. Video coding standards usually divide video sequences into groups of pictures (GOP). The GOP is the smallest segment of video which is self-contained, i.e. does not rely on any other frames for temporal or spatial information to be decoded. As illustrated in Figure 2.7, the first frame in each GOP is encoded without reference to other frames. Such encoding frame, known as I-frame, is added at regular intervals in the GOP representing key-frames or access points for random access of video files. The remaining frames are temporally predicted, including previous I-frame and other frames in forward inter-frame prediction mode (P-frame). Finally the bi-predictive inter frame known as B-frame, makes reference to both previous and future reference frames which are based on I-frame and/or P-frames. The temporally predicted i.e. P and B-frames help to reduce the bit rate while preserving the quality. However, they are sensitive to transmission errors because of their dependency on reference frames. As shown in Figure 2.8, transmission errors due to a burst error or packet loss may lead to decoder de-synchronization hence to temporal error propagation. Hence, the transmission of compressed video over error-prone

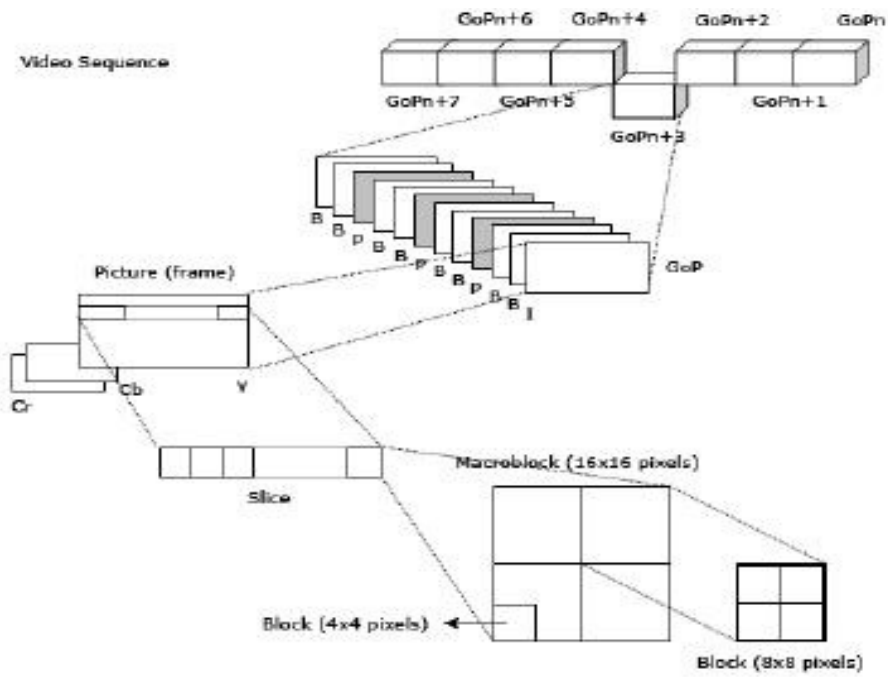


Figure 2.7: **Encoding structure in H264/AVC.**

channels, needs to be robust enough to handle errors and reduce error propagation. This can be achieved by improving the ability of the decoder to detect errors and conceal their effects.

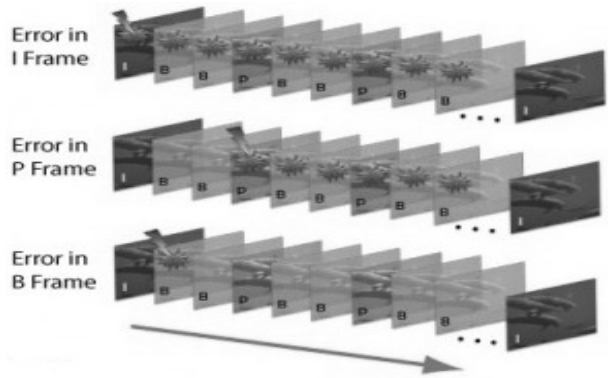


Figure 2.8: **Error propagation in frames sequence.**

2.2.2 Error detection

In general, before any error-concealment step, a first error/loss detection is needed to find transmission errors. Error detection can be performed either at the transport or the application layer i.e. the decoder. The first case concerns the lost packets or erasure channels where missing transport packets are detected using the packet sequence numbers in the header information. For instance, in packet-based video transmission, the parity bits or cyclic redundancy check (CRC) code is the widely used approach at the transport coder to detect errors to some finite accuracy.

2.2.3 Error concealment

Once the bit stream errors are detected, the decoder has to either conceal or ask for a packet retransmission of the corrupted data. Several approaches have been proposed in the literature to achieve this aim.

2.2.3.1 Forward Error Concealment

In contrary to classical source-coder design which aims to eliminate the statistical redundancy of the source signal to achieve the best compression gain, most of Forward Error Concealment (FEC) techniques introduce some redundancy at the source or channel coder that will help the decoder to recover transmission errors. We introduce in the following some of the widely used source coding and transport strategies to minimize the effect of transmission errors e.g. the Layered Coding with Transport Prioritization [63], Multiple Description Coding [64] and Joint Source and Channel Coding.

Layered Coding with Transport Prioritization: this is one of the most used and effective scheme for error resilience [12]. The video signal is divided into layers transported with different levels of priority or quality of service. Two types of layers are usually used namely base and enhancement layers. The base layer is transmitted with higher priority since it is able to produce a video signal of acceptable quality. While the enhancement layers are transmitted with lower priority and it is used to improve the video quality once it is correctly received. The main limitation of this approach is that it assumes to guarantee lossless transmission of the base layer which is not usually possible.

Multiple Description Coding (MDC): this technique produces several bit streams of the video and sends each one of them over different channels between the source and destination so that the probability that all channels simultaneously experience losses is small. The channels could be physically distinct paths

or a single physical path divided into several virtual channels by using time interleaving, frequency division, etc. To ensure the quality of decoded video, the descriptions are chosen in a way that each one can produce a video sequence with acceptable quality. Therefore, the descriptions have redundant information between them, which may decrease the coding efficiency and make the technique only suitable for highly losses channels.
As illustrated in Figure 2.9, a famous and simple way to compute multiple descriptions in H264/AVC codec consists in splitting adjacent samples among several subsampling and then coding each sub image independently. If one sub image is lost, it can be recovered based on the local correlation between adjacent samples in the image.

- *Interleaved slice groups* (*type* 0): every row is a different slice, alternating as many times as slice groups. Only horizontal prediction vectors are allowed.
- *Scattered or dispersed slice groups* (*type* 1): every macro block is a different slice. With two slice groups, it creates a checkerboard pattern; four or more groups also interleave rows, and with six slice groups, no macro block will ever touch another from the same slice group in any direction, maximizing error concealment opportunities.
- *Foreground groups* (*type* 2): specifying only the top-left and bottom-right of static rectangles to create regions of interest. All areas not covered are assigned to a final group.
- *Changing groups* (*types* $3-5$): Similar to type 2, but dynamic types that grow and shrink in a cyclic way. Only the growth rate, the direction and the position in the cycle have to be known.
- *Explicit groups* (*type* 6): an entire MBAmap is transmitted with groups arranged in any way the encoder wishes.

Joint Source and channel coding: source-channel interaction at a lower level can also be used to deal with transmission errors. This interaction can be performed using several forms such as the quantizer and entropy-coder design at the source coder and the design of FEC and modulation schemes at the channel coder. For instance, for noisier channels, fewer bits are used for the high-frequency coefficients and more bits are allocated to the low-frequency coefficients to ensure the minimum quality video.

Waveform Coding: this technique is similar to the Layered coding and MDC approaches in terms of intentionally keeping some redundancy in the source-coding step to ensure the robustness to channel errors. The main difference in the waveform coding is that the coded source signal is assumed to be transmitted in a single channel. A famous method of this technique is the motion-compensated temporal interpolation. The motion vector of the damaged macro blocks is first

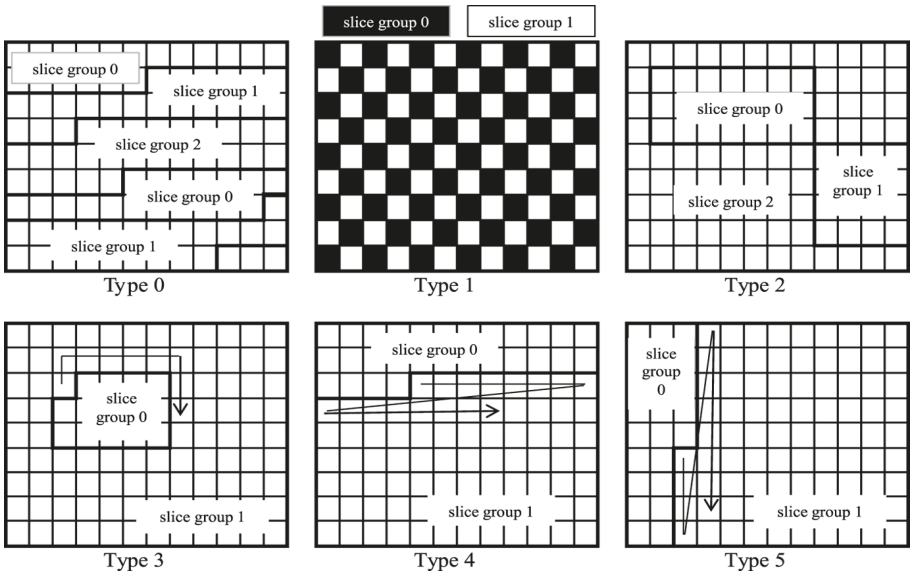


Figure 2.9: **Slices coding types.** (This image is taken from [65]).

estimated from those of the surrounding received macro blocks. Then, the damaged macro block is recovered from the corresponding motion-compensated macro blocks in the previous frame.

Entropy Coding: the redundancy can also be added in the entropy-coding step to help detect bit errors and limit error propagation. If Variable Length Coding (VLC) is used in video coding, a single bit error can lead to the loss of synchronization and the subsequent bits cannot be decoded. To solve this problem a synchronization code word has to be used at a fixed interval in the entropy coder such that the entropy decoder will regain synchronization once a code word is captured. Error propagation is then limited to the maximum separation between two synchronization code words. MPEG-4 allows the use of additional synchronization code word, known as motion marker, within each coded block between the motion information and the texture information [66]. If only the texture information is damaged, the motion information for a block can be used for better error concealment.

Transport-Level Control: forward error concealment can also be performed at the transport level with isolating the error using the structured packetization schemes in packet video. The main idea behind this technique is that when a packet is lost, the other packets can still be useful because the header and coding mode information is embedded into successive packets [67]. This is achieved by using interleaved packetization such that successive blocks are put into nonadja-

cent packets. Thus, a packet loss will affect blocks in an interleaved order i.e., a damaged block is surrounded by undamaged blocks, easy to be concealed at the decoder.

2.2.3.2 Post processing Error Concealment

Error concealment can also be performed as a post processing step by using the correctly decoded bit streams at the decoder and sometimes the auxiliary information provided by the source coder to improve the recovery quality. The idea is to dissimulate the effects of the transmission errors by smoothly propagating both in space and time data from the neighborhood of the lost/error area.

Motion-compensated temporal interpolation: this approach is one of the widely used method because of its simplicity. The concealment of each erroneous macro block is performed by using a motion compensation such as the compensated block is the one identified by the motion vectors of the damaged block. To provide better concealment quality this method is usually combined with the layered coding technique which includes the motion information in the base layer. When the motion vector (MV) is also corrupted, it needs to be estimated from the motion vectors of neighboring macro blocks before performing motion compensation of the block from the previous image. Lam et al. [14] estimate the missing motion vectors using a boundary matching algorithm (BMA). For each MV of the neighboring blocks, motion compensation is used. The side matched distortion (SMD) across boundaries of the block to be filled in and the compensated one is computed. The candidate MV that minimizes SMD is selected as the MV of the damaged block. This method can fail if the motion within the corrupted block is not purely translational or if the actual motion vector is not close to any of the neighboring vectors. The Bilinear motion field interpolation (BMFI) approach is known to produce better accurate motion estimation for different types of motion [16]. This method estimates the motion vector of each pixel p in the lost or corrupted block as a weighted bilinear interpolation of MV of the MVs of 4 nearest neighboring blocks. The weight of each MV neighboring block is computed according to its distance to p. This technique is widely used in various video standards, e.g. the MPEG standards. However, the main problem is that incorrect estimates of motion vectors can lead to large errors in the reconstructed images.

Maximally smooth recovery: this technique uses both the temporal and spatial redundancies of the video sequence and minimizes the energy difference between pixels of the corrupted block and its neighbor blocks (space-time neighborhood system). The idea behind it is to produce as smooth as possible video

signal. The smoothness energy function consists in minimizing a weighted sum difference between space-time adjacent pixels. Three kind of interpolations are performed namely frequency (within the block), temporal and spatial. If all coefficients of the block were lost, the problem became a motion compensated temporal estimation.

2.2.3.3 Interactive Error Concealment

The main idea behind these techniques is to use interaction between coder and decoder to enhance the error concealment performances. The decoder informs the coder about detected errors via the feedback channel. Then, the coder can uses this information to adapt the source coding or retransmit the erroneous data. This approach helps to produce better results than the forward and post processing error concealment techniques.

Several forms of interactive error concealment approaches can be used. For instance in the selective encoding approach, the decoder applies an error concealment technique in the damaged blocks. Once receiving the feedback from the decoder, the encoder adapts the prediction coding in a way to reduce the error propagation e.g. by sending the next frame in intramode. Other interactive concealment approaches can be based on retransmission process to correct the damaged received data. However, retransmission implies that the decoder has to wait for the retransmitted data to proceed which introduces a delay leading to freezes in the decoded video sequence. To deal with this problem, once a transmission error is detected, the decoder sends the retransmission request to the encoder, and continues the decoding with recording a trace of the error affected data due to error propagation. Once the retransmission data is received, the decoder uses this information and the trace to correct the damaged area [68].

2.2.4 Model-based EC methods

In summary, error concealment techniques aim at ensuring an acceptable video quality over error prone channel based on a trade-off between coding efficiency and error resilience. To this goal, classical methods described above are mainly based on the spatial or temporal correlation to predict the motion vectors (MVs) of the missing blocks, and interpolation, extrapolation or boundary matching schemes may produce good concealment results for low loss rate. Unfortunately, these approaches usually depend on the availability and reliability of the neighboring MVs or pixels making the methods not suitable in the case of heavily corrupted sequences. Several approaches have been proposed in the literature to provide high quality concealment in this case [69].

In this thesis we propose to solve this problem by using the video inpainting

techniques which are based on both textures and motion similarities in video data. Besides the spatial and temporal consistency constraint of these approaches may help to provide better visually pleasing results.

2.3 Conclusion

In this chapter, we reviewed the most used inpainting and error concealment approaches. Both techniques aim at estimating a region in the image or video with the information from surrounding area. This survey provides the necessary background to understand the limits of existing approaches and the motivation of providing efficient video inpainting approaches suitable for both applications. The rest of this manuscript describes our contributions for video inpainting in object removal as well as error concealment applications. In Chapter 3 the examplar-based video inpainting approach has been considered. Image inpainting based on neighbor embedding framework has been extended to the video inpainting problem. Additionally to the template matching technique two different dimensionality reduction methods namely the locally linear embedding and non-negative matrix factorization are used for approximating missing pixels. These methods lead to formulating the inpainting problem as a least squares problem with different constraints one with non-negativity and the other with sum-to-one for approximating the template signal. Each time, the best estimation of the three methods is used to inpaint the target patch. Then, a multiresolution approach is proposed to reduce computation time. In Chapter 4 an analysis of the most used similarity metrics for patches matching has been presented. First, a subjective test gathered by a group of observers is performed to find the best correlated similarity metric with perceptual decisions. Their impact on the results of two examplar-based approaches namely the Non Local Means denoising [70] and image inpainting [2] has been then studied. Then, in Chapter 5 the video inpainting problem is considered as a MRF optimization problem. A newly defined energy function is proposed to provide high inpainting quality for large holes. Finally in Chapter 6, the problem of inpainting the moving camera sequence is considered. Existing registration approaches as well as our contributions for registering neighboring frames are presented to study the impact of this step on the inpainting quality.

Chapter 3

Examplar-based video inpainting using neighbor embedding methods

In this chapter [1], we focus on the family of examplar-based inpainting methods. These greedy optimized approaches are fast enough to make them easy to be adapted to several video processing applications. A well-known video inpainting approach of this category has been proposed by Patwardhan at al. in [8]. We present an overview of this method in Section 3.1. Then, we propose to improve the inpainting results by extending the approach along several directions. In Section 3.2, a neighbor embedding patches sampling approach is first proposed to estimate missing patches in the hole. Thus, the unknown pixels in each patch in the hole are estimated as a linear combination of the K closest patches using neighbor embedding techniques. A multi-resolution approach is later proposed in Section 3.3 to reduce the computation time. Finally, Section 3.4 describes the application of the proposed approach to solve the error concealment problem in a HEVC-based communication chain. Two main contributions are proposed for this goal namely a preprocessing step for estimating the missing motion data and a motion-compensated window.

3.1 Algorithm overview: background work

As illustrated in Figure 3.1, the method in [3, 8] is a three-steps algorithm. The missing region corresponding to the moving objects is first inpainted. Then, the remaining missing background pixels are recovered into two steps. A temporal filling is first performed to copy known collocated pixels in the neighboring frames. Finally, the spatial examplar-based algorithm in [2] is applied to fill in the remaining holes. Thus, motion information of each pixel in the sequence is

1. The content of this chapter is related to our publications in [71, 72]

necessary to determine whether a pixel p belongs to a moving object ($M_c(p) = 1$) or to the static background ($M_c(p) = 0$). A preprocessing step is then performed to estimate the motion of each pixel in the scene before applying the above inpainting algorithm. The motion estimation can be provided by using an optical flow or a simple block matching algorithm. A threshold applied on the horizontal (V_x) and vertical (V_y) components allows determining whether the pixel belongs to the moving foreground object or to the stationary background. The process then follows the three above-mentioned inpainting steps.

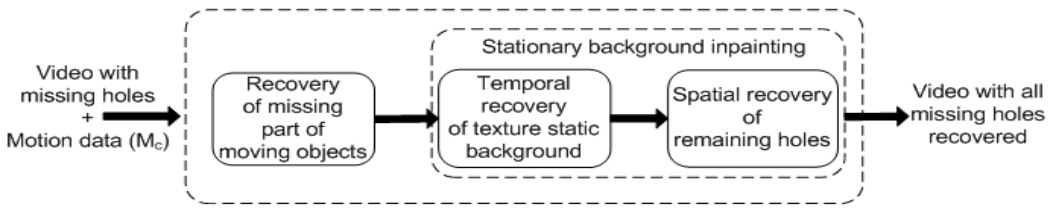


Figure 3.1: **Overview of the video inpainting algorithm.**

3.1.1 Inpainting the moving objects

The first step of the video inpainting algorithm which consists in filling in only the missing pixels belonging to the moving objects in the scene is achieved using the main steps described in the algorithm 1. Let I_t be the image at instant t which needs to be inpainted and $\Omega_t \in I_t$ the region to be filled in. Let $\phi = I_t \backslash \Omega_t$ be the known (or source) region in the image and $\delta\Omega_t$ be the border (or front line) of the region to be filled in. As described in Algorithm 1, at each iteration, the highest priority moving patch Ψ_p centered at pixel $p \in \delta\Omega_t$ is filled in with its most similar moving patch in the source region. A patch is considered moving if it contains at least one moving pixel i.e. $\exists x, x \in \Psi_p$ s.t. $M_c(x) = 1$.

The priority computation is performed on all pixels on the front line as follows: $P(p) = D(p)C(p), \forall p \in \delta\Omega$, where the confidence term $C(p)$ counts the amount of known moving pixels in the patch and the data term $D(p)$ aims at giving more priority to patches for which motion direction is orthogonal to the front line.

The most similar patch to the highest priority patch Ψ_p is then searched in the neighboring frames. The similarity between patches is computed by using a Template Matching (TM) given by the Sum Squared Differences (SSD) between vectors of five components (R, G, B, V_x, V_y), of known pixels in Ψ_p and their corresponding samples in each source patch, where R,G,B are the color pixel values and V_x and V_y denote respectively the horizontal and vertical components of the motion vector.

Algorithm 1: Inpainting of the moving foreground.

Input :

V: video sequence with a missing hole (Ω).
M_c: motion data.

Output:

$\tilde{V}$: where missing pixels of moving objects are inpainted.

Init : $C(p) = \begin{cases} 0 & \text{if p is missing.} \\ 1 & \text{otherwise.} \end{cases}$; $M_c(p) = \begin{cases} 1 & \text{if p is moving.} \\ 0 & \text{otherwise.} \end{cases}$

for $t = 1 \cdots N_{frames}$ **do**

repeat

Find the highest priority patch $\Psi_p \in I_t$ for motion filling:

$p = \arg\max(C(p)D(p))$

$C(p) = \frac{\sum_{i \in \Psi_p} C(i)}{\|\Psi_p\|}$, $D(p) = \frac{|\nabla M_c^{\perp} \cdot n_p|}{255}$

Find the best matching patch Ψ_q to Ψ_p.

Fill-in missing moving pixels of Ψ_p with their colocated in Ψ_q and update M_c of Ψ_p.

until *no more patch need motion filling in.*

end

(a) (b) (c)

Figure 3.2: **Foreground inpainting.** (a) Input image. (b) Mask of the object to be removed. (c) Output: only missing pixels of moving object are inpainted.

Once, Ψ_q, the patch with the closest template to Ψ_p is found, only the moving pixels in Ψ_q are copied to recover their collocated missing pixels in Ψ_p. The

priority of the remaining missing pixels in Ψ_p is set 0 to consider them non moving pixels and therefore have to be inpainted in the next steps. The algorithm iterates until all the pixels in the hole are either inpainted or have their priority equal to 0. The output of this step is illustrated in Figure 3.2.

3.1.2 Inpainting the stationary background

In this step the static background inpainting is addressed. Similar to the first step the priority of all patches centered on the fill front is computed as: $P(p) = C(p)D(p)$.

Algorithm 2: Background inpainting.

Input :

V: video sequence with missing pixels corresponding to the static background.

M_c: motion data.

Output:

$\tilde{V}$: inpainted video.

Init:

$C(p) = \begin{cases} 0 & \text{if p is damaged.} \\ 1 & \text{otherwise.} \end{cases}$; $M(p) = \begin{cases} 0 & \text{if p is moving or damaged.} \\ 1 & \text{otherwise.} \end{cases}$

repeat

Find the highest priority patch Ψ_p in the video for temporal filling:

$\{I_t, p\} = \arg\max(C(p).D(p))$

$C(p) = \frac{\sum_{i \in \Psi_p} C(i)}{\|\Psi_p\|}$; $D(p) = \frac{\sum_{i=t-n}^{t+n} M_i(p)}{2n+1}$

Find the highest confident patch $\Psi_{p\,F_i}$ in neighboring frames of I_t.

Copy non moving pixels of $\Psi_{p\,F_i}$ to Ψ_p.

until *no temporal info can be copied.*

Here again, the confidence term $C(p)$ computes the amount of known pixels per patch, while the data term $D(p)$ measures the amount of information available in previous and next frames at the location of the pixel p. Patches with the highest priority are copied first until $D(p) = 0$ which means that no more temporal information is available in the neighboring frames to fill in the remaining patches. Each inpainted patch is also copied to all collocated patches in the neighboring

frames to ensure background stationary. The remaining holes are then filled in using the spatial inpainting method in [2].

3.2 Patches sampling

In this section we focus on the step of patches sampling (texture generation) method used to fill in the missing pixels in the highest priority patch. Similar to the classical examplar-based inpainting methods, the algorithm proposed in [3,8] makes use of the Template Matching (TM) technique.

3.2.1 Patches sampling based on Template Matching (TM)

The template matching (TM) method estimates the unknown pixel values of each patch with copying their collocated pixel values in the best matching patch from the source region (ϕ) in the image or video. In others words, given a target patch Ψ_p (centered on a pixel p) where Ψ_p^u denotes the missing pixels values. Ψ_p^u is estimated with the best matching patch Ψ_q with respect to the known pixels in the target patch (Ψ_p^k) so-called template. Therefore, Ψ_q is the patch centered on q such that $q = \arg\min_{j \in \phi} \left\| \Psi_p^k - \Psi_j^k \right\|_2^2$. The unknown pixel values (Ψ_p^u) are then filled in with copying their collocated pixel values in Ψ_q.

3.2.2 Patches sampling based on neighbor embedding techniques

Template matching approach copy the collocated pixel values of a single best matching patch to the template. The degree of matching is usually computed by using the average of difference between collocated pixels. Thus, there might be several patch candidates which are either equally similar, or with a varying degree of matching but useful in filling in the missing pixel values. A combination of this set of patches, so-called nearest neighbors, may produces better results. Several forms of linear combination for computing the weighting coefficients of the patch candidates can be used. For instance, heuristic methods such as the simple Average Template Matching (ATM) uses the same weights for all candidates ($\omega_k = \frac{1}{K} \quad \forall k = 1 \ldots K$) while the Non-Local Means (NLM) approach computes each candidate weight as a function of the distance between the template and the corresponding values in the patch candidate. A better approach of weighting coefficients computation consists in optimizing the error between the template pixels and their collocated pixel values in the patch resulting of the weighted combination of the K-NN patches. The weights of the K-NN patches can then be

computed by solving the following least-squares problem:

$$min_{\omega} \left\| b - A\omega \right\|_2 \tag{3.1}$$

where b is the vector formed by the template pixel values Ψ_p^k, and the matrix A is defined such that the columns correspond to the set of the collocated values in the K patches i.e. $A = |a_1| \dots |a_K|$. a_i is the vector formed by the pixel values in Ψ_i^k. This kind of weighting coefficients computation helps to optimize the contribution of each patch candidate to approximate the missing pixels in the target (Ψ_p^u).

Authors in [17] show that solving the problem in equation 3.1 by using dimensionality reduction methods helps to provide better results in image inpainting than TM, ATM and NLM approaches. Thus, we propose here to extend these techniques to the video inpainting problem. Our extension consider two main data dimensionality reduction techniques: non-negative matrix factorization (NMF) and locally linear embedding (LLE), known as neighbor embedding methods. Both of these methods aim at approximating each input data as a linear combination of the K nearest neighbors (KNN) determined on the template pixels in the patch. The main difference between the two techniques is the constraint used to compute the weight of each candidate which makes their results different. Both techniques use the same steps to approximate the missing pixel values in the target patch. First, a training step is performed by approximating the template pixel values (Ψ_p^k) by a linear combination of the collocated pixels in the K-nearest neighbors patches $\Psi_i, i = 1 \dots K$. Then, the same weights are used to estimate the unknown values of Ψ_p^u.

3.2.2.1 Locally Linear Embedding (LLE)

Locally linear embedding (LLE) technique approximates each input data by a linear combination of its K-nearest neighbors (KNN) under the constraint that the sum of the weights is equal to 1 ($\sum_k \omega_k = 1$). The optimal weighting coefficients ω_k in ω are given by:

$$\omega = \frac{D^{-1}1}{1^T D^{-1} 1} \tag{3.2}$$

where D denotes the local covariance (Gram) matrix of the selected K-NN patches in A, and 1 is the column vector of ones. Finally, the unknown pixel values in the target patch are estimated by using the same weights as: $\Psi_p^u = \sum_{i=1}^{K} \omega_i \Psi_i^u$.

3.2.2.2 Nonnegative Matrix Factorization (NMF)

Non-negative Matrix Factorization (NMF) method searches for two lower dimensional nonnegative matrices whose product gives a good approximation of

the input data matrix [73]. The problem is here a least squares (LS) problem with constraints of positivity of the component matrices. The application of NMF technique to solve the inpainting problem is performed by fixing one matrix which is here the matrix A of the vectorized K-NN patches, and only the second matrix which represents the weights vector of the linear combination (ω) has to be computed. The NMF method then solves the least-squares problem as follow:

$$min_{\omega} \left(\|b - A\omega\|_2^2 \right) \text{ where } \omega \geq 0 \tag{3.3}$$

The most widely used solution for solving this constrained LS problem is the multiplicative update procedure [74]. This approach solves the NMF problem by iteratively updating the elements of each component matrix of the product. Here, we update only the randomly initialized non-negative matrix containing the weights ω_k of $\omega^T = (\omega_1 \ldots \omega_K)$ as:

$$\omega_k \leftarrow \omega_k \frac{\left(A^T b\right)_k}{\left(A^T A\omega\right)_k + \epsilon} \quad k = 1 \ldots K \tag{3.4}$$

where, k represents the kth element of the vectors corresponding to the input or K-NN patches stored in the matrix A and ϵ is a small constant ($\epsilon = 10^{-9}$).
The update in equation 3.4 is iterated until a pre-defined iteration number is reached or the difference, between two consecutive iterations, in the elements of ω is very small. Finally, the weights ω_k obtained in the last iteration of the NMF optimization are used to approximate the missing pixel values in the target patch such that $\Psi_p^u = \sum_{i=1}^{K} \omega_i \Psi_i^u$.

3.2.3 Adaptive selection of the neighboring patches number

The number K of the nearest neighboring patches to the template has an important impact on the inpainting result as well as the computational time. A low value may lead to low quality results and a high value increases the complexity while introducing smoothing effects on the inpainting results. Several methods could be used to estimate the best value of K. Here we use the method in [17] which is based on the similarity to the template as:

$$d_{min} = \min_{q \in \Phi} \frac{\left\| \Psi_p^k - \Psi_q^k \right\|_2^2}{\left| \Psi_p^k \right|} \tag{3.5}$$

All patches which the similarity distance to the template is less than αd_{min} ($\alpha \geq 1$) are combined to estimate the unknown pixel values in the target patch.

3.2.4 Proposed patch sampling approach: competition between neighbor embedding techniques

The proposed sampling approach aims at making advantages of different estimation techniques. In each iteration, three estimates of the target patch (Ψ_p) are computed using TM, LLE and NMF methods. Then, the estimation having the lowest distance with respect to the template Ψ_p^k is retained. Besides the textural information, it is required to compute the motion information M_c. For the template matching, this is simply given by the motion information of the best candidate patch. For LLE and NMF that combine several patches, the motion data of the best approximated patch is set to the motion data of the neighbor that has the highest correlation in terms of M_c with the known pixels of Ψ_p.

Figure 3.3: **Illustration of the proposed approach.** Frame being filled (top of the first column) and estimated patches (bottom row) using TM, LLE and NMF. Corresponding SSDs are 11064, 3295 and 4325 for TM, LLE, NMF respectively.

3.3 Multi-resolution approach

The second contribution in this chapter concerns the complexity of the inpainting approach. A multi-resolution extension is proposed to reduce the computational time. Figure 3.4 illustrates the architecture of this method. The video at the native resolution is first filtered and down sampled. Then the inpainting algorithm proceeds in two steps.

The first step consists in inpainting the low resolution version of the input video, with the method described in Section 3.1. The rationale of this first step is twofold. First, it is much easier to inpaint a low-resolution video sequence

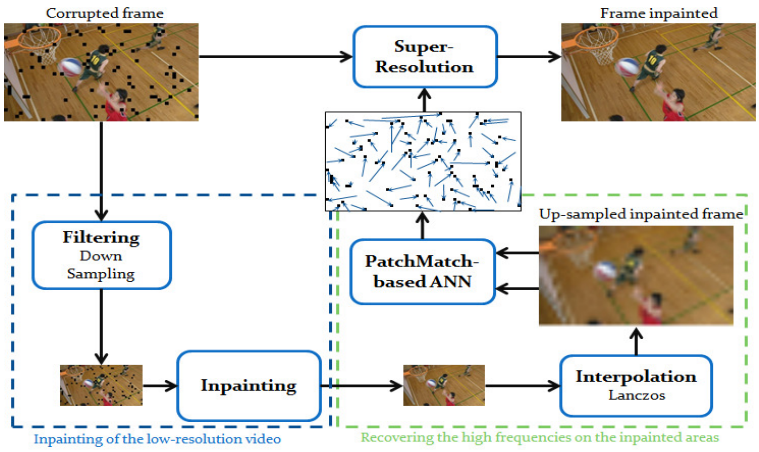


Figure 3.4: **Two-steps multi-resolution extension**. A low resolution version of the input video is first inpainted, and the details of the inpainted areas are retrieved using an ANN-based super-resolution technique (ANN stands for Approximate Nearest Neighbors.

than a high resolution one. The low-resolution picture contains less noise and less singularities which may affect the inpainting process. The second reason is simply the computational time required to inpaint a high resolution picture. Reducing the resolution significantly decreases the computational time.

The second step of the proposed method consists in retrieving the native resolution of the input video, by recovering the details of the inpainted areas. Two approaches are considered. The inpainted low resolution video is first interpolated using the Lanczos interpolation. As illustrated in Figure 3.5, the idea is then to search for the nearest neighbor (the best match) of the interpolated version of each inpainted block, within the known part of the current image at the native resolution. The found correspondences form a so-called nearest neighbor field (NNF). In other words, the NNF connects inpainted and interpolated patches of the low resolution video to high resolution patches of known parts of the high resolution (HR) video. The vectors of the NNF are then used to copy patch per patch HR pixels corresponding to the inpainted LR pixels.
The following sections describe in more details two important points which are the search for an approximate NNF and an iterative approach which further improves the visual quality of the inpainting.

3.3.1 Approximate NNF estimation

As the number of patches in an image is very high, an exhaustive search of the NNF is too time-consuming. A more suitable solution is to perform an approx-

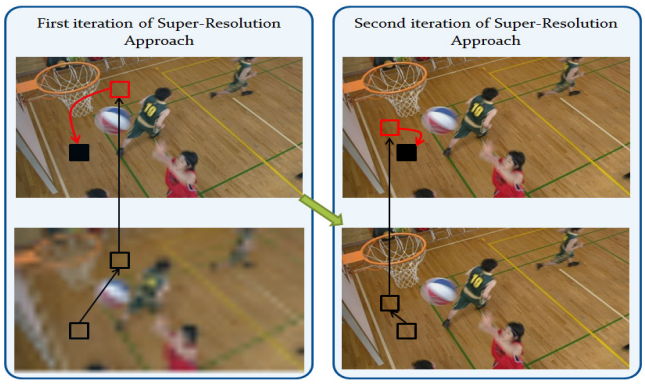


Figure 3.5: **Approximate NNF search.** The NNF is computed by using an iterated PatchMatch-based technique. At the first iteration, the PatchMath algorithm is fed with the upsampled low-resolution inpainted image. The result of the previous iteration is used as input to next iterations.

imate search of the NNF. We consider here the PatchMatch [41] method which is known to be a good trade-off between complexity and quality of the results. PatchMatch relies on the notion of image coherency. That is, if we find a pair of similar patches in two images, then their neighbors in the image plane are also likely to be similar. The method searches for a NNF by proceeding as follows. The NNF is first initialized by either random values or prior information. Initializing the NNF with a random guess is likely to provide few good guesses. The NNF is then iteratively refined by interleaving two operations called Propagation and Random search at the patch level. The propagation step updates a patch offset with the known offsets of its causal neighborhood, exploiting the image coherency. At even iterations, offsets are propagated from up and left patches, whereas at odd iterations, offsets are propagated from right and bottom patches. The second operation performs a local random search to seed the patch matches, and these matches are then propagated by iterating a small number of times.

In our context, the approximate NNF is computed from the upsampled low-resolution inpainted image, as illustrated in figure 3.4 (bottom right). The low-resolution inpainted image is first upsampled in order to get the native resolution, avoiding the need to rescale the displacement vectors. The PatchMatch method is then applied considering that the first guess is chosen randomly and that the method iterates 5 times. Note that the search for nearest neighbors is just performed in the neighborhood of missing areas (i.e. within a window of size 5×5).

3.3.2 Iterated NNF

The approximate NNF is here refined iteratively. Indeed, at the first iteration, the NNF is computed from the upsampled low-resolution inpainted image. However, to take into account that the inpainting quality is likely to be better near the front line than inside the block to be filled-in, the computation of NNF is done several times. At iteration i, the result of the previous iteration $i-1$ is used as an input to the next iteration of the PatchMatch, as illustrated by Figure 3.5.

Figure 3.6 illustrates the visual improvement per iteration for a given image of the BasketBallDrill sequence (in Figure 3.5). The gain is visually perceptible especially for the second iteration which is confirmed by visual quality shown in Figure 3.7. Figure 3.6 gives the gain of PSNR compared to the previous

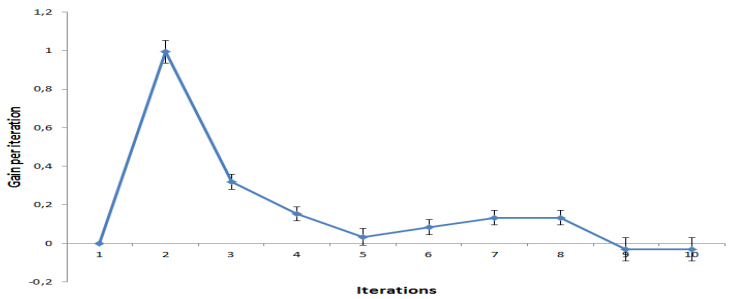


Figure 3.6: **Evolution of the objective gain (in dB) between two successive iterations**. Error bars represents SEM (standard error of the mean).

iteration. It suggests that the gain is the highest at the second iteration and decreases to become negative after nine iterations. We have chosen to set the number of iterations to 5, which seems to be a good trade-off between quality and complexity.

3.4 Application to error concealment

Our goal here is to adapt the proposed video inpainting approach initially used for object removal to the error concealment application. The main difference between the two applications concerns the motion data. This input of the algorithm can be provided by a simple thresholding of motions vectors in object removal context. However, in error concealment context the motion vectors corresponding to the lost blocks are also lost and therefore need to be first estimated. We focus in this section on the preprocessing step of the estimation of motion data corresponding to the lost blocks in a communication chain with an HEVC codec.

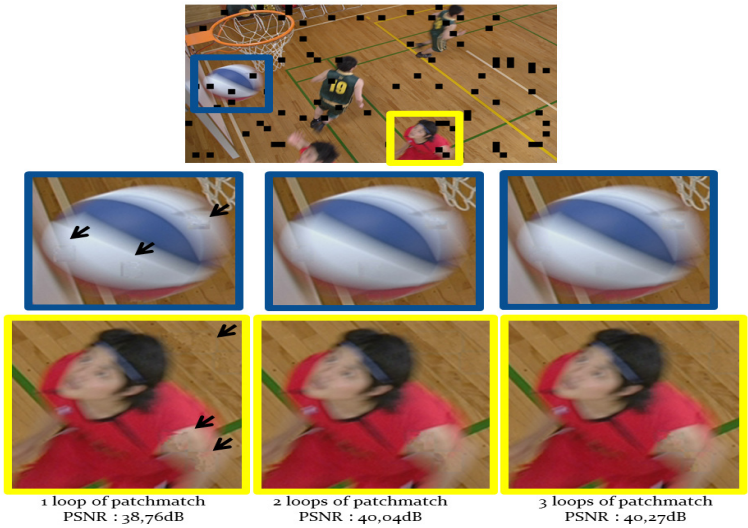


Figure 3.7: **Iteration number impact.** Left: impaired image. Right: results for the first three iterations. Blacks arrows indicate the areas where the benefit to use an iterated PatchMatch is visually perceptible.

3.4.1 HEVC-based communication chain

Figure 3.8 shows a typical communication chain with an HEVC encoder and decoder followed by loss concealment approach. Let we first review the key features of the HEVC compression algorithm which are particularly relevant for the loss concealment process, especially for motion information recovery.

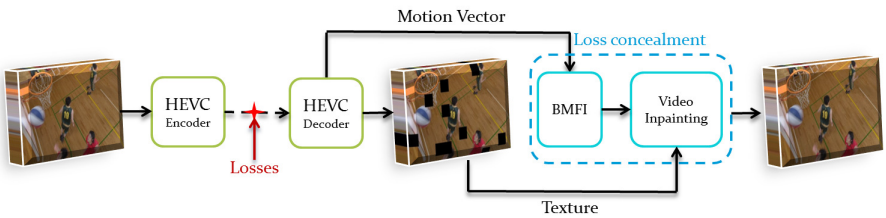


Figure 3.8: **Overview of the communication chain with the HEVC coder/decoder followed by the loss concealment.**

3.4.1.1 Image partitions

Each Intra or Inter-coded image is partitioned into blocks called coding units (CU). The coding unit size can vary from 8×8 up to 64×64. Each CU has an associated partitioning into prediction units (PUs) and a tree of transform

units (TUs). The partitioning is defined by a quadtree structure which is signaled to the decoder. A frame can in addition be partitioned into a number of independent tiles. Tiles are rectangular groups of Largest Coding Unit (LCUs). All dependencies are broken at tile boundaries, so there can be no pixel or motion vector prediction across them.
The motion vectors and the coding mode chosen for each block are transmitted to the decoder. The residue of the intra- or inter-picture prediction is then transformed by a linear spatial transform (DCT), quantized and entropy coded. Hence, the motion information of the video blocks which are not lost are known at the decoder side and can be used by the concealment module.

3.4.1.2 Prediction modes

In the same way as H.264/AVC, each PU partition is built with uni-directional or bi-prediction motion compensation, using $\frac{1}{4}$ (luma) or $\frac{1}{8}$ (chroma) pel precision motion vectors. But the motion vector (MV) values are predicted using a competition between motion vectors of neighboring PU. Two motion prediction modes have been defined: Merge and AMVP (Advanced Motion Vector Prediction). For each prediction unit (PU) of a frame, the encoder decides between these two modes and signals its choice in the bitstream by transmitting an index which is set to 1 for signaling the merge mode and to 0 for the AMVP mode. For each mode, two lists of MV candidates are constructed, one list L_0 referring to candidates in past frames, and one list L_1 referring to candidates in next frames. For both modes, an index corresponding to the best candidate in the constructed list of (spatial and temporal co-located) MV predictors is coded. In the AMVP mode, a residual MV is also transmitted to be added to the predictor candidate. The skip mode is treated as a special case of the merge mode because only a skip flag and the corresponding merge index are transmitted (no information of reference picture list or reference picture index).

If the skip flag is equal to 1, the decoder constructs the list of merging predictor candidates and selects the predictor corresponding to the received merge index. If the skip flag is equal to 0, the decoder decodes the motion information for every PU. If the merge index is equal to 1 (merge mode), the process for retrieving the motion vector is similar to the one of the skip mode. If the merge index is equal to 0 (AMVP), the first step is to retrieve the reference picture for each PU of lists L_0, L_1 or both. The next steps consist in constructing the list of AMVP predictor candidates from which the best predictor is retrieved to be added to the received residual MV.

3.4.1.3 Information used by the loss concealment method

The received motion vectors will be used by the loss concealment module for both computing similarity between patches and estimating by interpolation the motion information of lost blocks. To perform the motion field interpolation, the loss concealment module needs to retrieve from the decoder all data defining the motion vectors of neighboring PU. The first information needed is the mode (INTRA, INTER) and the size of the neighboring PU to know whether motion vectors are available for neighboring pixels. The horizontal and vertical components of the motion vectors, the corresponding reference picture list and the reference pictures (of list L_0 or L_1, or of both) to which the motion vectors point are then retrieved. This information is used to derive a motion vector for each pixel in the neighborhood of the lost blocks. In the case where for a given PU two motion vectors are available, e.g. pointing to reference pictures from list L_0 and L_1, the motion vector pointing to the closest reference frame is retained.

3.4.2 Motion recovery using space-time interpolation

Once the motion vectors (MV)s of correctly received blocks are extracted, MVs of lost blocks are first estimated using MVs of their neighboring blocks. For this aim, we consider the Bilinear Motion Field Interpolation (BMFI) which is known to produce accurate motion estimation for different types of motion [16]. This method illustrated in Figure 3.9, estimates the motion vector of each pixel $p(x, y)$ in the lost or corrupted block B_c using its coordinates in the block and a bilinear interpolation of MV of the left, right, top and bottom (respectively V_L, V_R, V_T and V_B) neighboring blocks. The motion vector $V(x, y)$ of the pixel $p(x, y)$ is computed as follows:

$$V(x, y) = \frac{1}{2}((1 - x_n)V_L + x_n V_R + (1 - y_n)V_T + y_n V_B) \tag{3.6}$$

where $x_n = \frac{x - x_L}{x_R - x_L}$ and $y_n = \frac{y - y_T}{y_B - y_T}$. x_L and x_R are respectively the x-coordinates of the left and right borders of B_c and y_T and y_B are respectively the y-coordinates of the top and bottom borders. The motion data are then considered by the examplar-based video inpainting algorithm. This one uses both texture similarities and motion information to select the best matching patches in the correctly received parts of the video sequence which helps to provide better results than a simple motion-compensation interpolation which may lead to severe visually artifacts due to the propagation of motion estimation errors.

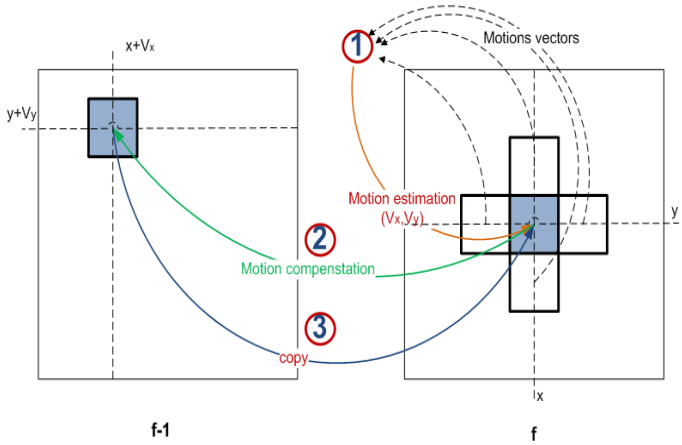


Figure 3.9: **Bilinear Motion Field Interpolation estimation (BMFI).**

3.5 Experimental results

In this section, we evaluate the contributions listed above into the object removal and error concealment applications.

3.5.1 Motion vectors recovery

Let us first evaluate the performance of the lost MV estimation with BMFI in comparison to the ideal case where the motion vectors are perfectly known. Figure 3.10 shows the PSNR values per frame for the two conditions with random losses on blocks of size 16×16. One can notice that there is a very small difference in term of PSNR between these two conditions. A slight PSNR drop at the position of the Intra coded frames where no motion information is provided by the HEVC decoder. In these cases only steps 2 and 3 of the video inpainting algorithm are applied. It implies that the moving objects are sometimes filled by patches of the background pixels.

3.5.2 Neighbor embedding techniques

Regarding the sampling step of the missing pixels, our experiments show that in average, the selection rate is about 75% for LLE and 25% for NMF. Note that TM is almost never selected. So, as illustrated in Figure 3.11, the estimation using neighbors embedding techniques (LLE and NMF) is usually better than the typical TM for video inpainting.

In the following, we compare the proposed sampling approach using motion-compensated window to existing methods in object removal and error conceal-

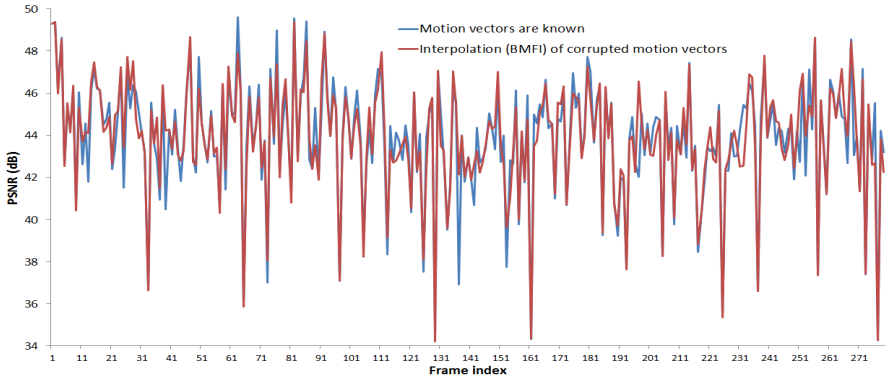


Figure 3.10: **Performances of the BMFI method.** Evolution of the quality (in dB) of the video sequence BasketBallDrill inpainted in native resolution with a loss rate of 5%. Two conditions are illustrated: in the first one (blue curve) all motion vectors are known whereas the motion vectors are interpolated by the BMFI method in the second case (Red curve).

ment applications.

3.5.2.1 Object removal

Figure 3.12 shows four images of the input sequence to be inpainted, as well as results of the inpainting when using the original algorithm described in [8] (Figure 3.12(b)). It also shows in Figure 3.12(c) and 3.12(d) the images obtained after the first step and the three steps of the proposed algorithm. One can notice that the proposed approach using a motion-compensated window and a competition between TM, LLE and NMF gives better results than the single template matching.

3.5.2.2 Error concealment

In a context of error concealment, the proposed approach is compared to state-of-the-art methods [2,12,13]. Figure 3.13 illustrates three images results of different methods. The first column shows the original images while the second column represents the corresponding corrupted images where 20% of blocks of 16×16 pixels are lost. Results of concealment simply using a motion-compensation method [12], or using only spatial inpainting algorithm described in [2] present a lot of artifacts. One can observe, in the last column, that these artifacts do not appear in images obtained using the proposed approach. Moving object as well as static background are correctly recovered.

Also in the case of heavy loss (50% of lost blocks) shown in Figure 3.15, the proposed technique performs better than the motion compensation method.

Figure 3.11: **Samples of patch estimation using TM and neighbor embedding techniques.**

Besides, the PSNR values summarized in Table 3.1 and obtained using each of the listed methods for the two test sequences with different percentage of lost blocks. We notice an average gain of about $2dB$ of the proposed approach compared to state-of-the-art methods.

Finally, the comparison with the spatio-temporal selective extrapolation method presented in [13] illustrated in Figure 3.14 shows that our method provides more natural looking results.

Figure 3.12: **Impact of the neighbor embedding technique on inpainting results.** Sequence Man. (a) Original image + mask. (b) Moving object inpainted with the TM-based algorithm in [8]. (c) Moving object inpainted with the proposed approach. (d) Inpainted result with the proposed approach.

3.5.3 Multi-resolution approach

In this section we focus on the proposed multi-resolution video inpainting approach. We validate this method in a context of loss concealment application performed at the HEVC decoder side. Two different set-ups are considered. The first set-up assumes the video sequence to be encoded in one layer with an Intra frame refreshment period of 32. All frames are impacted by random losses of blocks of different sizes (16×16 and 64×64). In the case of Intra frames, the

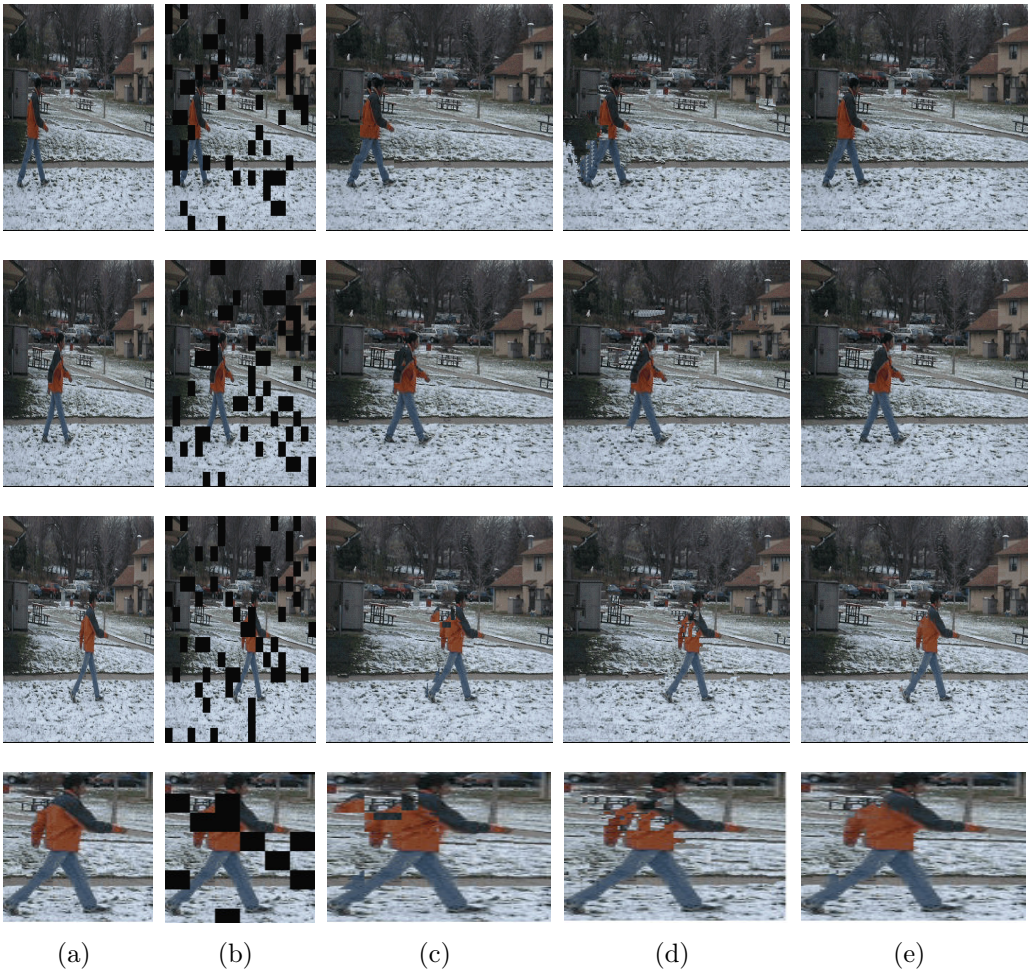
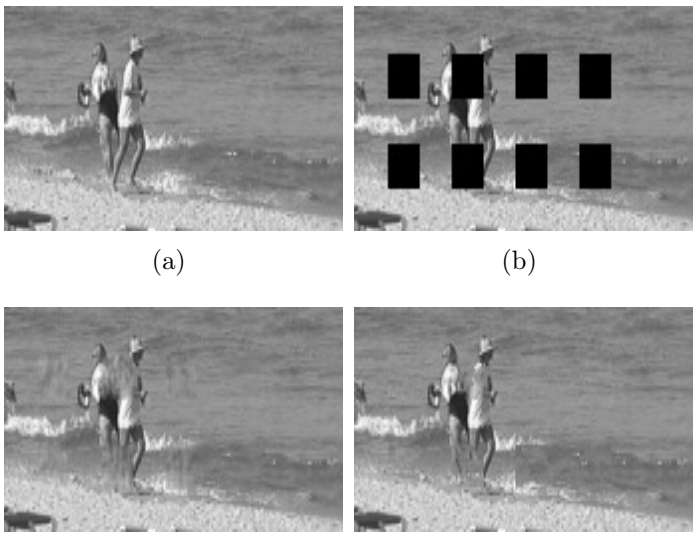


Figure 3.13: **Comparison of different methods for recovering corrupted video with 20% of loss.** (a) Original images. (b) Corrupted images, PSNR=14.22 dB. (c) Recovered images with motion-compensation, PSNR=31.23 dB. (d) Images inpainted with the TM-based algorithm of [2], PSNR=23.41 dB. (e) Images inpainted with video inpainting [8, 71] and a motion estimation step, PSNR=34.96 dB.

motion vectors are assumed to be $(0, 0)$, hence only step 2 (temporal inpainting of static background) and step 3 (spatial inpainting) of the algorithm are performed.

The second set-up considers the case where the hierarchical temporal prediction structure of HEVC is used (B-hierarchical mode). In this case, the base layer (which contains one out of 2 frames) is assumed to be received perfectly whereas the enhancement layer is impacted by random losses of HEVC tiles of size (64×64) with varying loss rates.



(a) (b)

(c) (d)

Figure 3.14: **Performance comparison of the proposed approach with the state-of-the-art method in a context of error concealment.** (a) Original image. (b) Corrupted image, PSNR= 13.2 dB. (c) Inpainting result using the algorithm in [13], PSNR= 20.59 dB. (d) Inpainting result using the proposed algorithm, PSNR= 20.65 dB.

(a) (b) (c)

Figure 3.15: **Performances of the proposed approach in the error concealment application.** Sequence Duo. (a) Image with 50% of 16×16 blocks are lost (first column), PSNR= 9.01 dB. (b) Image recovered using motion compensation (second column), PSNR= 28.26 dB. (c) Image recovered using the proposed algorithm (third column), PSNR= 32.23 dB.

3.5.3.1 Quality-complexity trade-off of the multi-resolution approach

Let we first discuss the performances of the proposed multi-resolution approach in terms of quality and complexity. In these tests we consider two video sequences BasketBallDrill and BQMall (Figure 3.16) which contain respectively

Video Sequence	Percentage of lost blocks	Motion compensation	Spatial inpainting	Proposed method
Man	20	31.47	23.45	34.17
Man	50	27.27	19.37	29.27
Duo	20	30.74	23.84	33.39
Duo	50	27.13	19.90	28.84

Table 3.1: **PSNR values of concealed videos presented in Figure 3.13 and 3.15.**

500 and 200 frames. Tables 3.2, 3.3, 3.4, and 3.5 show that a significant saving of execution time can be achieved by using the two-steps multi-resolution approach instead of the one-step native resolution approach. The execution time saving is obviously higher for a scale factor of $\frac{1}{4}$. When the loss rate is high, the execution time can be reduced by approximately 50%. This time saving is achieved at the expense of some PSNR loss, especially for the scale factor of $\frac{1}{4}$. However, the quality of the inpainted video remains superior to the one obtained with spatial inpainting and motion compensation, when using a scale factor of $\frac{1}{2}$. However, in the case of images with a lot of details (as in BQMall), when using the scale factor of $\frac{1}{4}$, too many high-frequency details are sacrificed which are hard to recover with the super-resolution step.

When comparing the simple interpolation to the NNF-based method for recovering the native resolution, one can observe in Tables 3.2, 3.3, 3.4, and 3.5 that the simple interpolation performs well in terms of PSNR. However, it introduces visually annoying blurring effect on the inpainted regions as illustrated by Figure 3.17. The inpainted videos using the NNF-based super-resolution steps turn out to be of better visual quality than a simple interpolation.

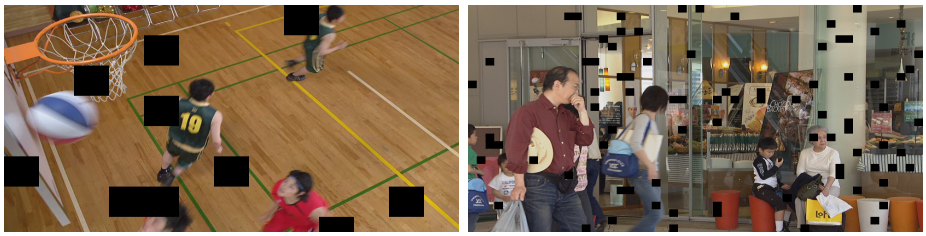
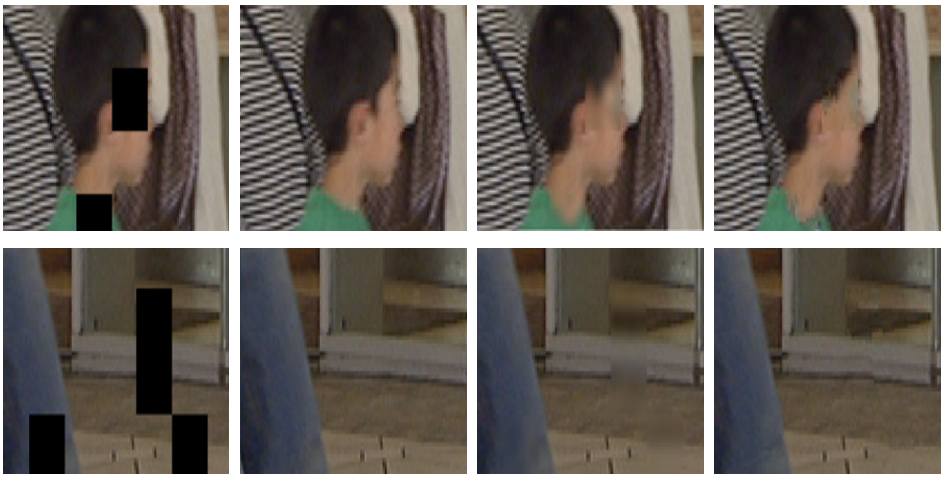


Figure 3.16: **Test video sequences:** BasketBallDrill and BQMall of resolution 832×480 and the frame rate is 50 fps with a loss rate of 5% and blocks of sizes 64×64 and 16×16.



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

Figure 3.17: **Zoom on 4 blocks inpainted with the different methods.** (a) Image with corrupted blocks. (b) Inpainting at the native resolution. (c) Inpainting of the low-resolution frame followed by a simple interpolation. (d) Inpainting using the NNF-based super-resolution.

Loss Rate		**1%**	**5%**	**10%**
Known motion vectors for lost blocks (without BMFI)	**Native**	50.04	42.59	39.12
	Interp.	46.23	38.81	35.45
	NNF	46.40	38.98	35.61
Estimation of motion vectors for lost blocks (with BMFI)	**Native**	50.25	42.69	39.22
	Interp.	46.29	38.79	35.44
	NNF	46.39	38.94	35.57
Spatial inpainting		42.66	35.15	31.56
Motion compensation		45.53	37.94	34.51
Execution time (s)	**Native**	1,30	4,47	9,5
	Interp.	0,22	0,24	0,28
	NNF	0,81	2,75	5,34
	Spatial	1,26	3,10	7,07
	Comp.	0,23	0,25	0,27

Table 3.2: PSNR and computational times obtained by the video inpainting algorithm for BasketBallDrill, assuming the motion vectors to be known, or estimated with BMFI. The scale factor for the interpolation with a Lanczos filter and with NNF is 0.25 (832×480 to 416×240).

3.5.3.2 HEVC hierarchical temporal structure

In this second test, the hierarchical B frames structure of HEVC is used and, in this case, the losses occur on HEVC bitstream segments corresponding to tiles of sizes 64×64 of B-frames of the enhancement layer only. The Intra-refreshment period is 32. Tables 3.6 and 3.7 show the PSNR results and execution times obtained with the different inpainting methods in the case where the motion vectors

Loss Rate		1%	5%	10%
Known motion vectors for lost blocks (without BMFI)	**Native**	50.04	42.59	39.12
	Interp.	48.29	40.88	37.42
	NNF	48.16	40.78	37.34
Estimation of motion vectors for lost blocks (with BMFI)	**Native**	50.25	42.69	39.22
	Interp.	48,29	40.80	37.42
	NNF	48.17	40.72	37.32
Spatial inpainting		42.66	35.15	31.56
Motion compensation		45.53	37.94	34.51
Execution time (s)	**Native**	1,3	4,47	9,5
	Interp.	0,33	0,63	1,03
	NNF	0,92	3,14	6,08
	Spatial	1,26	3,1	7,07
	Comp.	0,23	0,25	0,27

Table 3.3: PSNR and computational times obtained by the video inpainting algorithm for BasketBallDrill, assuming the motion vectors to be known, or estimated with BMFI. The scale factor for the interpolation using the lanczos filter and with NNF is 0.5 (832×480 to 416×240).

Loss Rate		1%	5%	10%
Known motion vectors for lost blocks (without BMFI)	**Native**	48.93	41.95	38.33
	Interp.	44.20	37.29	34.11
	NNF	44.01	36.97	33.81
Estimation of motion vectors for lost blocks (with BMFI)	**Native**	48.40	41.68	38.21
	Interp.	44.08	37.16	34.03
	NNF	43.83	36.89	33.72
Spatial inpainting		40.43	33.03	29.76
Motion compensation		45.05	38.05	34.65
Execution time (s)	**Native**	1,80	6,00	11,03
	Interp.	0,19	0,23	0,27
	NNF	0,76	2,53	7,86
	Spatial	1,25	3,03	7,86
	Comp.	0,23	0,25	0,27

Table 3.4: PSNR and computational times obtained by the video inpainting algorithm for BQMall, assuming the motion vectors to be known, or estimated with BMFI. The scale factor for the interpolation with the Lanczos filter and with NNF is 0.25.

are not estimated using BMFI (they are supposed to be NULL) and in the case where the motion vectors are estimated with the BMFI technique. One can see in both tables that inpainting at the native resolution again leads to the highest quality. But, the execution time of the inpainting process can be significantly reduced by using the two steps approach at the expense of a negligible loss in terms of PSNR and visual quality. The PSNR values (and the visual quality) obtained with the proposed methods are significantly higher that the one given by a spatial inpainting. The inpainting algorithm at native resolution outperforms loss concealment using motion compensation. The two steps approach (using either interpolation or NNF-based super-resolution) outperforms motion compensation for a scale factor of 2 between the two levels. For a scale factor of 4, there is

Loss Rate		**1%**	**5%**	**10%**
Known motion vectors for lost blocks (without BMFI)	**Native**	48.93	41.95	38.33
	Interp.	46.05	39.10	35.80
	NNF	45.73	38.65	35.43
Estimation of motion vectors for lost blocks (with BMFI)	**Native**	48.40	41.68	38.21
	Interp.	45.60	38.84	35.53
	NNF	45.31	38.49	35.22
Spatial inpainting		40.43	33.03	29.76
Motion compensation		45.05	38.05	34.65
Execution time(s)	**Native**	1,80	6,00	11,03
	Interp.	0,33	0,70	1,18
	NNF	0,88	2,99	5,5
	Spatial	1,25	3,03	7,86
	Comp.	0,23	0,25	0,27

Table 3.5: PSNR and computational times obtained by the video inpainting algorithm for BQMall, assuming the motion vectors to be known, or estimated with BMFI. The scale factor for the interpolation using the lanczos filter and with NNF is 0.5.

still some good PSNR performance gains for smaller loss rates (between 0.5% and 2.5%). However, for higher loss rates, the two steps approach loses high frequency details in a higher number of blocks, and these high frequency details are not sufficiently well recovered by the interpolation or NNF-based super-resolution, which leads to a PSNR loss. However, the inpainted videos turn out to be more visually pleasing, they are smoother and have less visible blocking artifacts that when using motion compensation.

Loss Rate		**1%**	**5%**	**10%**	**15%**	**20%**
Known motion vectors for lost blocks (without BMFI)	**Native**	49.12	41.87	38.71	36.95	32.97
	Interp.	47.45	40.36	37.16	35.38	32.54
	NNF	47.38	40.35	37.13	35.36	32.62
Estimation of motion vectors for lost blocks (with BMFI)	**Native**	49.31	42.49	39.27	37.18	36.12
	Interp.	48.01	40.74	37.39	35.52	34.5
	NNF	47.88	40.67	37.30	35.45	34.42
Spatial inpainting		42.46	35.37	32.02	30.27	29.14
Motion compensation		47.73	40.72	37.79	35.60	34.85
Execution time(s)	**Native**	1,08	2,87	3,97	5,55	7,68
	Interp.	0,20	0,21	0,22	0,23	0,26
	NNF	0,52	1,38	2,56	3,52	4,57
	Spatial	0,56	1,50	2,54	3,77	4,52
	Comp.	0,21	0,22	0,23	0,24	0,24

Table 3.6: PSNR and computational times of the three methods with HEVC hierarchical B frames and tiles of size 64×64: inpainting of input video at native high-resolution (Native), with the two-steps approach using interpolation (lanczos filter) or using the NNF-based super-resolution method (NNF). The scale factor for the interpolation and NNF is 0.25.

Loss Rate		1%	5%	10%	15%	20%
Known motion vectors for lost blocks (without BMFI)	**Native**	49.12	41.87	38.71	37.89	37.90
	Interp.	48.27	41.12	36.95	36.14	36.21
	NNF	48.27	41.19	32.97	32.63	32.80
Estimation of motion vectors for lost blocks (with BMFI)	**Native**	49.31	42.49	39.27	38.17	38.20
	Interp.	48.27	41.48	37.18	36.14	36.20
	NNF	48.26	41.51	36.12	35.09	35.14
Spatial inpainting		42.46	35.37	32.02	30.27	29.14
Motion compensation		47.73	40.72	37.79	35.60	34.85
Execution time(s)	**Native**	1,08	2,87	3,97	5,55	7,68
	Interp.	0,25	0,39	0,56	0,72	0,94
	NNF	0,58	1,61	2,93	4,00	6,65
	Spatial	0,56	1,50	2,54	3,77	4,52
	Comp.	0,21	0,22	0,23	0,24	0,24

Table 3.7: PSNR and computational times of the three methods with HEVC hierarchical B frames and tiles of size 64×64: inpainting of input video at native high-resolution (Native), with the two-steps approach using interpolation (lanczos filter) or using the NNF-based super-resolution method (NNF). The scale factor for the interpolation and NNF is 0.5.

3.6 Conclusion

This chapter described several extensions of examplar-based video inpainting algorithm. A patches sampling approach based on neighbor embedding techniques has been first proposed. This technique provides better patch approximation by considering for each target patch, the best approximation method among TM, LLE, and NMF. Then, the proposed inpainting approach has been validated into both object removal and error concealment applications by using a preprocessing step of motion estimation. This step helps to provide improved performances compared to using a simple motion compensation of blocks from previous frames by limiting error propagation caused by the uncertainties on the estimated motion data. Finally, a two-steps multi-resolution extension has also been used for decreasing the execution time. The second step of this approach is inspired from single image super-resolution techniques, but is here based on an iterative approximate nearest neighbor field computation technique. The assessment in a HEVC-based communication chain showed the good performance of the proposed video inpainting for loss concealment at the native resolution as well as the trade-off between quality and execution time achieved by the multi-resolution method.

However, some artifacts still appear in the inpainted video especially for heavy loss. This is due to the errors in the estimation of the motion vectors corresponding to the lost blocks. The accuracy of the motion estimation approach (BMFI) is highly dependent of the availability of the motion vectors of neighboring blocks. With high loss rate, the errors in motion vectors estimation increase. This leads to consider some lost moving pixels as background pixels and vice versa.

In the next Chapter we will focus on another parameter of the examplar-based inpainting algorithm which is the patches matching computation.

Chapter 4

Patches matching metrics

This chapter[1] focus on the patches matching step of the video inpainting algorithm. This step is usually performed assuming that any patch in the image or video content can be approximated with one or more patches in the same image/video. For instance, the pixel values of the target patch can be estimated with a linear combination of the corresponding pixels in its most similar patches. The results of examplar-based algorithms are then highly dependent on the similarity metric used to compute the degree of matching between patches.

In this chapter, we present a performance analysis of the most used patch similarity measures. This analysis is first performed by a subjective test showing the perceptual results of patches matching provided by a set of observers. Then, objective quality scores computed with a set of the most used similarity measures is computed to find the best correlated metric with the subjective tests. Finally, these measures are used with different examplar-based algorithms to find the similarity metric providing the best quality results.

4.1 Similarity metrics

Let we start our analysis by presenting, in this section, an overview of the most used similarity metrics in image processing algorithms. Almost all of these metrics provide a similarity score between two candidates. However, each of them gives more importance to some image features than others which makes different their similarity decisions. The considered metrics can be broadly classified into three main categories: pixel-based, statistic-based and structure-based similarity metrics.

1. The content of this chapter is related to our publication in [75]

4.1.1 Pixel-based similarity metric

Pixel-based metrics compute the similarity between two patches/images using the differences between their pixel values. L_p-norm and SSIM are the most widely used metrics of this category to compute the performance of an image processing algorithm with comparing the result of the algorithm and the original images.

4.1.1.1 L_p-norm

The L_p-norm or p-norm of a given signal $x = (x_1, x_2, \ldots, x_n) \in R^n$, $p \geq 1$, is defined as: $\|x\|_p = (\sum_{i=1}^{n} |x_i|^p)^{\frac{1}{p}}$.
In practice, three main values of p correspond to the three most used norms:

- L_1-norm, also called Manhattan metric: $\|x\|_1 = \sum_{i=1}^{n} |x_i|$.
- L_2-norm, or Euclidean norm: $\|x\|_2 = (\sum_{i=1}^{n} |x_i|^2)^{\frac{1}{2}}$.
- L_∞-norm, or Chebyshev norm: $\|x\|_\infty = \max_{1 \leq i \leq n} |x_i|$.

The L_2-norm is the simplest and most used fidelity metric in the literature of image processing. It computes the similarity of two given patches x and y as the sum of squared intensity differences (SSD) of their pixels values:
$SSD(x, y) = \sum_{i=1}^{n} (x_i - y_i)^2$. MSE, which is the mean of SSD, is usually converted into a peak signal-to-noise ratio (PSNR) measure for assessing degradation quality of a noisy image compared to its original version [76] as:
$PSNR = 10log_{10}\frac{L^2}{MSE}$. L is the range of pixel intensity (255 for gray-scale image).

This metric presents many advantages: symmetry, convexity and energy preserving property. However, it does not match the perceived visual quality. The similarity between two patches is given by the average similarity of their collocated pixels. Therefore, two patches x and y having the same degree of similarities with a given patch z (i.e. $SSD(x, z) = SSD(y, z)$), may be visually dramatically different.

4.1.1.2 Structural Similarity Information Measure (SSIM)

The Structural Similarity Information Measure (SSIM) computes the similarity between two images or patches x and y based on three terms:

- luminance: $l(x, y) = \frac{2\mu_x\mu_y + C_1}{\mu_x^2 + \mu_y^2 + C_1}$
- contrast: $c(x, y) = \frac{2\sigma_x\sigma_y + C_2}{\sigma_x^2 + \sigma_y^2 + C_2}$
- structure: $s(x, y) = \frac{\sigma_{xy} + C_3}{\sigma_x\sigma_y + C_3}$

where μ_x is the mean of the pixel values in x and σ_x is the standard deviation. The constants C_1 and C_2 ($C_3 = \frac{C_2}{2}$) are given by $C_1 = (K_1 L)^2$, $K_1 \prec\prec 1$, $C_2 = (K_2 L)^2$, $K_2 \prec\prec 1$ and $L = 255$ for gray scale images.
Then, $SSIM(x, y) = [l(x, y)]^\alpha \cdot [c(x, y)]^\beta \cdot [s(x, y)]^\gamma$. Here α, β and γ are parameters used to adjust the relative importance of each of the three components [77]. A specific form of the SSIM index where $\alpha = \beta = \gamma = 1$ is usually used:

$$SSIM(x, y) = \frac{(2\mu_x\mu_y + C_1)(2\sigma_{xy} + C_2)}{(\mu_x^2\mu_y^2 + C_1)(\sigma_x^2\sigma_y^2 + C_2)}$$

4.1.2 Statistic-based similarity metrics

Similarity between two patches can be also computed based on their statistics. These metrics provide a high degree of similarity when the probability density functions (pdf) of the patches are close. A review of classical statistic-based metrics can be found in [78]. The pdf of a patch x is usually given by $p(x) = \frac{h(x)}{n}$, where $h(x)$ is the histogram of the pixel values and n the number of pixels in the patch. This histogram represents the frequency of each interval of values (so-called bin), providing the basis for a non-parametric estimation of the pdf. In our analysis, we consider three main Statistic-based measures namely NMI (Normalized Mutual Information) and KL-Div (Kullback-Leibler Divergence) and Bhattacharyya probability density [37, 38].

4.1.2.1 Normalized Mutual Information (NMI)

The mutual information $I(X; Y)$ of two discrete random variables X and Y computes the amount of common information between them as:

$$I(X; Y) = \sum_{x_i \in X} \sum_{y_i \in Y} p(x_i, y_i) \log(\frac{p(x_i, y_i)}{p(x_i)p(y_i)})$$

where $p(x_i)$ the occurrence of the probability of x_i. Usually, the normalized form of the mutual information is more used:

$$NMI(X, Y) = \frac{I(X; Y)}{\sqrt{H(X) \times H(Y)}}$$

$H(X)$ and $H(Y)$ are respectively the entropy of X and Y defined as:

$$H(X) = -\sum_{x_i \in X} p(x_i) \log(p(x_i))$$

The NMI metric is bounded between 0 (if X and Y are independents) and 1 (if their pdf are exactly the same).

4.1.2.2 Kullback-Leibler Divergence (KL-Div)

The Kullback-Leibler divergence (KL-Div) of two probability distributions X and Y is defined as:

$$D_{KL}(X \,||\, Y) = H(X,Y) - H(X) \text{ where } H(X,Y) = -\sum_{x_i \in X, y_i \in Y} p(x_i, y_i) \log\left(p(x_i, y_i)\right)$$

can also be used to compute the degree of similarity between two patches. This non-symmetric metric is not upper-bounded but null when the distributions of the two patches are exactly the same.

4.1.3 Hybrid distances: Bhattacharyya probability density based metric

NMI and KL-Div metrics measure the similarities between two samples assuming that their closeness is only dependent on their probability distributions. As illustrated in Figure 4.1, this constraint is not enough. The two patches are different although their probability distributions is the same. A combination be-

(a) x (b) y

Figure 4.1: **Performances of the probability-based metrics.** Samples of two patches of 21×21 pixels estimated as similar using the probability-based metrics: $d_{KL-Div}(x,y) = 0$ and $d_{NMI}(x,y) = 1$ and quite different using SSD ($d_{SSD}(x,y) = 5355$).

tween the probability distribution and SSD proposed in [37] helps to deal with this drawback. This metric is given by:

$$d(x,y) = \frac{1}{|x|} \times SSD(x,y) \times d_B(x,y) \tag{4.1}$$

where $d_B(x,y) = \sqrt{1 - \sum_{i=1}^{B} \sqrt{p_x(i) \times p_y(i)}}$ is the Bhattacharyya coefficient. p_x and p_y are the pdf of patches x and y respectively. However, if the two patches have the same distribution the Bhattacharyya coefficient d_B is null no matter for the rotation. To be able to distinguish this case, the distance is computed using the following formula [38]:

$$d(x,y) = \frac{1}{|x|} \times SSD(x,y) \times (1 + d_B(x,y)) \tag{4.2}$$

In the rest of the chapter we notice Bh_1 the metric distance in equation 4.1 and Bh_2 the metric in equation 4.2.

4.1.4 Dominant orientation Template

Similarity metrics can also be based on the image structures of the considered patches as shown in [79–81]. For instance, authors in [80] show that the dominant orientation template (DoT) approach which is related to the Histograms-of-Gradients representation can be efficiently used to compute the degree of matching between two given patches. The basic idea of this metric is that the two patches are considered similar if the magnitude orientations of their dominant gradients are similar. Although, this measure does not take into account the difference of texture values, it has the advantage to be robust to noise and illumination change.

The dominants gradient magnitude in each patch are first discretized to a small set of dominants orientations values to be robust to the small deformations. Then, the orientation magnitudes in the patches are compared to evaluate their similarity. We consider here 7 dicretized orientation values. If the gradient magnitude is lower than a fixed threshold, the patch is considered uniform.

4.1.5 Similarity metrics characteristics

In summary, each one of the aforementioned similarity metric quantify the degree of resemblance between patches or images with giving more importance to an image characteristic than another. Besides, they are sensitive to some image criteria making their similarity decisions different. Table 4.1 summarized the main characteristics of each measure. For instance, the DoT is less sensitive to noise and orientation variation, although, it can give wrong similarity results for uniform patches since it does not take into account the difference between collocated pixels values. Two patches with similar structures may be visually quite different. In another hand, Bh_2 seems to have better similarity decision since it makes advantages of both patches statistics and texture similarity provided by the SSD.

Sensitivity	**SSD**	**SSIM**	**NMI**	**KL-Div**	Bh_1	Bh_2	**DoT**
Noise	Medium	Low	Medium	Medium	Medium	Medium	very low
Rotation	Medium	Medium	High	High	High	Low	Medium
Structure	Medium	Low	Medium	Medium	Medium	Medium	very low
Texture	Low	Low	Medium	Medium	Low	Low	High

Table 4.1: Similarity metrics sensitivity.

4.2 Subjective similarity decisions

In order to provide a subjective performance analysis of the aforedescribed similarity metrics, our experiments are conducted on 100 natural images. As illustrated in Figure 4.2, for each image, two patches (templates) of 21×21 pixels are chosen. Each template patch is compared to a list of 14 candidates composed with its 8 neighbors and 6 patches randomly selected within the same image. The

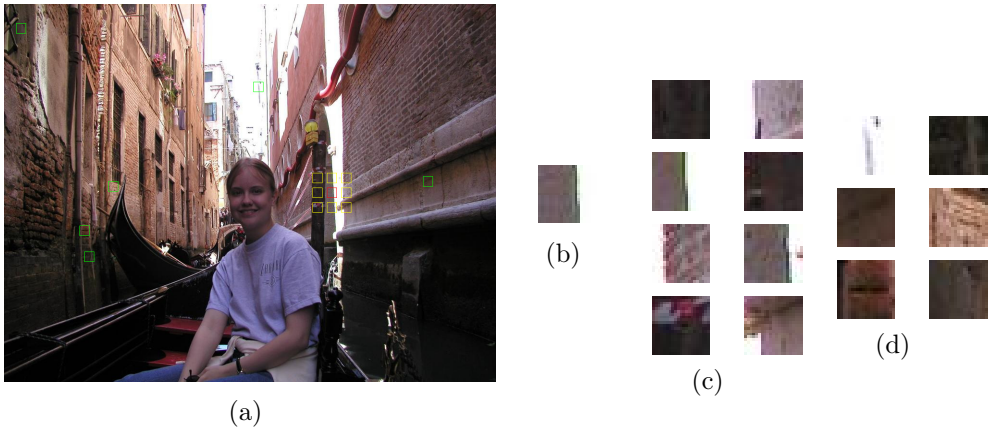


Figure 4.2: **Image test for subjective analysis.** (a) Image test. (b) Patch template represented with red square in a). (c) Selected patch candidates in the neighboring window of the template (represented with yellow squares in a). (d) Patch candidates randomly selected outside the neighboring window of the candidate (green squares in a).

perceptual similarity decisions are then gathered, by using a web page (illustrated in Figure 4.3) representing the patch templates and their list of candidates. 12 observers have been asked to choose, for each template, the three first closest patches.

4.3 Experimental Results

4.3.1 Subjective test

Once the observers' similarity decisions are gathered, the agreement between subjective and objective similarity results is computed in order to find the best fidelity metric that respect the human visual system sensitivity. For this purpose, we compare the order of similarity given by the observers decisions with the one computed using each of the aforementioned fidelity metrics. For each metric m_i,

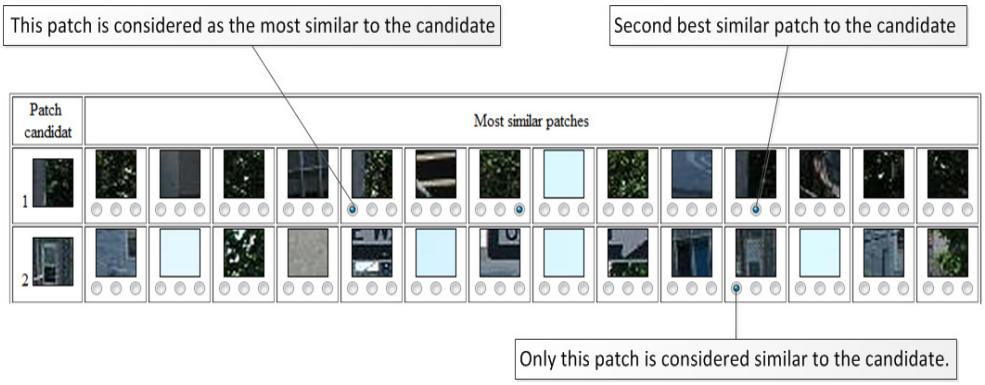


Figure 4.3: **Part of the web page used for gathering observers similarity decisions.** Patch templates (first column). The corresponding list of patch candidates, selected from the same image, for similarity comparison.

this comparison can be represented with the following formula:

$$Corr(m_i) = \frac{\sum_{j=1}^{N} \delta(j)}{N},$$

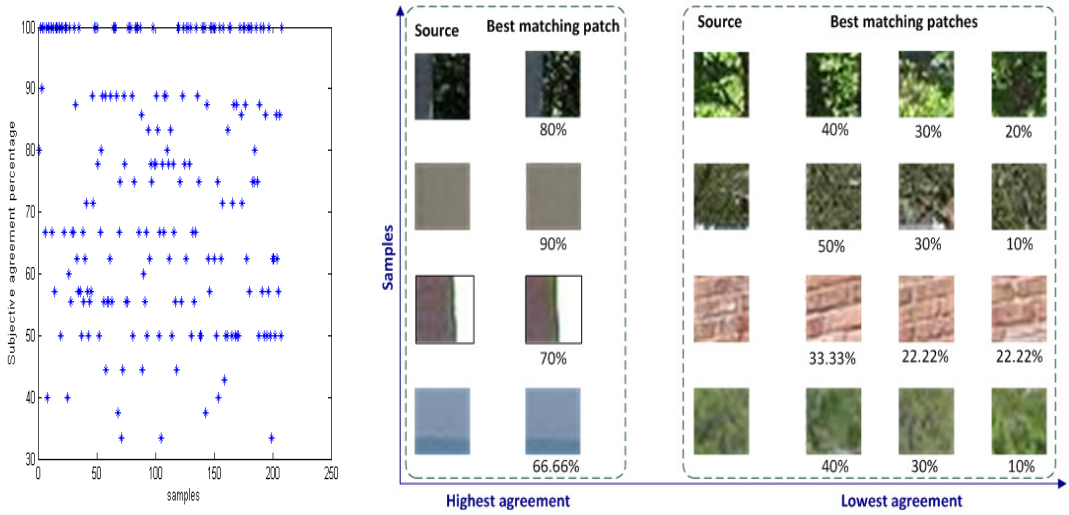
where N is the total number of patch candidates, and

$$\delta(j) = \begin{cases} 1 & \text{if } Sim(j, m_i) = BM(j) \\ 0 & \text{otherwise.} \end{cases}$$

$Sim(j, m_i)$ gives the index of the best similar patch to the candidate Ψ_j according to the considered metric m_i and $BM(j)$ the one found using the subjective test.

For each patch template, the proposed list may contains patches quite different so that the most similar patch to the template can be easily selected. The candidates may also be quite similar, so that the observers can't distinguish between them. Then, their similarity decision about the most similar patch to the template will be distributed on more than one patch as illustrated in Figure 4.4. To deal with this problem, we consider three main constraints for computing the correlation between subjective and metric-based comparison results:

- constraint1: we consider all the subjective decisions where one of the three most similar patches according to observers corresponds to the best matching patch found using the similarity metric.
- constraint2: idem to constraint1 with considering only candidate patches for which more than 50% of the observers have the same decision.
- constraint3: consider only candidates patches for which more than 50% of the observers have the same decision and this patch correspond to the best matching patch to the candidate according to the similarity metric.



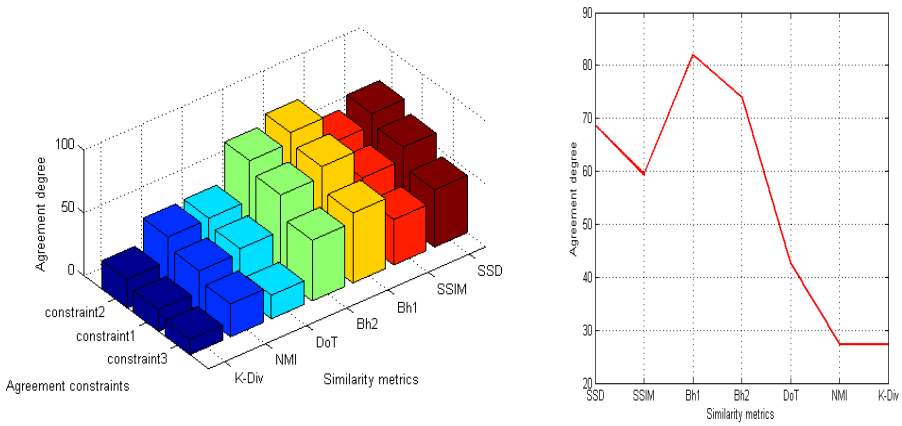
(a) (b)

Figure 4.4: **Agreement of observers' similarity decisions.** (a) Observers agreement. (b) Samples of patches for which observers have almost the same decisions (highest agreement) and others for which observers have different decisions (lowest agreement).

Figure 4.5 illustrates the results of correlation using the three listed constraints. One can mention that, using each of the considered correlation constraint, the similarity metrics based only on the probability distribution i.e. NMI and KL-Div, does not confirm the perceptual results. This conclusion is expected because the probability distribution helps to detect structure of the patch such that making difference between a smooth patch and textured one. But it does not quantify the dissimilarity between the collocated pixels values in the two patches. Although, combining the probability distribution information with the SSD (Bh_1 and Bh_2), is a good compromise to give better similarity decision. We notice also that considering only the dominant gradient magnitude in the patches to compare their similarity can give wrong results since two patches having similar structure may have totally different textures.

4.3.2 Impact of similarity metrics on examplar-based approaches

In this section, we propose to further analyze the impact of the similarity metrics on the results provided by examplar-based algorithms. For this aim, we compare the performances of the four best correlated metrics with the subjective resemblance results, i.e. SSD, SSIM, Bh_1 and Bh_2 using two examplar-based



(a) (b)

Figure 4.5: **Agreement between subjective-based and metrics-based similarity decisions.** (a) Agreement using different constraints for correlation. (b) Agreement corresponding to constraint 2.

algorithms namely the Non Local means (NL-means) denoising approach [70] and Examplar-based inpainting method [2].

4.3.2.1 Non Local means (NL-means) denoising

We first compare the results of Non Local means (NL-means) denoising algorithm proposed in [70] using each of the similarity metrics. This denoising approach considers that each pixel p in the image I can be estimated using a weighted average of collocated pixels values in the most similar patches to the one centered on p (Ψ_p) as: $NL(p) = \sum_{q \in I} \omega(p,q) I(q)$, where $0 \leq \omega(p,q) \leq 1$, is the weight of the pixel q, defined as a function of the similarity distance between the patch centered on q and Ψ_p:

$$\omega(p,q) = \frac{1}{Z(p)} \exp\left(-\frac{\|\Psi_p - \Psi_q\|_2^{\,2}}{h^2}\right)$$

while $Z(p) = \sum_q \exp\left(-\frac{\|\Psi_p - \Psi_q\|_2^{\,2}}{h^2}\right)$ is a normalizing term to validate the constraint $\sum_q \omega(p,q) = 1$. The value of h which controls the decay of the exponential function may have an important impact on the denoising results. So, first we compute, for each similarity metric, the value of h that provides the highest mean of PSNR of the 10 natural images presented in Figure 4.6. These PSNR values illustrated in Figure 4.7 show that the similarity metric Bh_2 usually provides the highest PSNR gain. Also in terms of subjective quality, denoising results

(a) Barbara (512 × 512) (b) boat (512 × 512) (c) C.Man (256 × 256) (d) couple (512 × 512) (e) F.print (512 × 512)

(f) Lena (512 × 512) (g) house (256 × 256) (h) F16 (512 × 512) (i) Peppers (512 × 512) (j) Babon (512 × 512)

Figure 4.6: **Image tests used for image denoising application.**

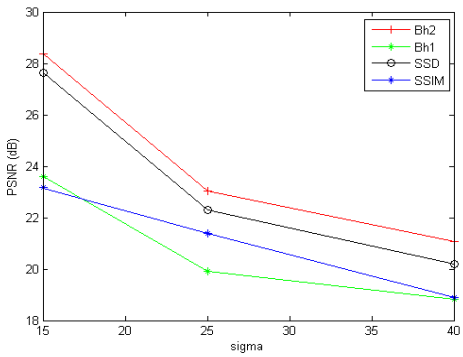


Figure 4.7: **PSNR values of NL-means denoising results using different similarity metrics and the best value of h.**

illustrated in Figure 4.8 show that Bh_2 provides much better noiseless image quality than SSD. The PSNR values of denoising results of standard test images summarized in Table 4.3.2.1 and 4.3 show that even with a higher noise ($\sigma = 40$) this metric provides an average gain of 1.3 dB compared to SSIM, SSD and Bh_1.

4.3.2.2 Inpainting application

The second examplar-based approach concerns the inpainting application. The PSNR values of inpainted results of images illustrated in Figure 4.9 are summurized in Table 4.4. Here also the Bh_2 metric shows improvement in terms

	SSD	SSIM	Bh_1	Bh_2
barbara	23.45	22.32	21.28	**24.89**
boat	22.59	21.73	21.37	**23.88**
C.Man	22.15	21.22	20.09	**23.78**
couple	22.2	21.83	21.14	**23.41**
F.print	19.98	19.91	17.26	**21.61**
Lena	23.7	21.78	23.75	**25.21**
house	24.27	21.83	24.08	**26.62**
F16	25.62	22.47	25.28	**28.03**
Peppers	22.75	20.49	22.74	**23.97**
Babon	19.64	19.76	19.48	**22.19**
Mean	22.63	21.32	21.64	**24.35**

Table 4.2: PSNR values of denoising results of noisy images with $\sigma = 25$.

	SSD	SSIM	Bh_1	Bh_2
barbara	19.74	19.37	18.88	**20.87**
boat	20.01	19.32	19.41	**20.93**
C.Man	18.47	18.26	17.65	**19.5**
couple	19.95	19.48	19.45	**20.8**
F.print	16.82	18.13	15.86	**18.05**
Lena	20.04	19.36	20.15	**21.32**
house	20.11	19.48	20.26	**22.29**
F16	21.08	20.12	21.2	**23.39**
Peppers	19.03	18.17	19.15	**20.42**
Babon	17.76	18.21	17.8	**19.14**
Mean	19.3	18.99	18.98	**20.67**

Table 4.3: PSNR values of denoising results of noisy images with $\sigma = 40$.

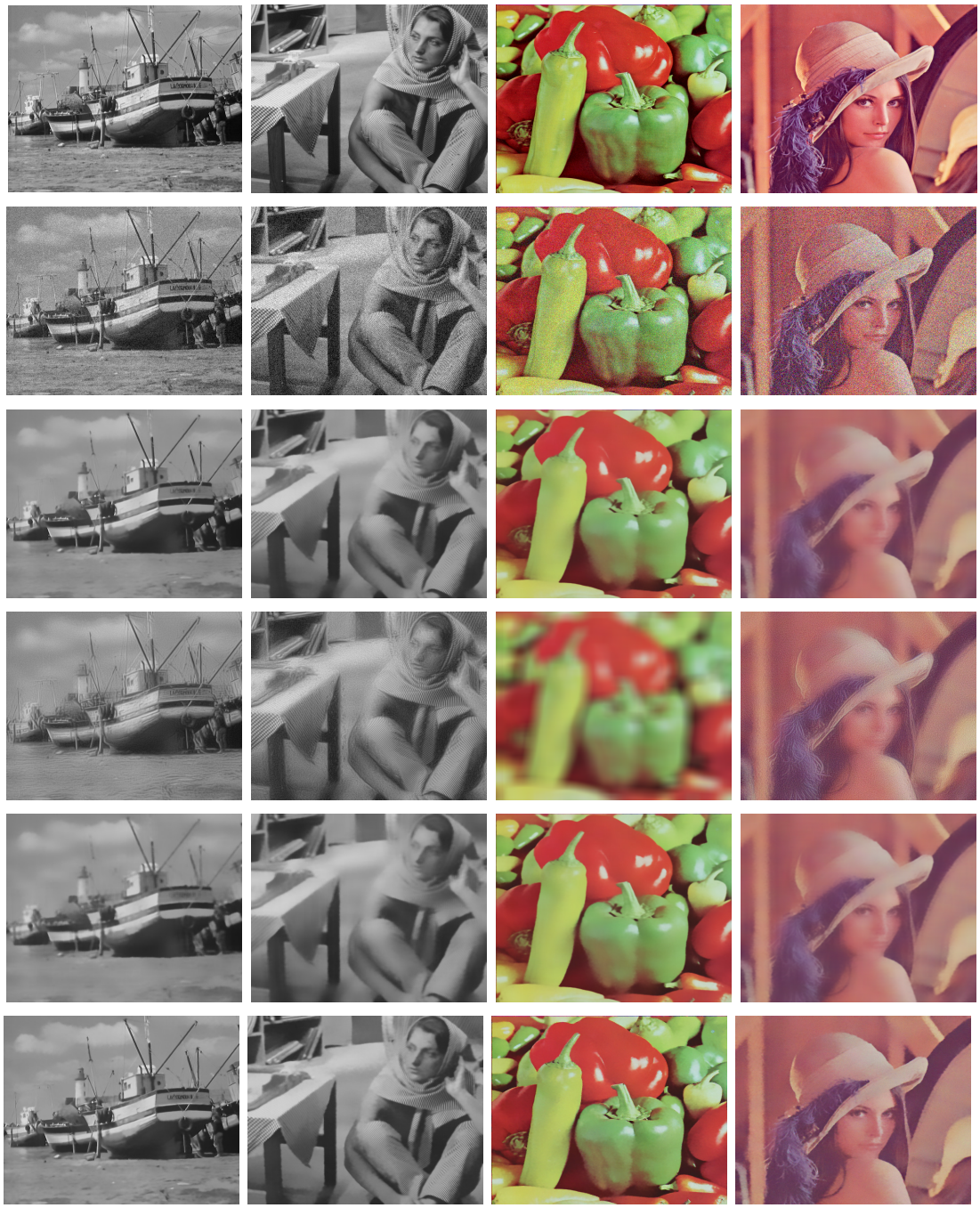
of PSNR of inpainted results compared to the others metrics. One can mention that the difference between the results gathered using SSD and Bh_2 metrics are here lower than in the NLM algorithm. This can be explained by the fact that in the inpainting method in [2] both the priority computation and patch matching has impact in the final results. Therefore, if different metrics are used for inpainting the same image the filling order will not be necessary the same and therefore the patches to be matched in each iteration will not be the also same for the considered metrics.

	SSD	SSIM	Bh_1	Bh_2
barbara	**30.87**	29.95	28.42	30.7
boat	28.06	27.16	27.7	**28.27**
C.Man	23.03	21.55	20.76	**23.16**
couple	29.52	28.03	27.36	**29.73**
F.print	27.11	26.84	25.93	**27.29**
Lena	**26.83**	25.27	26.50	26.40
house	28.13	27.55	27.90	**28.53**
F16	25.51	24.28	25.55	**26.15**
Peppers	**29.87**	27.99	28.5	29.55
Babon	**30.31**	29.07	29.58	31.02
Mean	27.92	26.77	26.82	**28.08**

Table 4.4: PSNR values of inpainted images using the method in [2] (TM, searching window= 50×50).

4.4 Conclusion

This chapter have presented a performance analysis of several similarity metrics that can be used in the patches matching step of the examplar-based algorithms. The subjective results provided by the decision of a group of observers showed that SSD which is a widely used similarity metric is not the best correlated with the perceptual sensitivity. This conclusion is also confirmed with experiments on NL-means denoising and inpainting algorithms. Results using the combination of SSD and the Bhattacharyya coefficient (Bh_2) showed better image quality compared to SSD and SSIM. This combination takes advantage of SSD to compute the signals dissimilarities and helps to make difference between structured and textured patches with using probability distributions.



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

Figure 4.8: **NL-means denoising results using different similarity metrics.** From top to bottom: Original image; Noisy image; Denoised image using SSD; Denoised image using SSIM; Denoised image using Bh_1; Denoised image using Bh_2. (a) Boats ($\sigma = 15$). (b) Barbara ($\sigma = 25$). (c) Pepper ($\sigma = 25$). (d) Lena ($\sigma = 40$).

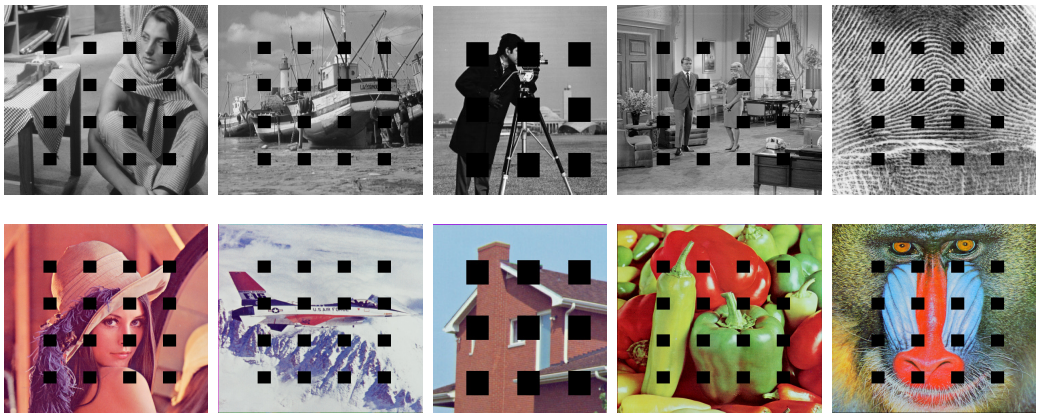


Figure 4.9: **Image tests used for image inpainting application.**

Chapter 5

Graph-based background inpainting

Video inpainting algorithms need to deal with various high resolution sequences with complex textures. One of the crucial problems with these sequences is that the holes to be inpainted are large enough to make having visually pleasing results a challenging task. Examplar-based approaches do not provide high quality of inpainting since the optimization constraint in these methods is computed only according to the patch to be inpainted i.e. the current highest priority patch in each iteration. Thus, the lack of global coherency constraint over all missing pixels in the hole leads to severe artefacts which are more visible with large holes.

Video inpainting approaches which are based on a global optimization function may provide better inpainting quality in the case of large holes and high resolution sequences. These methods named MRF-based, Graph-based or multi-labels inpainting approaches recover holes in a video sequence using a Markov Random Field (MRF). Each pixel in the missing region is inpainted with the pixel value, from the set of candidates, that minimizes the global cost. This cost is computed for all the hole pixels using a well-defined energy function that should expresses the space-time consistency of the inpainted result.

Unfortunately, these methods are time consuming making hard their use in video editing applications. The purpose in this chapter is to propose a video inpainting algorithm based on the global optimization of an energy function ensuring space-time consistency result without being time consuming. To simplify the problem, only the inpainting of missing regions corresponding to background in static camera videos is tackled in this chapter. In video inpainting context, such case occurs when the object to be removed from the scene does not occlude any other moving object. Thus, the missing pixels have to be inpainted with static background pixel values. Another application of this case may be the background estimation. This application aims at estimating the background of

an image sequence with removing all moving objects in the scene. This application differs from the inpainting application only in a preprocessing step where a binary mask corresponding to the moving objects has to be first detected. Hence, background estimation are sometimes called non-parametric video inpainting.

Several forms of energy functions for inpainting the background in a video sequence are reported in the literature. A survey of the well-known state-of-the-art functions is first presented in Section 5.1. Then, we describe in Section 5.2 the proposed energy function and therefore the inpainting approach. Finally, experiments with several natural video sequences with complex textures are illustrated in Section 6.5. We show that the proposed method provides globally consistent results and can be efficiently used in several applications namely the object removal, error concealment and background estimation.

5.1 Graph-based inpainting approaches: state-of-the-art

Graph-based inpainting approaches recover each pixel p in the missing region Ω with the best candidate pixel value (so-called label) that yields a global coherent inpainted hole. The coherency is ensured by using an energy function that expresses the space-time consistency brought by each possible solution. The inpainting solution is then the best set of pixel values to copy for all missing pixels that provides the better visually quality result. The inpainting problem is therefore formulated as a global minimization problem.

5.1.1 Graph-based energy functions

5.1.1.1 Graph building

An efficient method to solve this energy minimization problem is the graph cuts optimization. Thereby, the inpainting is considered as a specific graph for the energy function such that the minimum cut on the graph also minimizes the energy. The inputs of the graph are both the set of missing pixels Ω and the set of labels L representing the possible known candidates for each missing pixel. Then, the goal became to find a mapping from Ω to L which minimizes the predefined energy cost. The result of such minimization algorithms is either the global minimum or a local minimum that is within a known factor of the global minimum of the energy cost [45].

5.1.1.2 Energy functions

The standard form of the energy ξ is often a weighted sum of two main terms respectively named data and smoothness terms as follows:

$$\xi = \sum_{p \in \Omega} E_d(p) + \alpha \sum_{(p,q) \in N(\Omega)} E_s(p,q)) \tag{5.1}$$

For each missing pixel p, the data term E_d computes the cost of each one of the candidate pixels to correctly recover p. This cost can be driven by several parameters: a coarse initialization, local correlation, temporal consistency and so on.

On the other side, the smoothness term E_s which the classical form is as follows:

$$E_s(p,q) = \left\| I_{l_p}(p) - I_{l_q}(p) \right\|_2 + \left\| I_{l_p}(q) - I_{lq}(q) \right\|_2 \tag{5.2}$$

constrains the global consistency of the inpainted result by minimizing the discontinuity between each pairs of neighboring pixels as expressed by Equation 5.2. The consistency is then ensured by preventing incoherent seams (Figure 5.1). In others words, the seam provided by candidate pixels chosen to inpaint two neighboring pixels in the hole should be similar to the seam of each candidate with its neighbors in the source region. In such way, the segment or sample that will be constructed inside the hole region should be similar to a segment in the known region. Therefore, the visual consistency of the result is achieved by encouraging

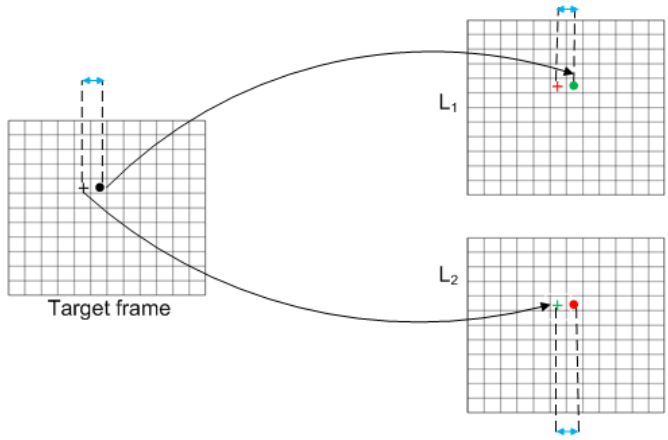


Figure 5.1: **Smoothness constraint.** (Illustration inspired from [58])

inpainting each two pairs of neighboring pixels with the same frame segment. The consistency can be spatial and/or temporal depending on the considered neighborhood system $N(\Omega)$. For instance $N(\Omega)$ can be defined as the 4, 8 or 26 nearest neighbors of each missing pixel $p \in \Omega$.

5.1.2 Graph-based background inpainting approaches

In this section we review the design of energy terms proposed in existing graph-based inpainting approaches dealing with either object removal or background estimation applications. The main difference between these methods is the definition of each term of the energy which has a great impact on the inpainting quality. However, as we are inpainting missing holes corresponding to the background pixels, almost all of these methods define the energy terms in a way ensuring the stationarity between frames. A well-known method dealing with the problem of background estimation has been proposed in [82]. The data term in this method is the combination of two energy terms expressing respectively the stationary and the consistency of the result. The stationary term is given by the variance of the collocated pixel values in the images set. While the consistency term helps to penalize pixel candidates belonging to the moving objects. Here, a pixel in image f_i is assumed to belong to a moving object if it has a small intensity gradient while the gradient of the difference between the target and source image f_i (so-called gradient motion) is large. This method has been extended in [83] for background estimation of a set of images capturing the same scene in different times (non-time sequence). For this aim, instead of the gradient motion, a background model has been considered to compute the consistency term. This background model defined as the mean of all image candidates is a guiding term such that the most probably candidate pixel value is the most similar to the collocated in the background model. Additionally, a third term is added (so-called hard term) to enforce the local consistency of the result by avoiding hard transition between neighboring pixels. The result should be similar to at least one image of the candidates. Chen et al. [46] improve the last energy using two main modifications. The background model is a guiding term computed as a weighted spatial interpolation and the stationary is computed based on the candidates redundancy in the images stack.

In a context of image inpainting a similar data term has been proposed by White et el. [52]. This approach inpaints the missing holes using an energy optimization of pixel candidates provided by similar images from the internet. The data term is composed of two terms. The first one expresses the similarity of the candidate image to the target. While the second term is based on the similarity of the pixel candidate to the guiding term which is the median of the image candidates. Finally the smoothness term of this approach is computed on the gradient of the image to be insensitive to the lighting change between the image candidates. A similar approach has been recently proposed for video inpainting by Granados et al. in [1]. The data term in this case is simply computed based on the similarity of the candidate to the guiding term which is the weighted temporal

interpolation of neighboring frames. The last method provides a good inpainting quality even for large holes and high resolution videos.

Almost all of these approaches are time consuming even for low resolution sequences. Furthermore, the energy terms are defined in a way to be adapted to either object removal or background segmentation. A better idea will be to provide a unique well-defined energy function able to provide high quality results in both applications. Besides, the minimization process which is driven by a guiding term sometimes named predicted term lacks accuracy to be very helpful for inpainting. For instance, the predicted term in [1] which is a simple weighted interpolation of collocated pixels in the images set assumes that there is at least one known pixel in the stack of pictures for each missing pixel of the target image. This assumption turns out to be true when the temporal window is very large. To avoid these limitations we propose in this chapter a novel energy function for inpainting background holes in video sequences able to provide high quality results with reduced sliding temporal window in both object removal and background estimation applications. While being built upon existing background estimation techniques [1, 46, 52, 82], the proposed approach extends them by bringing the following main contributions:

- A temporal inpainting with a new well-defined cost function ensuring both spatial and temporal coherence.
- An efficient spatial inpainting initialization to guide the choice of the most likely pixel value in the neighboring frames. Thereby, a small sliding window (at most 20 images) is enough to provide more natural results than the most recent techniques while being drastically less complex.

5.2 Proposed graph-based inpainting approach

5.2.1 Overview

The proposed inpainting method illustrated in Figure 5.2, recovers missing areas Ω_t in a target frame I_t using data from the stack of the neighboring frames. Each missing pixel has to be copied from the best collocated pixel value that globally minimizes a predefined cost function ξ. Let $I^*(p)$ denotes the labeling of the best pixel value collocated to each missing pixel p that ensure the best globally pleasing result. This label indicates the neighboring frame I_l from which the pixel p has to be copied. I^* is obtained by minimizing the cost function ξ defined on all pixels in Ω_t as follows:

$$\xi(I_l) = \sum_{p \in \Omega_t} E_d(I_l(p)) + \alpha \sum_{(p,q) \in N(\Omega_t)} E_s(p,q)) \tag{5.3}$$

The data term of the energy cost i.e. $E_d(I^*(p))$, expresses here the stationary

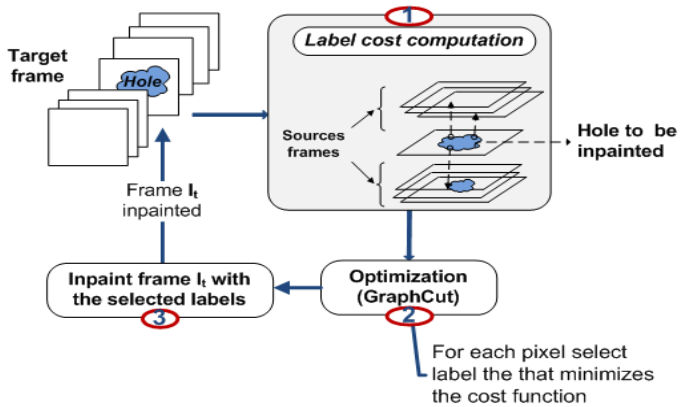


Figure 5.2: **Overview of the proposed approach.**

of the background information. For each possible label l (i.e. neighboring frame I_l), the data term $E_d(I_l(p))$ is the sum of three terms:

$$E_d(I_l(p)) = E_0(I_{tl}) + E_1(I_l(p)) + E_2(I_l(p)) \tag{5.4}$$

where $E_0(I_{tl}) = log(1 + MSE(I_t, I_l))$ expresses the similarity in terms of Mean of Squared Differences (MSE) between each of the neighboring frames I_l ($l \in [t - M \cdots t + M]$) and the current target frame I_t. This term helps to favor the pixel that belongs to the frame which is the best similar to the target. In a context of moving camera videos, this term will also gives idea about the registering quality with penalizing the source frame not correctly aligned to the target. The logarithm is used here to limit the dynamic range of this term.

The second term of the data energy $E_1(I_l(p))$, so-called stationary term is given by:

$$E_1(I_l(p)) = \frac{1}{2M\,|\Psi_l(p)|} \sum_{k=-M}^{M} \|\Psi_l(p) - \Psi_k(p)\|^2$$

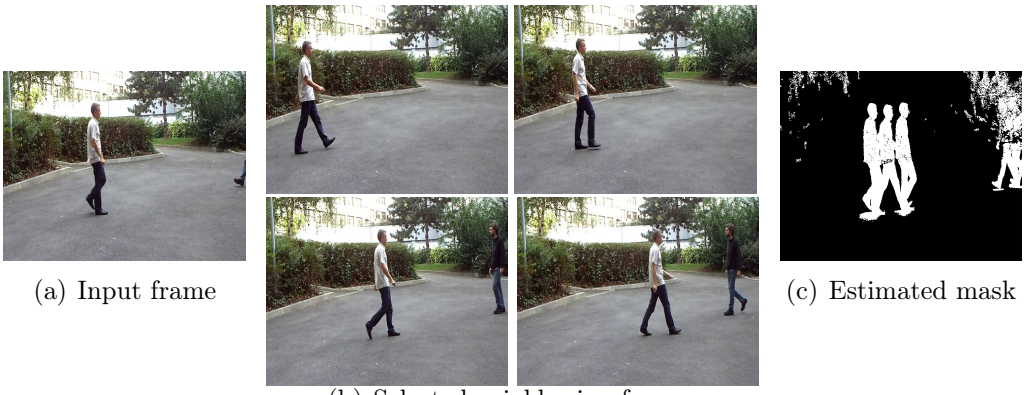
where $\Psi_l(p)$ is the patch centered on the pixel collocated to p in the frame I_l and $|\Psi_l(p)|$ is its number of pixels. In this way, we consider that most probable background pixel values among the possible labels are those which are the most redundant.

The term $E_2(I_l(p))$ is a predicted term and represents the similarity of the candidate pixel value to the guiding initialization. The initialization value is obtained by performing a spatial inpainting of the hole. When the pixel is missing in all the registered frames, the spatial initialization value is considered for inpainting. This term contributes to improve the spatial and temporal coherence of the inpainted result.

The last term of the energy function in the equation 5.3, is a smoothness term defined on each pair of neighboring pixels (p and q) in the hole. The quantity N refers to the 4-neighbors of the pixel $p \in \Omega$. Finally, the parameter α expresses the weight of the smoothness in the energy cost compared to the data term (E_d).

5.2.2 Graph-based energy for background estimation

The proposed inpainting approach can also be used for background estimation. This non-parametric object removal application has to first estimate a binary mask of all moving objects in the scene before applying the inpainting algorithm to correctly reconstruct their corresponding holes. The building of the binary mask is very important since it consists in deciding if a pixel belongs to the background or moving objects. Several methods can be used to correctly segment moving objects in the scene. We propose here to consider a simple and greedy mask building to provide a fast background estimation approach. This step illustrated in Figure 5.3, builds the hole mask (Ω_t) corresponding to the moving objects in each frame I_t by using a classification of each pixel as static or non-static values. Only a small number of distant neighboring frames are selected at each side of



(a) Input frame

(b) Selected neighboring frames

(c) Estimated mask

Figure 5.3: **Background estimation: hole estimation step.**

I_t to be able to correctly detect the moving objects. Then, a pixel is considered as moving if its intensity value changes in the selected frames set. Otherwise, it is considered a static background pixel. The same frames will provide the pixel values candidates for inpainting the estimated hole. This helps to provide better quality results since we have higher probability to find the unoccluded pixel value in a distant frame than in a neighboring one. Furthermore, the time computation of the proposed graph-based inpainting approach is dependent on the number of the pixels to inpaint and the candidates (labels) for each one. Therefore, using all the video or GOP frames makes the background estimation approach too slow.

A small number of distant neighboring frames helps to drastically reduce the computational time while ensuring better detection of moving objects.

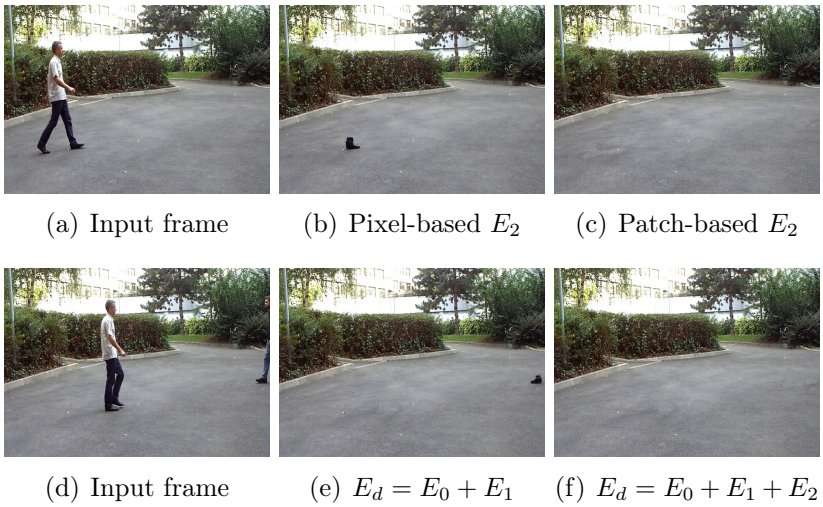
On the other hand, in such application the estimated hole corresponding to all non-static regions in the scene is much larger than the one used for object removal where it is often limited to one or two objects to be removed from the scene. Correctly reconstructing such large hole (please see Figure 5.3(c)) makes the application of background estimation a challenging test of the proposed inpainting algorithm.

5.2.3 Impact of energy terms

In this section we focus on the proposed energy function by studying the impact of each term on the inpainted result.

5.2.3.1 Data term

Figure 5.4 illustrates the impact of the data term on the result in a context of background estimation of a video sequence containing 240 frames of 720×404. One can notice in Figure 5.4(e) that the data term based only on the similarity



(a) Input frame (b) Pixel-based E_2 (c) Patch-based E_2

(d) Input frame (e) $E_d = E_0 + E_1$ (f) $E_d = E_0 + E_1 + E_2$

Figure 5.4: **Impact of the stationary term.** (a) Original image. (b) Estimated background using pixel-based stationary term. (c) Estimated background using patch-based stationary term(patch of 9×9). (d) Original image. (e) Estimated background without stationary term. (f) Estimated background using patch-based stationary term(patch of 9×9).

of the candidate with the initialization ($E_d = E_0 + E_1$) as proposed in [1] is not sufficient to correctly find the best background pixels in the neighboring frames.

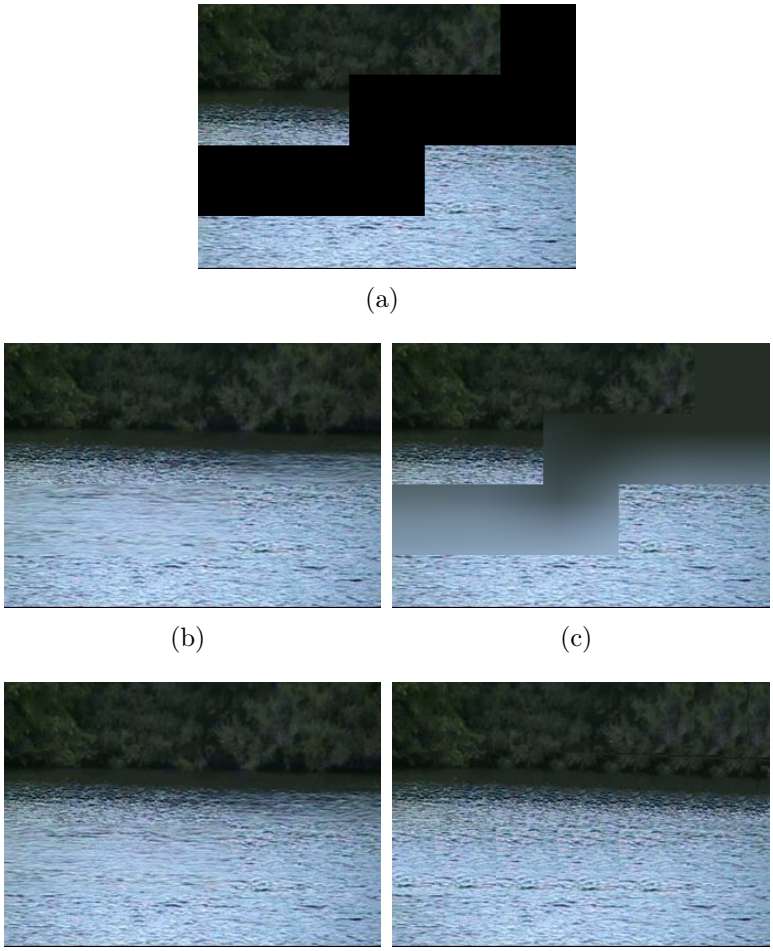
Moreover, computing this stationary using the patch centered on p (Figure 5.5(c)) instead of a pixel-based approach (Figure 5.5(b)) as proposed in [46] makes the approach more robust to the noise and illumination variations between frames. The proposed data term helps to better distinguish between pixels belonging to the background and those of moving objects which helps to provides better quality of background reconstruction.

5.2.3.2 Guiding term

The guiding term also called predicted term ($E_2(p)$ in equation 5.4) is often used in MRF-based inpainting methods to guide decision for inpainting of missing pixels. Higher priority is given to pixel values of the neighboring frames that are the most similar to the initialization prediction. As we mention before, in state-of-the-art methods, this term may be a simple spatial interpolation as proposed in [46] where each missing pixel value is updated as follows:

$$I(p) = \frac{1}{C_w} \sum_{p' \in W \setminus \Omega} (1 - \frac{\|p - p'\|^2}{|W|}) I(p') \tag{5.5}$$

here W is a window centered on p and C_w is a normalizing factor. The guiding term may also be a temporal interpolation of collocated pixel values in neighboring frames. This interpolation can be a simple average of the values, the median as proposed in [52] or a weighted combination as proposed in [1]. The results of these approaches illustrated in Figure 5.5 show that such coarse interpolations do not provide high quality of estimation. However, the goal here is to have an inpainting framework easy to be used for different applications. For instance, in error concealment application only a few number of neighboring frames are available (GOP frames). Therefore, a robust spatial inpainting is needed to provide good results when the missing hole is the same all over the set of images. For that, we consider a spatial inpainting inspired from a recent method based on the patch offsets in the image [51]. The Approximate Nearest Neighbor Field (ANNF) of each source patch in the target image (so-called patch offsets) is first computed using the Patch Match method [41]. From the ANNF, the 30 most principal modes of the offsets distribution are deduced. A stitching of the shifted images (using the selected dominant offsets) is performed. Inpainting is finally obtained by minimizing a predefined energy cost [51]. The results of this method is illustrated in Figure 5.5(e). This spatial inpainting is used as a guiding term if the missing pixel is discovered at least in one of the neighboring images otherwise it will be used as the inpainting result.

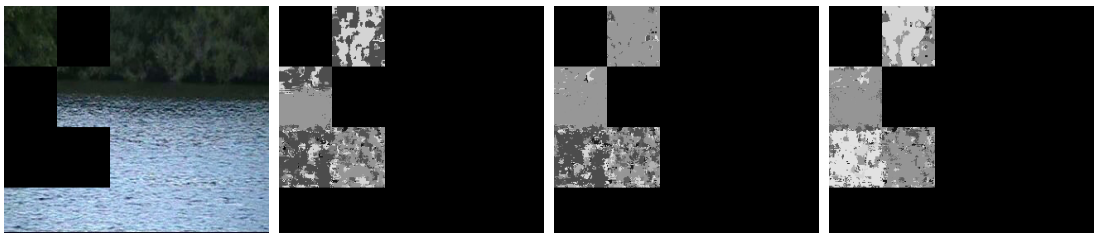


(a) (b) (c) (d) (e)

Figure 5.5: **Guiding term using several approaches.** Here a temporal sliding window of 11 is used (5 frames in each side). (a) Original image + mask. (b) Mean of temporal candidates. (c) Spatial interpolation proposed in [46]. (d) Temporal interpolation proposed in [1]. (e) Spatial inpainting using method in [51].

5.2.3.3 Smoothness term

The smoothness term which is used to penalize seams between neighboring pixels in the inpainted hole aims here at encouraging their inpainting using the same frame candidate. The weighting factor α measures the importance of this term compared to the data term in the inpainting energy. Optimized labels used for inpainting missing pixels for various values of α are illustrated in Figure 5.6. One can mention that using the same weighting for data and smoothness terms (i.e. $\alpha = 1$) may produces discontinuities in the inpainted results. The static part



(a) Original image + mask (b) $\alpha = 1$ (c) $\alpha = 10$ (d) $\alpha = 100$

Figure 5.6: **Impact of smoothness term: inpainting labels for several value of α.**

of the video (trees) is not inpainted using the same labels i.e. the neighboring pixels in the hole are not inpainted with pixels coming from the same frame segment. A higher weighting of the smoothness term equal i.e $\alpha = 10$ provides a better consistency with considering almost a single frame candidate for inpainting neighboring pixels in the static region (trees). For missing region corresponding to dynamic background (water) more than one labels is used since the pixels are not static and the most redundant pixels has to be chosen. In the rest of the thesis we consider a fixed value $\alpha = 10$.

5.2.3.4 Energy convergence

The inpainting is performed by minimizing globally the predefined energy function. The minimization is achieved by the expansion-move algorithm [49,84]. This algorithm initialize a random solution for all missing pixels and then iterates to refine the results with the aim at each time to reduce the energy cost obtained in the previous iteration. The algorithm stops when a maximum iterations number

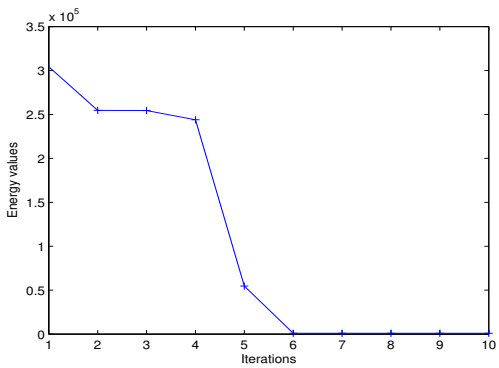


Figure 5.7: **Energy convergence.**

is achieved or no improvement between two successive iterations. Thus, the value of maximum iterations is here very important since a higher value may lead to a high complexity and a few one lead to lower inpainting quality. The adequate iteration number should then satisfy the tradeoff of quality and computation time. Our experiments of energy convergence illustrated in Figure 5.7 show that 5 iterations are usually enough to ensure good quality of inpainting. In the rest of the thesis, 5 iterations are made for all optimization steps.

5.3 Experiments results

We evaluate the performances of the proposed approach in three main applications: object removal, error concealment and background estimation. The main difference between these applications is the form of the hole to be inpainted. For object removal and error concealment, the spatial location of missing areas are known. However, there is a difference in the shape of the missing areas. In the context of loss concealment, the missing areas correspond to the lost blocks detected after decoding. Whereas in the context of object removal the hole to be inpainted corresponds to the object to remove from the scene which can be of various shape. Concerning the background estimation, as previously described, there is no prior knowledge about the areas that should be inpainted. The inpainting method has to classify all pixels as belonging to the background or not.

All the following results are obtained with the same configuration. The sliding window is composed of $21(M = 10)$ frames.

5.3.1 Object removal

The first experiments assess qualitatively the performance of the proposed method on a set of challenging video sequences of the change detection dataset [1] in a context of object removal. This dataset has been initially designed to benchmark change detection algorithms [85]. The dataset is composed of 31 real-world videos totaling over $80,000$ frames and spanning 6 categories selected to include diverse motion and change detection challenges. It is representative of indoor and outdoor visual data captured today in surveillance and smart environment scenarios. For all these sequences, a ground truth is provided in the form of binary maps indicating where the changes occur. We extract from this dataset several sequences from the following categories:

- baseline: this category contains simple videos without camera motion. Three video sequences have been used to test the proposed method namely High-

1. http://changedetection.net/

way (325 frames of 320×240), Office (150 frames of 360×240) and pets2006 (500 frames of 720×576).

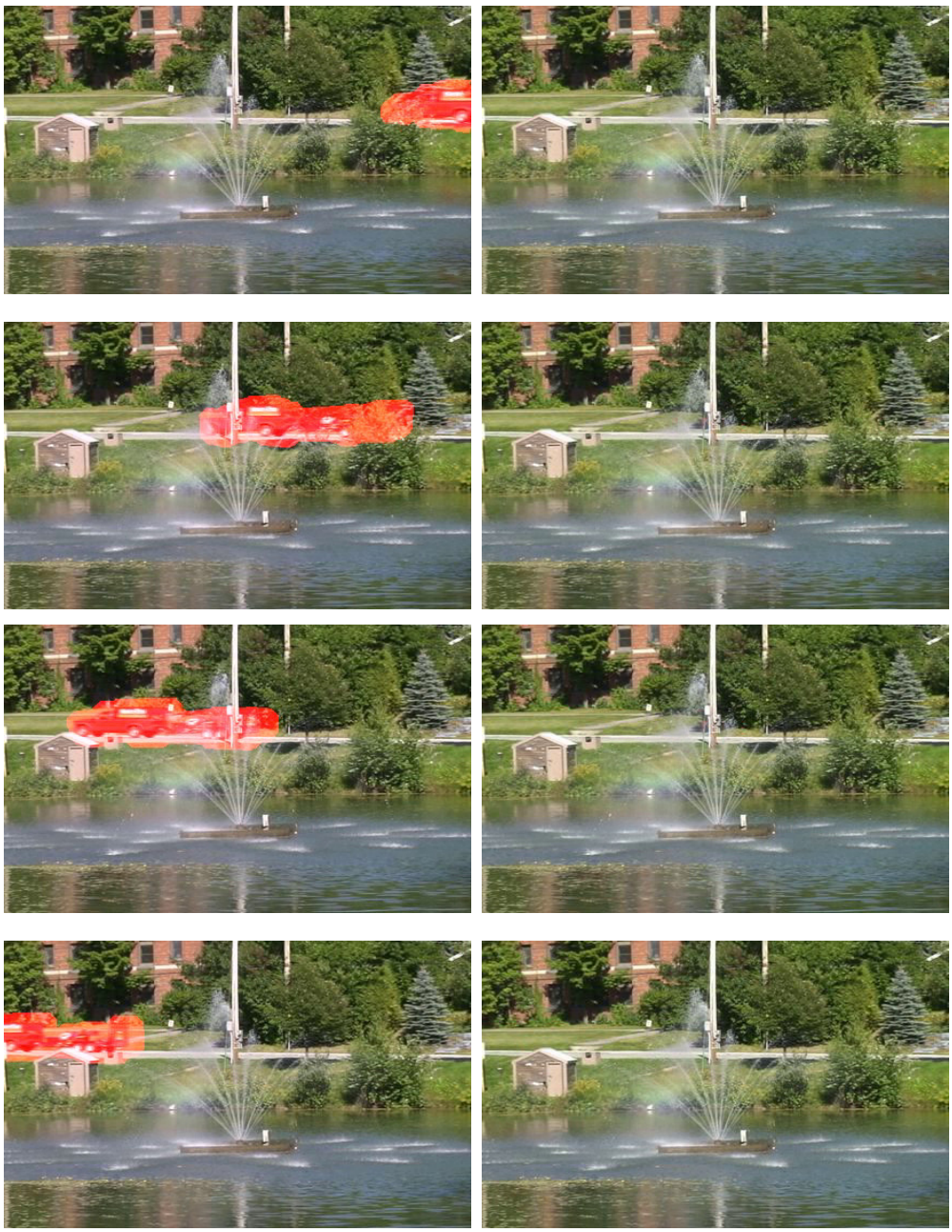
- dynamic background: video sequences of this category exhibit dynamic background motion with complex textures (for instance water). Four sequences have been tested, boats (150 frames of 320×240), fountain01 (200 frames of 432×288), fountain02 (140 frames of 432×288), canoe (300 frames of 320×240) and fall (475 frames of 720×480).
- intermittent object motion: it contains videos with background objects moving away, abandoned objects and objects stopping for a short time and then moving away. Two video sequences have been selected namely sofa (400 frames of 320×240) and winterDriveway (420 frames of 320×240).
- shadow: this category contains video sequences with a lot of shadow and illumination change. Two video sequences have been selected, busStation (400 frames of 360×240) and peopleInShade (250 frames of 380×244).

A total of 15 video sequences representing 4080 frames are then used to evaluate the performance of the proposed model. The object to remove is given by the ground truth (in the form of binary map). Note that to the binary maps have been manually modified in order to fit the constraints required by the inpainting. The use of this set of video sequences is interesting since these input video sequences are very challenging showing illumination problem, indoor and outdoor scenes, different sizes of objects to be removed, stochastic textures, different level of quality as well as resolution, etc.

The proposed inpainting approach provides high quality of recovering results with the different video textures. In fact, inpainted results of a dynamic background sequence illustrated in Figure 5.8 show high quality of inpainting. In this sequence, the removed object from the scene is far from the camera plane. Similarly, in the case of large holes corresponding to removed objects in front of the camera plane (as represented in Figure 5.9), the missing area is correctly recovered showing visually pleasing results. Besides high dynamic background illustrated in Figure 5.10 shows also temporal coherent inpainted results. Furthermore, video sequences with illumination change illustrated in Figures 5.11 and 5.12 are correctly recovered showing temporal consistent results.

5.3.2 Error concealment

We further evaluate the performances of the proposed approach in a context of error concealment. For this purpose, we consider the video sequence Fountain01 illustrated in Figure 5.13(b) where respectively 5%, 10% and 20% of 16×16 blocks are lost. The PSNR values of the inpainted sequences are respectively 26.08 dB, 25.97 dB and 25.73 dB. However, the visual quality of the inpainted

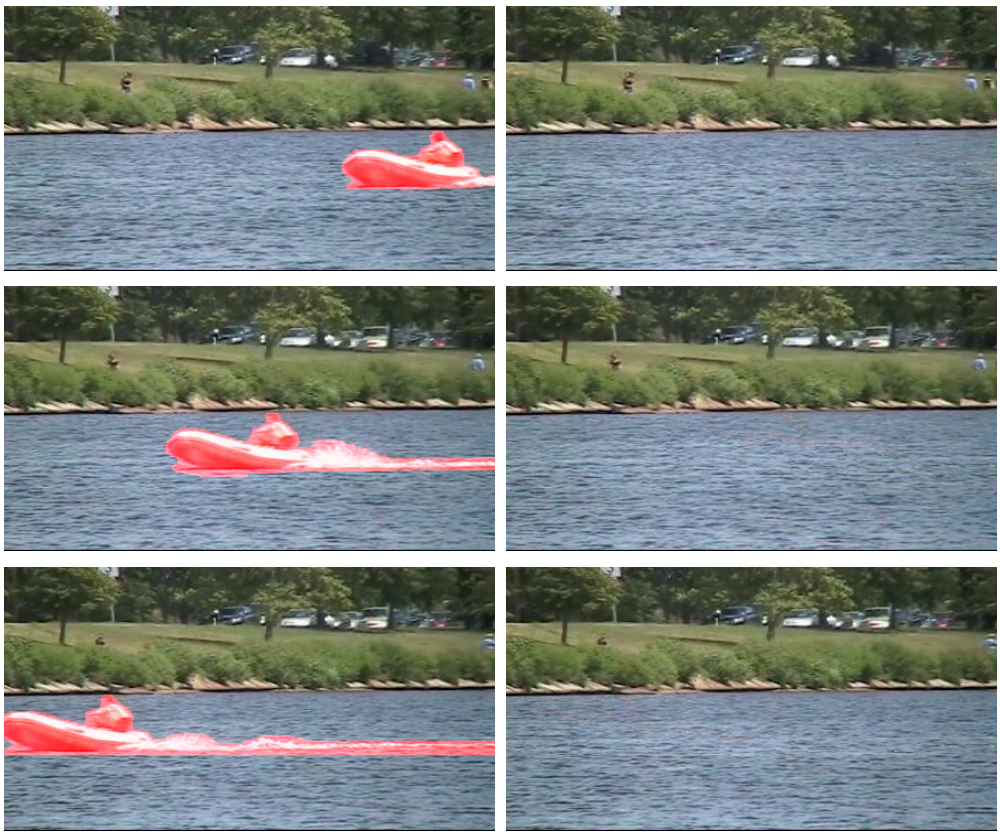


(a) (b)

Figure 5.8: **Inpainted results of change detection video sequence with dynamic background: fountain02 (432×288).** Sequence with small hole. (a) Input image with the mask of object to be removed. (b) inpainted result using the proposed approach.

(a) (b)

Figure 5.9: **Inpainted results of change detection video sequence with dynamic background: fall**(720×480). Sequence with large hole. (a) Input image with the mask of object to be removed. (b) Inpainted result using the proposed approach.



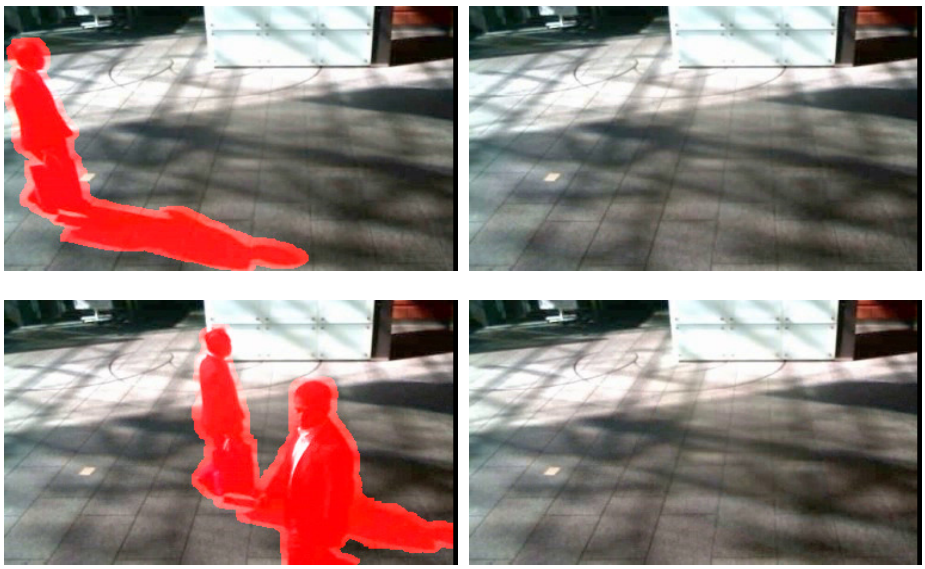
(a) (b)

Figure 5.10: **Inpainted results of change detection video sequence with dynamic background: boats**(320×240)**.** (a) Input image with the mask of object to be removed. (b) inpainted result using the proposed approach.

images illustrated in Figure 5.13 shows correctly recovered results. In this test, the sequence presents a dynamic background (water). Then, providing a high quality of recovering in terms of PSNR is not ensured either by graph-based energy or a simple patch-based copy of missing blocks in neighboring frames. However, the visual quality is here ensured thanks to the energy terms constraints.

5.3.3 Background estimation

The proposed approach is also validated in a context of background estimation of static camera sequences. For each frame I_t, the mask of the moving objects (non static regions Ω_t) is first estimated using four distant neighboring frames (2 past and 2 futures frames). Then, the proposed video inpainting technique is used to inpaint Ω_t. As shown in Figure 5.14(b), the non-static regions are the

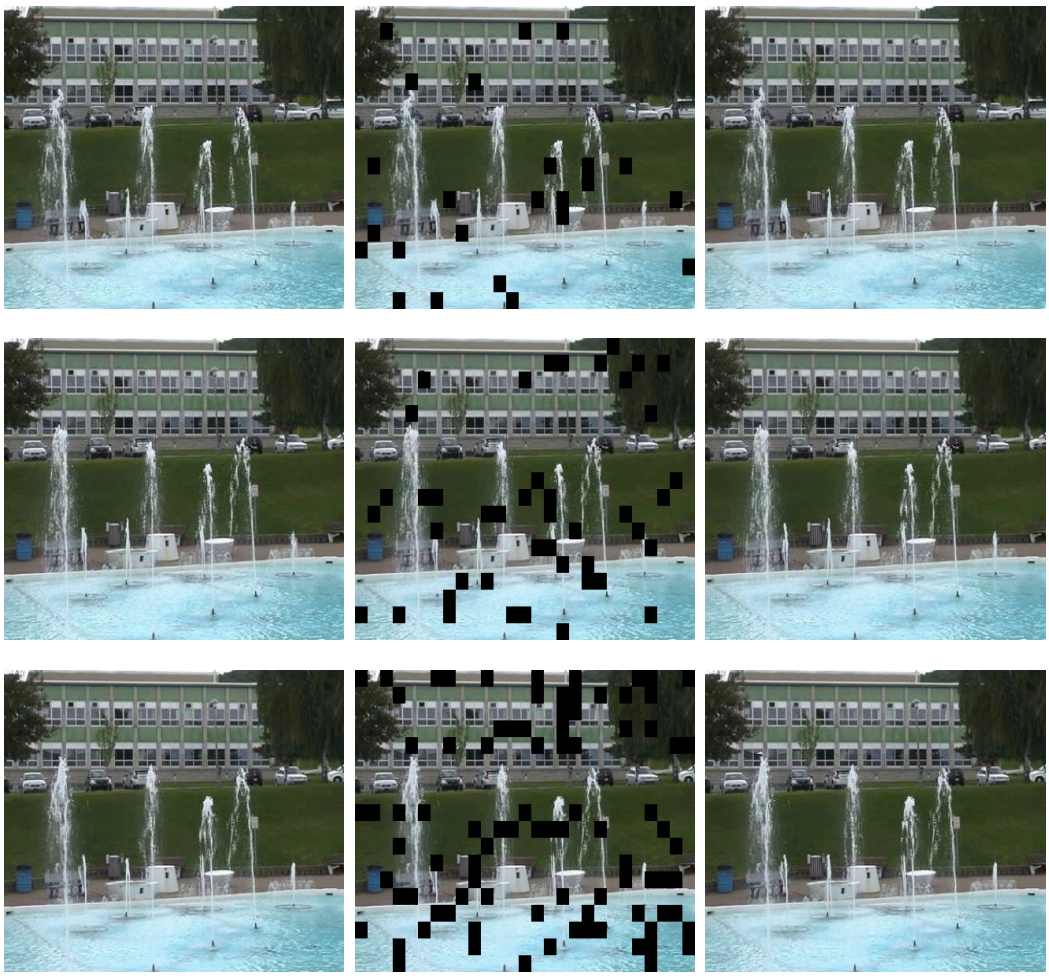


(a) (b)

Figure 5.11: **Inpainted results of change detection video sequence with shadow: peopleInShade** (380×244)**.** (a) Input image with the mask of object to be removed. (b) Inpainted result using the proposed approach.

(a) (b)

Figure 5.12: **Inpainted results of change detection video sequence with intermittent object motion: winter driver.** (a) Input image with the mask of object to be removed. (b) Inpainted result using the proposed approach.

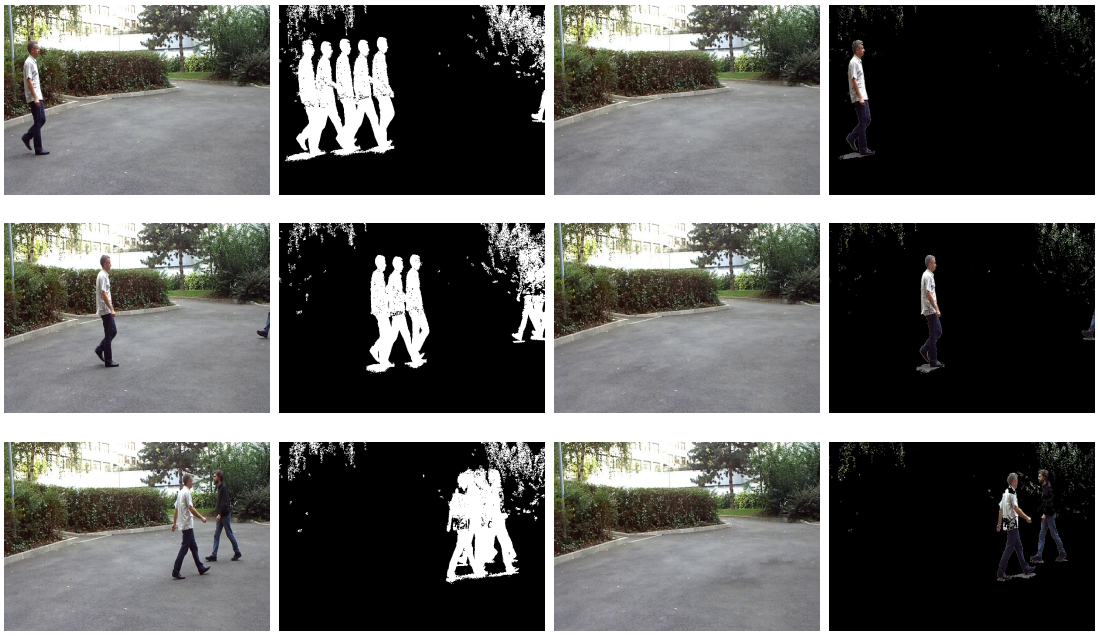


(a) (b) (c)

Figure 5.13: **Error concealment results of the sequence fountain01.** Three frames of the scene with three different loss rate of 16×16 blocks. From top to bottom: respectively 5%, 10% and 20% of blocks are lost. (a) Input frames. (b) Frames with the missing blocks. (c) Inpainting results using the proposed approach.

union of holes corresponding to moving objects in each considered frame which makes the mask larger than a hole in the context of object removal. Background estimation results illustrated in Figure 5.15 show that the proposed approach provides space-time consistent background results all over the sequence. Then, moving objects can easily obtained with a simple subtraction of the estimated background and the input image as illustrated in Figure 5.14(d).

Additionally, inpainting results of dynamic background scenes illustrated in Figure 5.16 show that the inpainting quality of both parametric (using a mask of object to be removed) and non-parametric (i.e. no input mask of the hole region) video inpainting are quite similar and with high quality of background estimation.



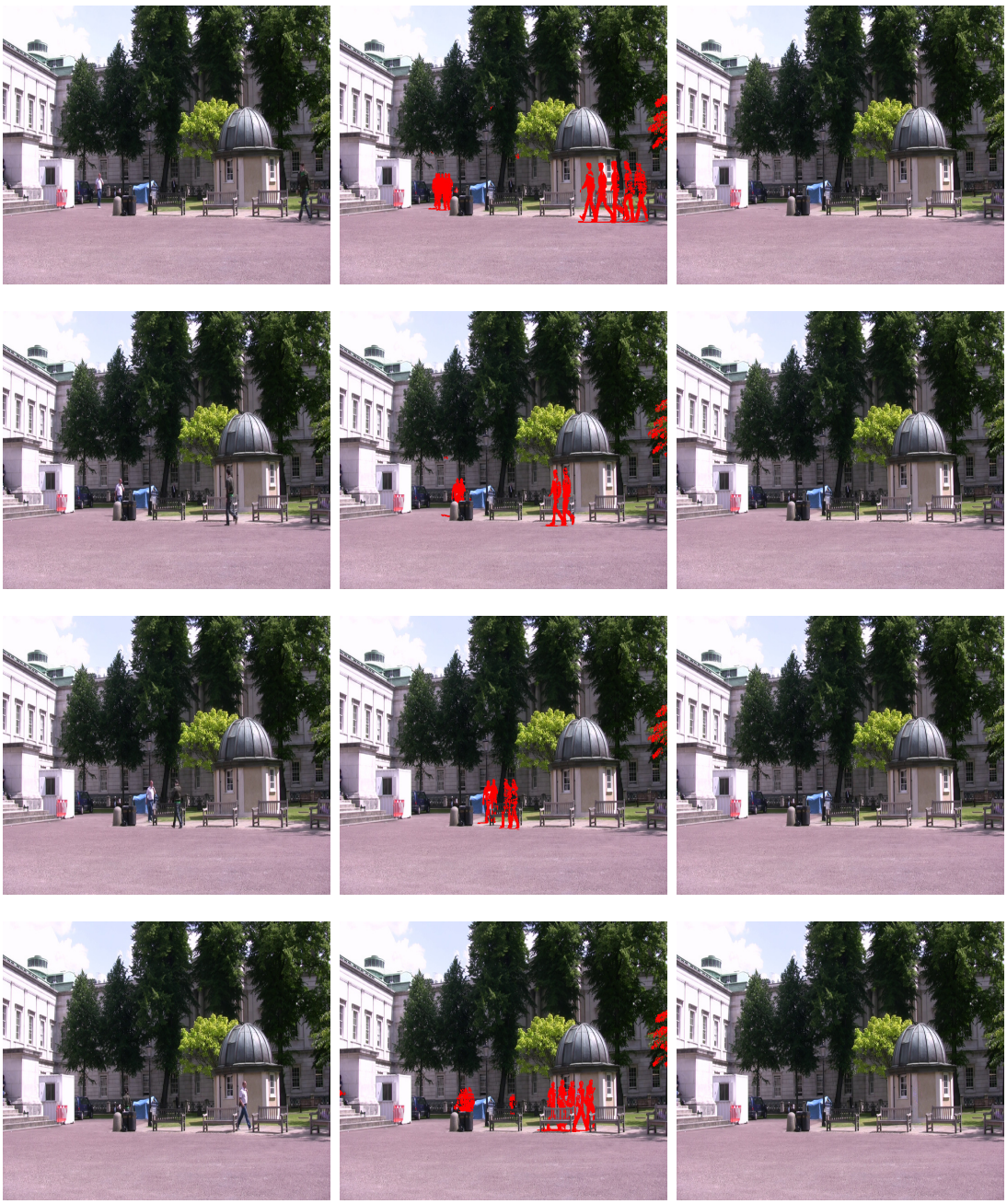
(a) (b) (c) (d)

Figure 5.14: **Background estimation: sequence terrace.** (a) Input frame. (b) Estimated non static regions (hole to inpaint). (c) Inpainted results. (d) Subtracted moving objects results.

5.4 Conclusion

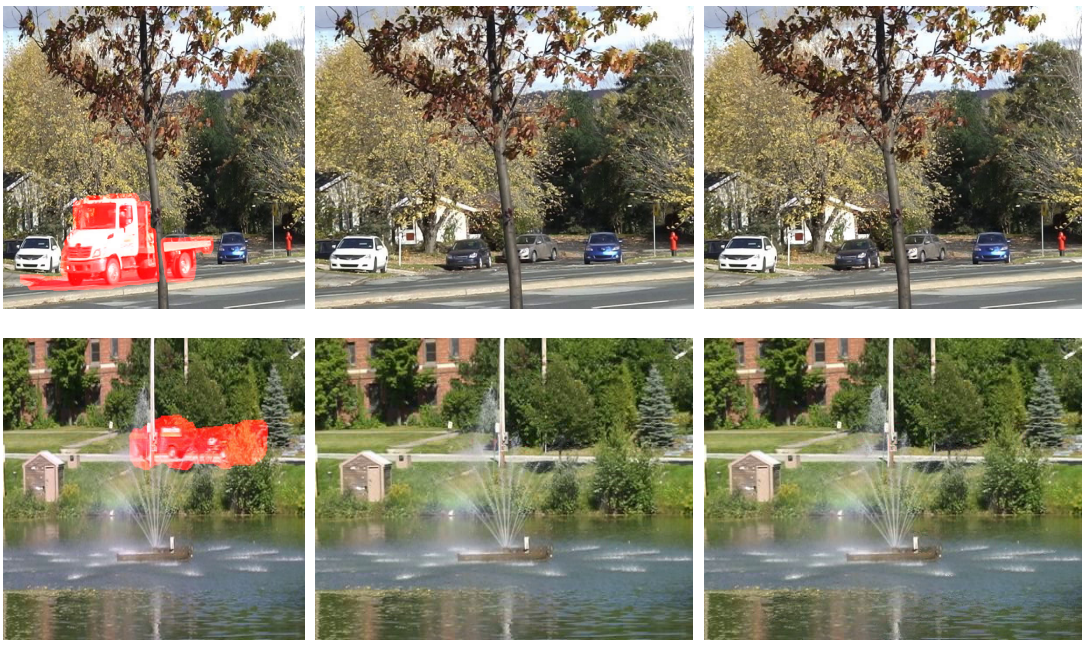
In this chapter, we proposed an extension of background inpainting methods by describing a new energy cost terms ensuring both space and time coherency of the result. Experimental results showed that this method can be efficiently used for several applications with providing high quality results for challenging sequences captured with static camera. In Chapter 6 we propose an extension of this approach for inpainting missing holes in moving camera videos.

Furthermore, we think that the proposed energy-based inpainting approach can be also extended for the inpainting of holes corresponding to missing pixels of moving objects. In object removal application, this case occurs when there is



(a) (b) (c)

Figure 5.15: **Background estimation.** (first row) Frame 0; (second row) Frame 60; (third row) Frame 120; (last row) Frame 240. (a) Input frame. (b) Estimated non static regions (hole to inpaint). (c) Inpainted results.



(a) (b) (c)

Figure 5.16: **Background inpainting comparison parametric vs. non-parametric approaches.** (a) Input frame. (b) Estimated background using the binary mask of the hole to inpaint. (c) Estimated background without using the binary mask of the hole to inpaint (unsupervised inpainting).

occlusion between moving objects and one of them has to be removed from the scene. Whereas, in error concealment context, some of the missing blocks may correspond to missing pixels of moving objects. In these cases the space-time coherence has to be enforced in occlusion regions to correctly recover missing regions of moving objects.

Chapter 6

Image registration for inpainting free moving camera videos

When the video sequence is captured with a moving (like handheld) camera, the neighboring frames in the scene does not present the same viewpoint change. Thus, video inpainting algorithms need to first register neighboring frames in order to remove the camera motion before inpainting the missing regions. This step is very important in the video inpainting algorithm since registration errors may lead to severe artifacts in the inpainted results. In this chapter we only focus on this preprocessing step commonly named image registration or alignment.

A large number of registration techniques can be found in the literature. A survey of several image registration methods can be found in [86]. The main difficulty of these methods is to be robust and flexible enough to efficiently deal with several types of scene characteristics: blurring, illumination change, scene movement and ghosting caused by parallax. Furthermore, the camera motion can be a single motion such as rotation around one of the three axes, translation, zoom in and out which can also be considered as a translation along the z-axis, or a combination of these motions.

Most registration approaches are based on camera motion estimation techniques. The camera motion can be estimated using a three-dimensional motion model as proposed by Yao et al. [87]. In this approach both the structure of the scene and the camera motion are simultaneously estimated using a recursive method based on extended Kalman filters. The three-dimensional motion models are usually time consuming. A faster approach computes the camera motion using a two-dimensional affine model describing the motion between each pair of consecutive frames. This approach is often based either on a dense minimization of the pixel-to-pixel dissimilarities or the matching of a sparse set of features ex-

tracted from each image [9, 10]. Optical flow and feature correspondences-based methods are the widely used two-dimensional registration approaches. Respectively, optical flow methods compute the camera motion using the motion vector of each pixel or block in the images while feature-based methods align images using the matching of sparse set of key points.

The two-dimensional image registration approach is less complex to implement and less time consuming compared to the three-dimensional method which makes it the most used approach in video processing applications dealing with moving camera scenes. In this work, we choose to focus on two-dimensional approaches to solve our registration problem. In this chapter, we first present a survey of the main used registration approaches. The camera model is described in Section 6.1 to better understand geometry relationship between image pairs in a sequence captured using a moving camera. This relationship is detailed in Section 6.2 before describing in Section 6.3 the mostly used motion estimation methods for computing the geometric transformation between images. Then, Section 6.4 focus on warping approaches used in video editing applications with presenting the proposed image alignment approach based on region segmentation. Finally, the performances of the proposed approach is compared to the most recent registration methods in Section 6.5.

6.1 Camera model

The first step to solve the registration problem consists in extracting the correspondence between pixels in the image pair. Then, the motion model that expresses the geometric transformation mapping one image to the other is computed. These pixel correspondences are then function of the camera motion assuming that the pixels are the projection of the same points in the 3D space. As illustrated in Figure 6.1, each point $m = (x, y)^T$ in the 2D image plane (inhomogeneous coordinates) is the projection of a point $M(X, Y, Z)$ in the 3D space (homogeneous coordinates). Let χ the perspective projection between m and M. χ can be defined as:

$$\begin{array}{llll} \chi : & \mathbb{R}^3 & \rightarrow & \mathbb{R}^2 \\ & M = (X, Y, Z) & \mapsto & m = (x, y) \end{array} \tag{6.1}$$

Furthermore the homogeneous and inhomogeneous coordinates are related with the following relation:

$$\begin{cases} x = & f_c \frac{X}{Z} \\ \\ y = & f_c \frac{Y}{Z} \end{cases} \tag{6.2}$$

where f_c is the focal distance between the projection center of the camera C and the image plane I. Therefore, m can be represented in homogeneous coordinates

as $m = (x, y, 1)^T$ and

$$\begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = \begin{pmatrix} f_c & 0 & 0 & 0 \\ 0 & f_c & 0 & 0 \\ 0 & 0 & 1 & 0 \end{pmatrix} \begin{bmatrix} X/Z \\ Y/Z \\ 1 \\ 1/Z \end{bmatrix} \Leftrightarrow m = PM \tag{6.3}$$

The focal distance f_c is a scale factor of the image. This parameter is usually used

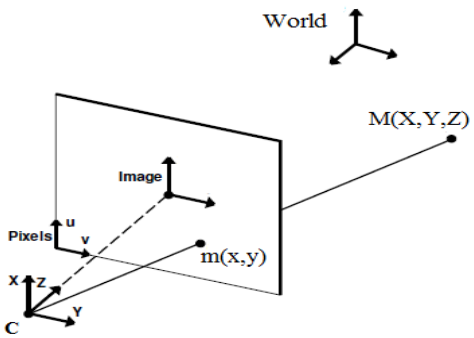


Figure 6.1: **Projection model.**

as the unit of the camera system coordinates ($f_c = 1$). The perspective projection χ between m and M is given by: $m = PM$ where P is the 3×4 camera matrix. However, the system coordinates of the camera is not necessary the same in 3D. Then, the basis transformation between the two coordinate systems needs to be considered. We note A the matrix expressing this basis transformation. This matrix also named the extrinsic parameters of the camera can be decomposed into a translation t and rotation matrix R. Moreover, the transformation between an image point and a ray in Euclidean 3-space is performed using a calibration matrix K describing the intrinsic parameter of the camera. Finally, the projection matrix P is defined as: $P = KP_0A$ where K and A represent the intrinsic and extrinsic parameters of the camera respectively. The matrix P can be then defined as follows:

$$P = [K|0] \begin{bmatrix} R & t \\ 0^T & 1 \end{bmatrix} \tag{6.4}$$

sometimes noted $P = [K|0]\,[R|t]$. One can mention that the optical center C of the camera is the unique point where: $PC = 0$.

6.2 Epipolar geometry notions

The most used calibration camera algorithms compute the matrix P from a set of point correspondences between images and decompose P into K, R and

t via the QR decomposition [88–90]. Another famous tool that helps to find correspondences between pixels of a image pair without computing the camera parameters consists in using the epipolar geometry concepts. In the following we present the basic notions of epipolar geometry that helps to achieve this goal.

6.2.1 Parallax effect

As previously mention, making correspondence between the pixels of two images or views is possible only if the pixels are the projection of the same points in the 3D space. However, this case is valid when the optical centers of the two cameras coincide, or when the captured scene is planar. Otherwise, e.g in the case of failure to rotate the camera around its optical center, the effect of parallax illustrated in Figure 6.2 occurs. In this case, a pixel in one image may correspond to the projection of more than one point in the 3D space which makes ambiguity in matching its corresponding pixel in the second image. Therefore, epipolar

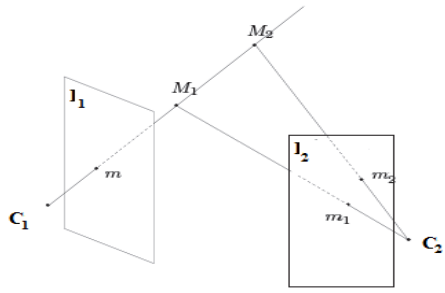


Figure 6.2: **Parallax effect.**

geometry can be used to find correspondences between pixels in image pairs only in parallax free scenes.

6.2.2 Epipolar constraint: fundamental matrix

The algebraic representation of the epipolar geometry between two images is performed using a 3×3 homogeneous matrix with 7 degrees of freedom so-called fundamental matrix [10]. This matrix relies the image of the ray through each point m_1 of the first image to a line I' named epipolar line on the second image (please see Figure 6.3). This epipolar line is defined as: $I' = e_2 m_1 m_2 = [e_2]_{m_1 m_2}$, where m_2 is the projection of m_1 on the second image. The epipolar line is usually used to restrict the search of corresponding pixel of m_1 to the line $I' = F m_1$ instead of the entire image I_2 (where F is the fundamental matrix between the two images). Furthermore, all corresponding pixels m_1 and m_2 should satisfy: $m_2^T F m_1 = 0$.

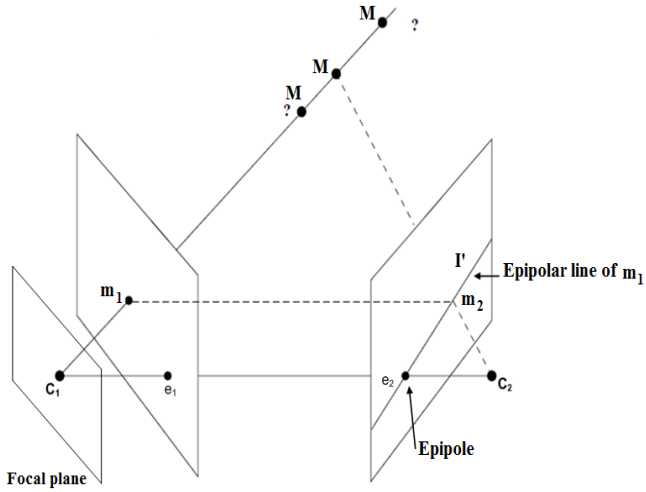


Figure 6.3: **Epipolar constraint.** Fundamental matrix relationship between two images.

6.2.3 Homography matrix

In addition to the epipolar geometry, two images or views can be related with the homography which projects points from one image to the other as if they were images of points on the plane [10]. For instance, as illustrated in Figure 6.4, if m_2 is the projection on the image I_2 of the pixel m_1 then, there is an homography H that maps m_1 to m_2 such as:

$$m_2 = Hm_1 \tag{6.5}$$

Homography estimation algorithm

Homography transformation is a 3×3 non-singular homogeneous matrix with 8 degrees of freedom. A simple method to compute this matrix is to consider the camera parameters. For instance, given a homography H that maps $m_1 \in I_1$ to $m_2 \in I_2$ (Figure 6.4) and assuming that m_1 and m_2 are both the projection of the same 3D point M, then

$$m_1 = \begin{bmatrix} K|0 \end{bmatrix} \begin{bmatrix} M \\ 1 \end{bmatrix} = KM$$

if m_2 is obtained after a rotation R of the camera, then,

$$m_2 = K \begin{bmatrix} R|0 \end{bmatrix} \begin{bmatrix} M \\ 1 \end{bmatrix} = KRM$$

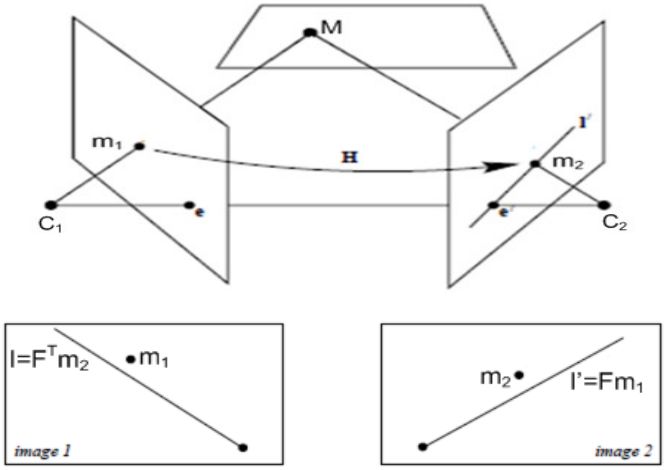


Figure 6.4: **Homography projection and epipolar constraints.**

which gives $m_2 = KRK^{-1}m_1$. Therefore, using equation 6.5, the homography H can be computed as: $H = KRK^{-1}$. In practice, the camera parameters are not available. The most commonly used approach to compute the homography matrix is however based on the pixel correspondences between images. For instance the Direct Linear Transform [91] computes H using the equation system provided by $m_2 = Hm_1$ i.e.

$$\begin{bmatrix} x_2 \\ y_2 \\ z_2 \end{bmatrix} = \begin{pmatrix} h_{11} & h_{12} & h_{13} \\ h_{21} & h_{22} & h_{23} \\ h_{31} & h_{32} & h_{33} \end{pmatrix} \begin{bmatrix} x_1 \\ y_1 \\ z_1 \end{bmatrix} \tag{6.6}$$

where $m_1 = (x_1, y_1, z_1)$ and $m_2 = (x_2, y_2, z_2)$. Each point correspondence gives two constraints :

$$x' = \frac{x_2}{z_2} = \frac{h_{11}x_1 + h_{12}y_1 + h_{13}}{h_{31}x_1 + h_{32}y_1 + h_{33}} \;\; and \;\; y' = \frac{y_2}{z_2} = \frac{h_{21}x_1 + h_{22}y_1 + h_{23}}{h_{31}x_1 + h_{32}y_1 + h_{33}}.$$

Hence,

$$x'(h_{31}x+h_{32}y+h_{33}) = h_{11}x+h_{12}y+h_{13} \; and \; y'(h_{31}x+h_{32}y+h_{33}) = h_{21}x+h_{22}y+h_{23}$$

Therefore, 4 feature correspondences are enough to compute the 8 components of the homography matrix H.

As described in Figure 6.4 the epipolar constraint and the homography matrix can be combined to deal with errors in homography matrix estimation. These errors are computed using the distance of the projected pixel m_2 (respectively m_1) given by the estimated homography H to the epipolar line $I' = Fm_1$ (respectively $I = F^T m_2$) computed using the fundamental matrix.

Assumptions of homography-based registration

The main advantage of homography relationship is that it is independent of the scene structure and the depth of the point to the camera center. However, registering two images using a homography [10] is valid only when the optical centers of the two cameras coincide, or when the imaged scene is planar. Therefore homography holds when there is no camera translation and the effect of parallax is very small. In others words, the homography of a given image should provides the image that would be obtained if the camera is rotated by R. In outdoor urban scenes, these conditions are approximately satisfied very frequently making image registration using homagraphy matrix widely used to deal with moving camera video sequences.

6.3 Motion model estimation for image registration

Based on the geometry relationship between two images, several approaches are proposed in the literature to solve the problem of motion model describing the correspondences between pixels in the images pair. Two widely used approaches namely pixel-based and features-based methods using either all or a set of points correspondences between the images are considered.

6.3.1 Feature-based registration

Feature-based registration approaches aim to align a pair of images based on a matching of sparse set of feature descriptors extracted from each image. These registration methods present the advantage of being potentially faster and more robust against various scene motions. As illustrated in Figure 6.5, feature-

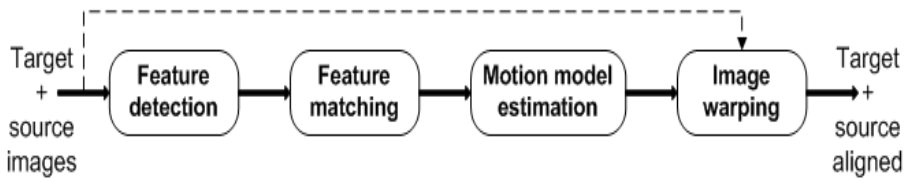


Figure 6.5: **Feature-based registration approach.**

based registration approaches proceed in four main steps. Distinctive key points also named interest points or feature descriptors are first extracted from each image. Then, matching between selected key points is performed to estimate the

geometric transformation or motion model that best provides mapping of one image to the other [92, 93].

6.3.1.1 Feature detection

The distinctive features may correspond to salient regions, corners or contour lines as illustrated in Figure 6.6. A good feature detection algorithm has to be robust to illumination, rotation, scale and viewpoint changes to provide accurate image alignment estimation [94–96]. Several features detection approaches are

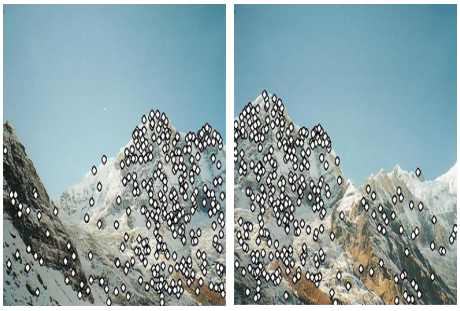


Figure 6.6: **Features detection.**

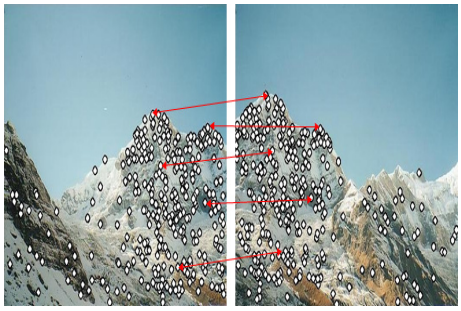


Figure 6.7: **Features matching.**

available in state-of-the-art [97–99]. However, the SIFT [100] and SURF [101] algorithms are the two widely used approaches for this purpose.

SIFT algorithm The Scale Invariant Feature Transform (SIFT) approach computes image key points based on the gradient orientations. First, local orientations are estimated using a histogram of the local gradient orientations. Then, the gradients are copied into different orientation planes, and blurred resampled versions of the images as used as the features. The computed descriptors are then insensitive to small localization errors and geometric distortions [100]. Several extensions of this approach was proposed in the literature with considering the principal component analysis (PCA) of feature descriptors (PCA-SIFT) as proposed in [102] and Gradient Location and Orientation Histogram (GLOH) proposed in [103].

SURF algorithm The Speeded Up Robust Features (SURF) detector [101] approach is much faster and more robust against different image transformations than the SIFT method. This is achieved using an integral image and fast operators in box filters technique to speed-up the computation time. Once the interest points of the image are selected, from the salient features of its linear box-space, a series of images are obtained by the convolution of the initial image with box filters at several scales. It uses an integer approximation to the determinant of Hessian blob detector, which can be computed extremely quickly with an integral image.

6.3.1.2 Features matching

The feature matching step, illustrated in Figure 6.7, aims at computing the most accurate correspondence function between the selected distinctive features. Exhaustively, considering all feature points in one image and looking theirs corresponding in the second image is time consuming. A faster approach can be to find a motion model that satisfies the maximum number of features correspondences. To achieve this aim, most of features matching algorithms consider two computation steps. A preliminary step is performed to select a coarse solution. Then, the second step helps to refine the matching result. The preliminary step is usually performed using a simple strategy that consists in setting a maximum distance threshold according to a given metric (e.g. Euclidean distance). All matches within this threshold are selected. The threshold value is very important since too many incorrect (respectively too few) matches may be kept if the threshold is too high (respectively too low). In the second step which consists in refining the preliminary matching results, several approaches are used in the literature. The RANdom SAmple Consensus (RANSAC) [104] and the Least Median of Squares (LMS) [105] are the most used methods. These approaches start by selecting a random set of correspondences points for which they compute both the corresponding motion transformation and the corresponding residuals of all matching correspondences. The RANSAC approach then counts the number of inliers that are within a given residual threshold. While the LMS approach computes the median value of the residual norm values of all matching set. This second step of matching computation is repeated many times to ensure finding a true set of inliers. Finally, the sample set with largest number of inliers respectively the smallest median residual, is finally returned. A recent version of the RANSAC approach named PROgressive SAmple Consensus (PROSAC) [106] proposes to speed-up the convergence of the algorithm by initializing the random samples from the most confident matches.

6.3.2 Warping

Once the two feature sets and theirs matching correspondences are determined, the geometric mapping function that maps one feature set to another needs to be computed. A linear approach to estimate this function consists in minimizing the sum residuals (Least Square Errors(LSE)) using linear regression. However, linear estimation approach is highly dependent on the scene structure. To deal with this problem, non-linear mapping function based on homography matrix is the widely used approach. As previously mention, this method only depends on the internal parameters of the camera.

6.3.3 Pixel-based method

On the other hand, the pixel-based registration approach, also named direct method, aligns two images based on a analysis of the optical flow of their pixels [107–110]. The optical flow of each pixel $p = (x, y)$ is a function that expresses the displacement of p from one image I_1 to another I_2. In other words, assuming that the two images were captured with the same lighting conditions, the optical flow is the velocity vector $f(\nu)$ given by:

$$I_1(p) \approx I_2(p + f(\nu)) \tag{6.7}$$

In a general context, let $(x(t), y(t), t)$ the trajectory of p in the video sequence, therefore $I(x(t); y(t); t) = constante$, $\Rightarrow$ $\frac{dI}{dt} = 0$ i.e.

$$I_x \frac{dx}{dt} + I_y \frac{dx}{dt} + I_t = 0. \tag{6.8}$$

Where I_x, I_y, I_t are the partial derivatives of I. Hence, the optical flow is given by : $f(\nu) = (\frac{dx}{dt}, \frac{dy}{dt})$.

To be able to solve equation 6.8 a second constraint is necessary. A classical method to solve this problem is to consider local correlations as performed by block matching approach which is widely used in video compression.
Pixel-based alignment algorithms are too slow especially when a full search is performed to explore all possible shifts. To reduce the computation time and to be more robust to noise, a hierarchical coarse-to-fine approach based on image pyramids is classically used. First, a search at the coarser levels is performed [111–113]. Then, the motion estimate from one level is used to initialize a local search at the next finer level of the pyramid.

A simple pixel-based registration approach for video inpainting application was proposed by Patwardhan et al. [3]. The camera motion between each pair of consecutive frames is computed using the median of optical flow of all pixels in the image. This approach may provide correct results only for slow camera motions and viewpoint change. However, it does not generalize well to different types of motions.
An interesting image alignment approach proposed in [114] efficiently register images of large viewpoint changes using a smoothly varying affine field. However, this method is too slow to be considered for video inpainting application. For instance more than 8 minutes are necessary to register 1200 features.

Moreover, camera motion estimation methods based on optical flow may be inaccurate in video sequences with large amount of objects motion or a high scene changes. Analysis of a limited set of local invariant features may be faster and more robust.

6.4 Image registration for video editing applications

In this chapter, we aim to remove viewpoint change between neighboring frames in a free moving sequence using a reliable registration approach. The considered approach should provides accurate alignment results while being fast enough to be considered in a preprocessing step of the video inpainting algorithm.

Pixel-based registration approaches compute a dense point correspondences which helps them to perform better than the feature-based methods with textureless images. However, such images rarely occur in practice. Besides, the direct methods have a limited range of convergence while being time consuming.

On the other side, the main advantage of feature-based registration approaches, is the optimal use of the redundant information in the images, since only sparse set of features are considered in the alignment. Additionally, recent feature detection and matching techniques are robust enough to correctly match correspondences between high viewpoint change in images pair.

For these reasons, we propose to consider, in our video inpainting algorithm of free moving sequences, a feature-based registration approach using homography matrix.

6.4.1 Analysis of multi-label homography-based registration

To register a pair of natural images which are most of the time composed of more than one plane, a single homography cannot provide an accurate alignment for all pixels. A better approach, when dealing with different view changes and camera motions is to consider one homography for each pixel in the image.

Existing homography-based alignment methods strive to determine the best homography transformation on a pixel-by-pixel basis. As illustrated by the algorithm 3, homography candidates are recursively estimated between the source and target images [1, 52, 115, 116]. At each step, an homography matrix is estimated from the correspondences between the sets of key points. Outliers are identified and removed from the estimation. At the next step, the sets of key points corresponding to outliers are used to compute a new homography matrix. The algorithm iterates until either a maximum number of homography candidates is reached or there is no enough features available. Finally, for each pixel p, the homography that minimizes the alignment errors, based on an optimization of a global energy function ξ (see equation 6.9), is chosen. For each pixel p and

homography candidate H_i, the energy function is defined as:

$$\xi(H_i) = \sum_{p \in I_s} E_d(H_i(p)) + \alpha \sum_{\substack{p \in I_s \\ q \in N(p)}} E_s(p, q) \tag{6.9}$$

The data term E_d of the energy represents the registration errors while the smoothness term E_s helps reducing discrepancy between neighboring pixels (p,q) and therefore enforces them to be registered using the same homography candidate. The neighboring system N is usually defined for each pixel p as the 4 nearest pixels (spatially) to p. The parameter α is a weighting scale of the energy terms.

The main difference between existing homography-based registration approaches is mainly the definition of the energy terms to better ensure reliable alignment results. Recent state-of-the-art inpainting algorithms [52] consider a simple approach based on the smoothness term of the energy cost to constraint two neighboring pixels to be registered using the same homography candidate. Authors in [1] added the epipolar constraints between the images to improve the registration quality. Then, a weighted score representing the compatibility of each homography with the fundamental matrix is also considered to compute the accuracy of each homography candidate. However, the two conditions are not sufficient to force each planar region in the images to be registered using a single homography matrix. The authors in [115] proposed an hybrid registration technique based on a piecewise planar region segmentation of the entire video. In a first step, the planes in the scene are detected using the structure from motion analysis algorithm [117]. Then, a MRF-based function is used to find the optimal homography for each grid cell in the image. Despite being robust enough to provide consistent registration results for challenging videos, this method is highly dependent on the structure from motion algorithm which may fail to detect all planes in the scene. Furthermore, two MRF-based energy optimization steps are used for the segmentation and homography selection which make the approach of high complexity. The aforementioned limitations provide the necessary motivations to propose a new fast and accurate registration technique based on the geometry segmentation of each frame. To this end, we start by analysing the performances of existing homography-based registration approaches using ten moving camera video sequences [1] with different motions (slow/fast view change, zooming, camera jitter).

1. The videos are composed of 7 videos provided by [1] and 3 videos from [115]

Algorithm 3: MRF-based registration

input : I_t, I_s
output: $\tilde{I}_{st}$

$F_t \leftarrow featuresExtract\,(I_t)$
$F_s \leftarrow featuresExtract\,(I_s)$
$H_{all} \leftarrow \emptyset$
$i \leftarrow 0$

while $((enoughfeatures)\,\&\&\,(i \prec N_{maxH}))$ **do**
  $\{H_i, outliers\} \leftarrow computeHomography(F_t, F_s)$
  $H_{all} \leftarrow H_{all} \cup H_i$
  $F_t \leftarrow outliers(F_t, H_i)$
  $F_s \leftarrow outliers(F_s, H_i)$
  $i \leftarrow i + 1$
end

$H_{st} \leftarrow optimize\,(H_{all}, \xi(H_{all}))$
$\tilde{I}_{st} \leftarrow register\,(I_s, H_{st})$

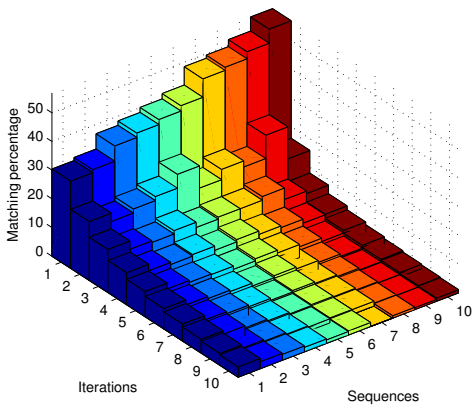
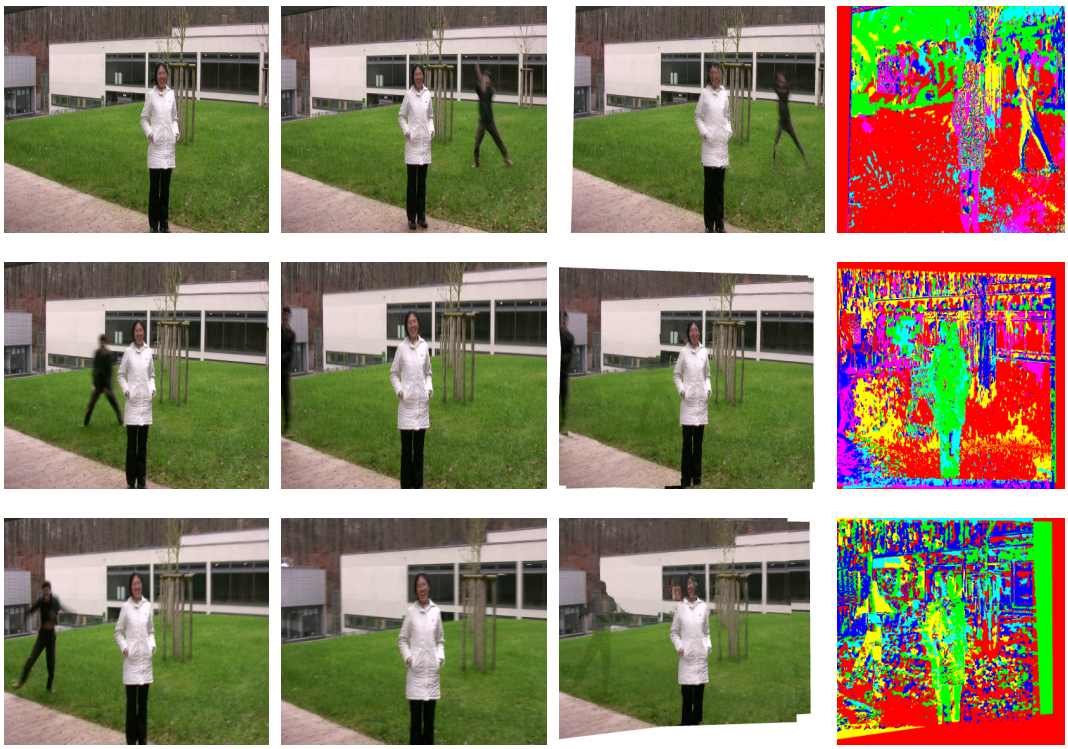


Figure 6.8: **Percentage of matching features for** 10 **video sequences.**

The analysis determines for each iteration the percentage of matched features between each pair of frames distant by 10 frames. This percentage simply represents the number of matched features used to compute a homography H_i in the iteration i. Figure 6.8 illustrates the percentage of matched points per iteration. One can remark that only few homographies matrices (at most 5) are sufficient to ensure almost all features matching between the pair of images. Five homography matrices allow to warp more than 80% of the pixels. Furthermore, Figure 6.9



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

Figure 6.9: **Alignment results using MRF-based optimization.** (a) Target image. (b) Source image. (c) Alignment result of the source to the target image. (d) Labels of homographies used to register each pixel in the source image.

illustrates the registration results of a well-known homography-based registration method using the RANSAC algorithm [104, 118] for homography computation and the expansion-move algorithm [45, 49, 84] for computing the optimized homograpgy per pixel. Five iterations are used in the optimization step. We notice that, if the motion between the two images is slow (first row), the above method provides good alignment results. However, when there is a zoom or when the camera motion is fast and causes important viewpoint changes (second and third rows of Figure 6.9) the alignment errors drastically increase. This can be explained by the fact that considering the registration errors as a main criterion to select the best homography for each pixel may lead to select the homography yielding pixels very similar to target ones. This may lead to wrong results mainly for regions corresponding to moving objects, as illustrated by Figures 6.9 and 6.12 (third row). Finally, Figure 6.9(d) gives a color representation in which one color corresponds to a particular homography matrix. When the motion is rather low (top of Figure 6.9), a rather good segmentation of the image plane is obtained in-

dicating that only one homography matrix is used per region (such as the grass). However, when the motion increases, the number of homographies used per plane drastically becomes much higher leading to less stable and relevant alignment. Moreover, registration results illustrated in Figure 6.12 using a weighted score representing the compatibility of each homography with the fundamental matrix proposed in [1], show that this epipolar constraint is finally not sufficient to provide good alignment.

6.4.2 Proposed region-based registration approach

We propose a new homography-based registration approach that helps to overcome the limitations of existing approaches. The proposed method aims to be well suited to various view changes and faster enough to be easily considered as a preprocessing step in video editing algorithms. As described in Figure 6.10,

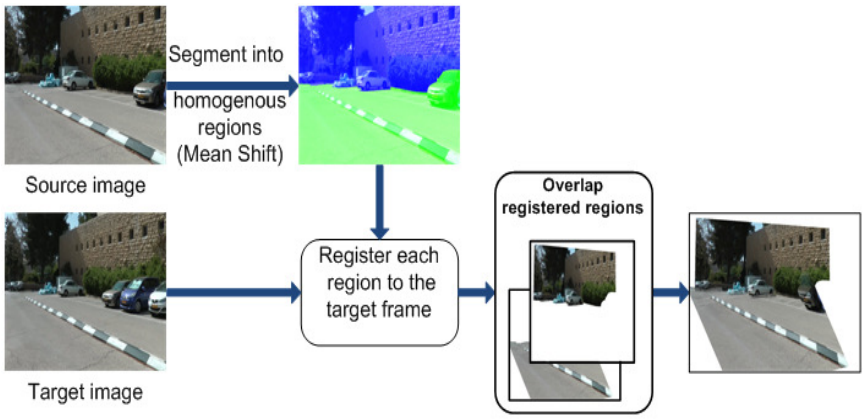


Figure 6.10: **Overview of the proposed registration approach.**

the source image (I_s) is first segmented into homogeneous regions using the mean shift algorithm [119] in order to find the different planes in the scene. We assume here that a plane is homogeneous in terms of colors. Examples of frame segmentation are presented in Figure 6.11 for various moving videos. The regions must be large enough to be associated to a plane and to present enough key points for correspondence matching. In a second step, a homography is estimated for mapping each region of I_s into the target image (I_t). The union of all aligned regions forms the registration of I_s to I_t (noted by I_{st}). Overlapped regions are averaged. Pixels which are not covered by any regions (see for instance the white parts on the left-hand side of the woman in Figure 6.12 second row last column) are not considered in the inpainting process.

A comparison of the proposed method with an implementation of the alignment method proposed in [1] is illustrated in Figure 6.12. One can remark that

(a) (b)

Figure 6.11: **Samples of region segmentation results using Mean Shift** [119]**.** (a) Original images. (b) Images with segmented regions.

our registration approach presents much less artefacts. Besides, registrating each region of the source image separately by using a single homography helps to make the alignment better suited for various viewpoint changes. Furthermore, a post

processing can also be done to process the seam between projected regions.

(a) Target (b) Source (c) Registration proposed in [1] (d) Our registration method

Figure 6.12: **Comparison of registration quality with scenes having different types of camera motion.** First row: slow rotation. Second row: fast view changes. Rows 3 and 4: small view change and zooming. (a) Target image. (b) Source image(to be registered to the target). (c) Registration result using our implementation of registration approach proposed in [1]. (d) Registration result using our proposed registration approach.

6.5 Experiments results

We evaluate the performance of the proposed approach in two main applications: object removal and error concealment. These applications present different specificities. For object removal and error concealment, the spatial location of missing areas are known. However, there is a difference in the shape of the missing areas. In the context of loss concealment, the geometric shape of missing areas is square or rectangle whereas the shape is undetermined for object removal.

All the following results are obtained with the same configuration. The registration applied per region is achieved by using the SURF algorithm [101] with a Hessian threshold equal to 500 and the RANSAC algorithm [118]. Once the frames are aligned, the inpainting is performed by minimizing globally the energy function. The minimization is achieved by the expansion-move algorithm [45, 49, 84]. Five iterations are made. The sliding window is composed of 21 ($M = 10$ in each side) frames and the minimal size of a region computed by the mean shift algorithm is equal to 20% of the frame size. For instance, in average, the number of regions is 2.7 regions per frame.

6.5.1 Object removal

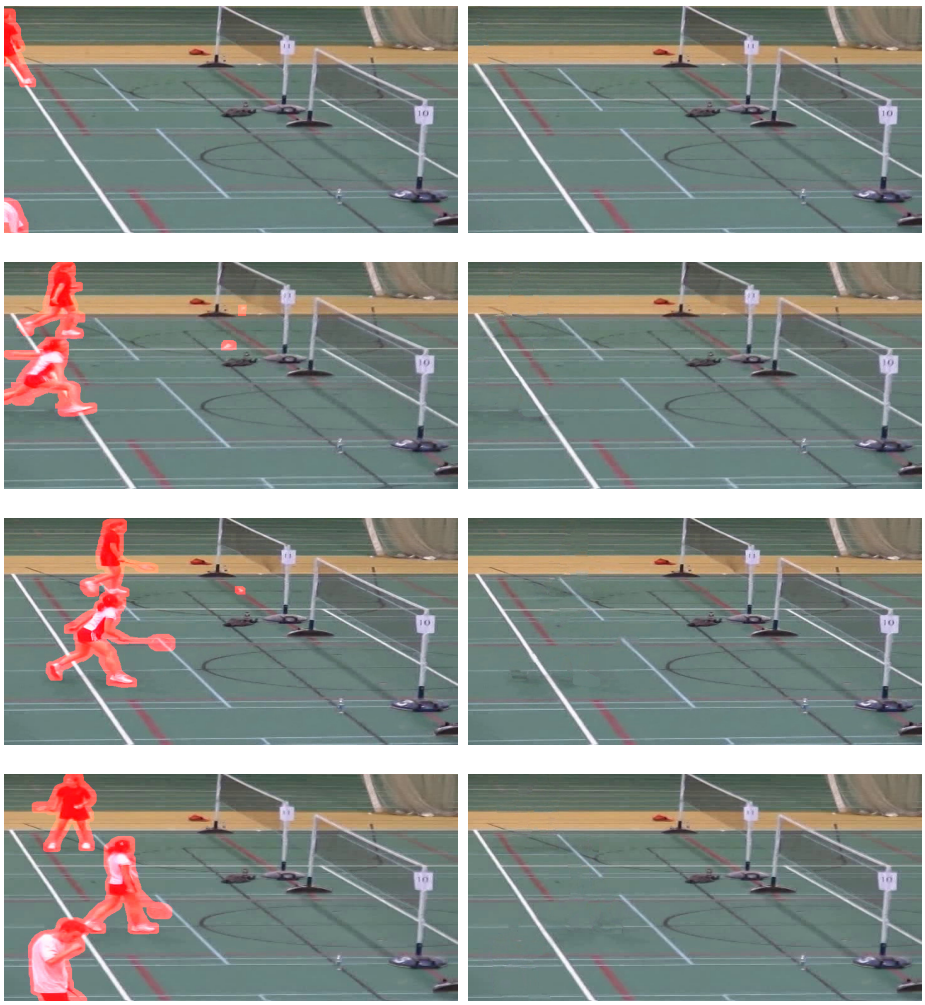
The first experiments assess qualitatively the performance of the proposed method on a set of video sequences in a context of object removal. Two inputs are used in this test: the video to be processed and a binary mask of one or more objects to be removed form the scene. Then, the output of the algorithm is the input sequence without the target objects.

6.5.1.1 Results obtained from Change Detection dataset

Figure 6.13 presents a video sequence captured using a jittering camera (like handheld camera). The red mask in Figure 6.13(a) indicates the moving objects to be removed from the scene. Inpainting results illustrated in Figure 6.13(b) show that the proposed inpainting approach provides high quality of recovering.

6.5.1.2 Comparison with state-of-the art method

The second test consists in comparing the results of the proposed approach to the most recent video inpainting that has been proposed by Granados et al. [1]. This comparison involves 5 video sequences (Figure 6.14) with different camera motions. Each video sequence contains two or three dominant planes and one or two moving objects. In addition to the binary mask indicating the object to be



(a) (b)

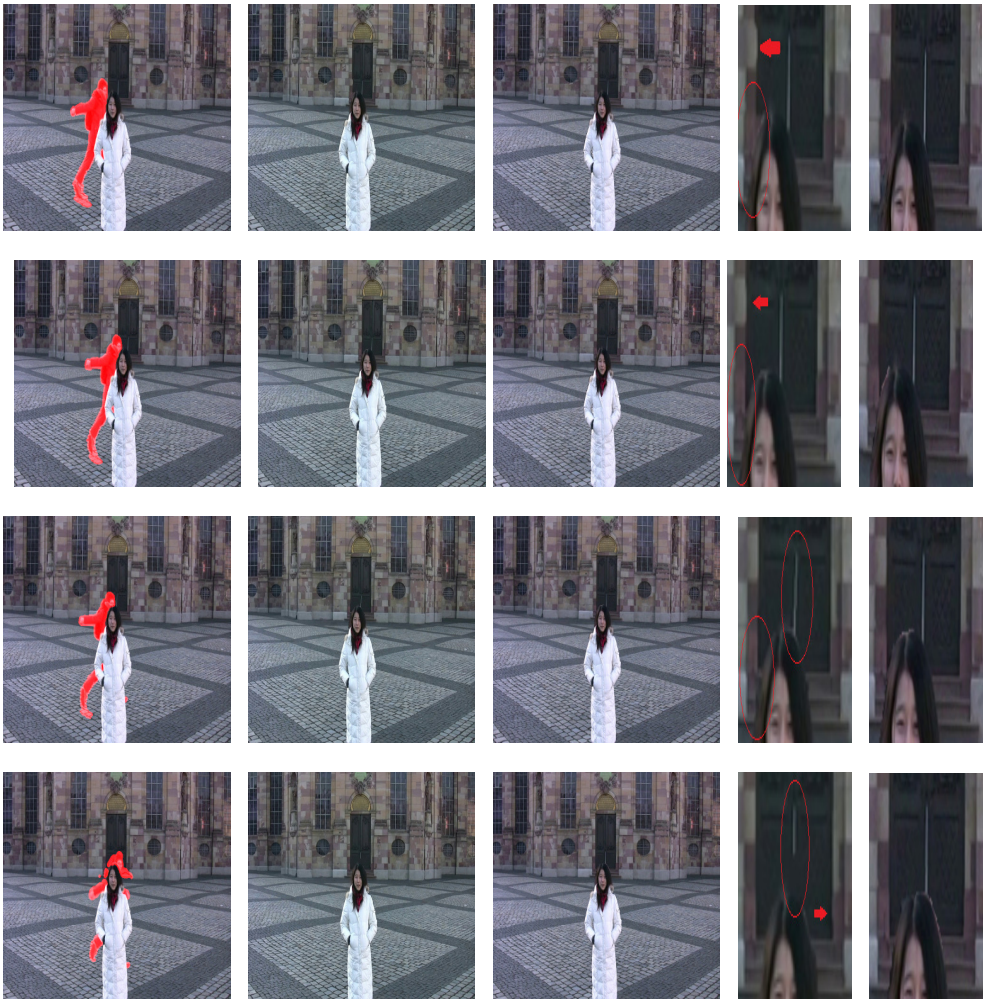
Figure 6.13: **Inpainting result of badmindton video sequence (camera jitter motion).** (a) Original image+mask. (a) Original image+mask. (b) Inpainting results using the proposed approach.

(a) Scene1 (b) Scene2 (c) Scene3 (d) Scene4 (e) Scene5

Figure 6.14: **Test frames of sequences proposed by Granados et al. [1].**

removed from the scene, a second mask is used to indicate the foreground objects. The goal is to prevent the propagation of foreground information into the missing areas.

First, consecutive inpainted frames of a video ($Scene3$) [2] are presented in the



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

Figure 6.15: **Comparison of our inpainting results with Granados's method [1]. Scene3.** (a) Original image+mask. (b) Granados's inpainting results. (c) Inpainting results using the proposed approach. (c) Zoom in Granados's results using [1]. (d) Zoom in our result.

last three rows in Figure 6.15. The first column shows the original images with

2. Videos sequences published by authors in [1] in http://www.mpi-inf.mpg.de/ granados/projects/vidbginp/index.html

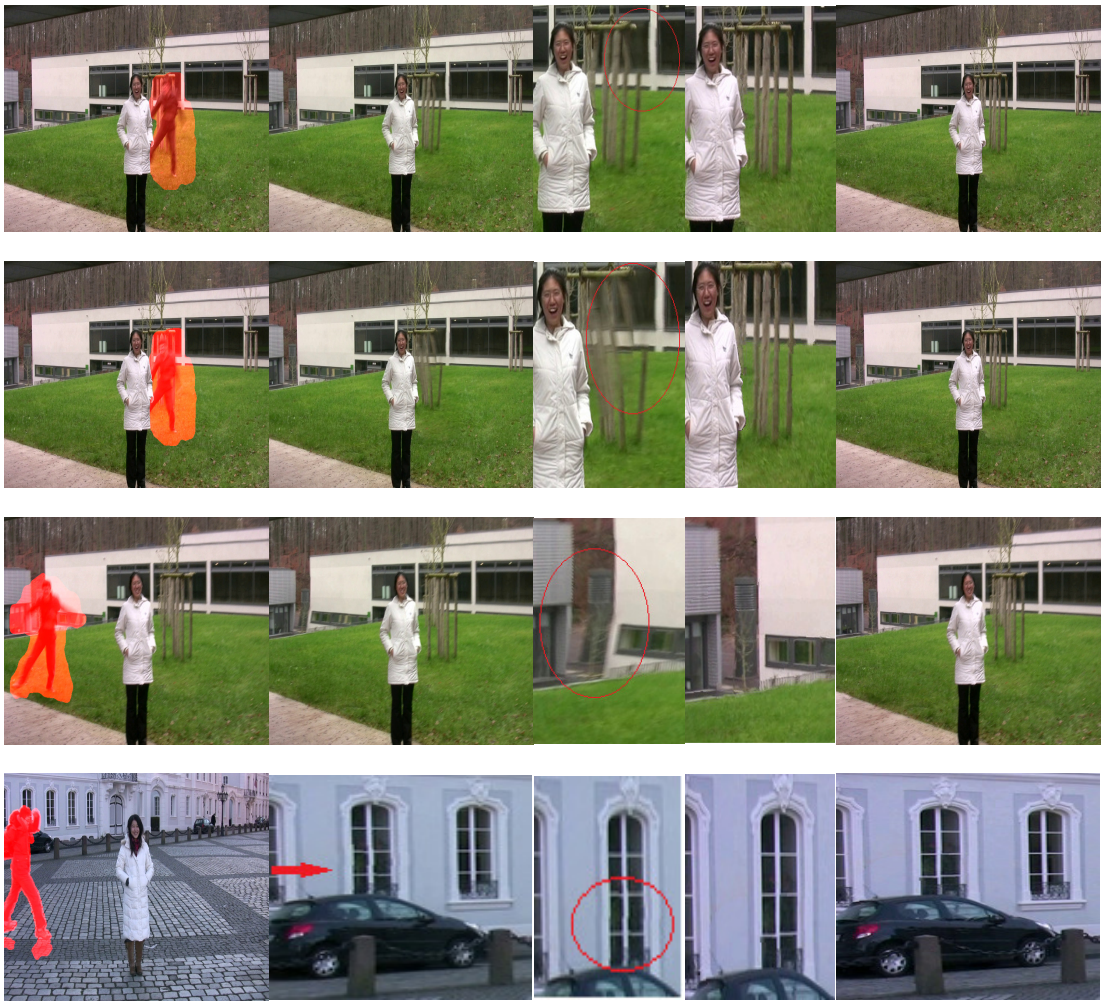
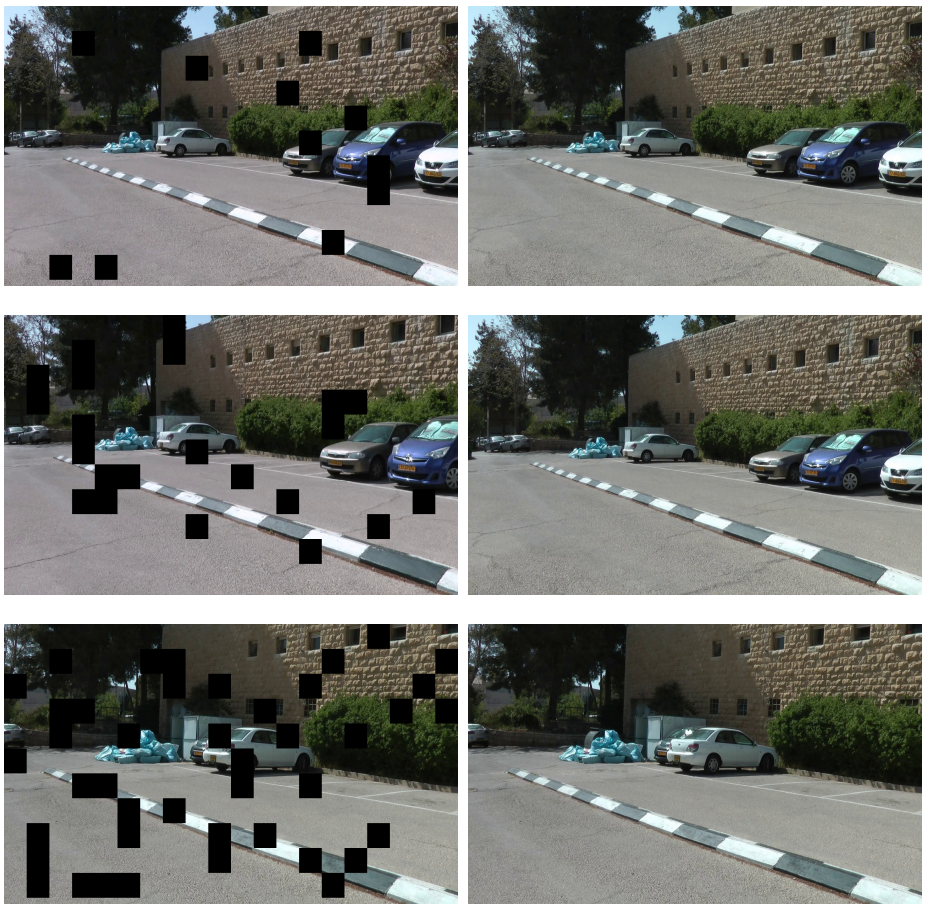


Figure 6.16: **Comparison of our inpainting results with Granados's method [1].** Scene1 and Scene4. (a) Original image+mask. (b) Inpainting results from [1]. (c) Zoom in results from [1]. (d) Zoom in our result. (e) Inpainting results using the proposed approach.

the mask of the object to be removed while the second column presents the corresponding results using the method in [1]. This video presents a small view point variation which makes the alignment easier. However, one can remark that inpainting results of the method in [1] show several temporal artefacts that are not present with our method. Besides, inpainted images with higher view point variations (*Scene*1 and *Scene*4) illustrated in Figure 6.16 are of better quality with our approach than with Granados' method.

6.5.2 Error concealment

To further evaluate the performances of the proposed approach, we evaluate the inpainting quality results in a context of error concealment. In this test



(a) Input image + lost blocks (b) Inpainted results

Figure 6.17: **Test frames from** *Scene*11 **video sequences with varying lost rate.** From top to bottom 5%, 10% and 20% of the 64×64 blocks are lost.

we consider three videos[3] (*Scene*11, *Scene*12 and *Scene*13) with different types of camera motion (zooming, camera jittering and small rotation, respectively). These video sequences have been used for a stabilization method recently proposed [115]. The sequences illustrated in Figure 6.17, 6.18 and 6.19 contain respectively 435, 250 and 360 frames. As summarized in Table 6.1, we consider three loss rates (5%, 10% and 20%) of 64×64 blocks in each video. The inpainting has been tested in function of the number of regions used to perform the frames

3. http://perception.csl.illinois.edu/stabilization/

registration.

As we are in a loss concealment context, it is possible to objectively evaluate the

(a) Input image + lost blocks (b) Inpainted results

Figure 6.18: **Test frames from** *Scene*12 **video sequences with varying lost rate.** From top to bottom 5%, 10% and 20% of the 64×64 blocks are lost.

quality of the reconstruction. For that we compute the PSNR between the original video sequence and the inpainted one. Several conclusions can be drawn from results of Table 6.1. First the number of region used in the registration method plays an important role. The worst quality is obtained when a single homography is used. The gain in terms of quality significantly increases when several regions are considered. The gain is up to 3 dB especially when the camera motion is complex (zooming and camera jittering). For simple camera motion such as small rotation for the sequence *Scene*13, the gain brought by the use of several regions is less significant but still positive compared to the results obtained by a unique homography. When the number of regions increases (more than 4), we

(a) Input image + lost blocks (b) Inpainted results

Figure 6.19: **Test frames from** *Scene*13 **video sequences with variying lost rate.** From top to bottom 5%, 10% and 20% of the 64×64 blocks are lost.

observe a loss of performance for the sequence *Scene*12 which presents camera shake and jittering. These results indicate that the number of regions could be adapted according to the complexity of the camera motion. For rotation and small translation, the number of regions can be relatively high whereas when there are brutal changes in the video, it is better to favor a small number of regions. In our approach we recommend that the region size should be at least equal to 25% of the image size.

6.5.3 Running time

Table 6.2 gives the running time of the proposed approach per frame for different videos. The software is written in C++. Simulations have been performed

Table 6.1: PSNR values of inpainted videos in function of region numbers.

Loss rate	Scene11 (1280 × 720)			Scene12 (640 × 360)			Scene13 (640 × 360)		
	Average number of regions/frame			Average number of regions/frame			Average number of regions/frame		
	1	2.32	4.54	1	2.44	4.9	1	2.45	4.67
5%	27.97	30.9	**31.14**	30.08	**32.01**	23.94	28.47	28.7	**28.75**
10%	27.84	30.63	**30.84**	30.32	**31.45**	26.97	28.59	28.98	**29.05**
20%	27.47	30.06	**30.27**	29.28	**31.02**	24.3	28.16	28.6	**28.79**

Table 6.2: Running time in function of the number of missing pixels.

Video	Resolution	Frames number	Missing pixels	Number of regions	Segmentation time (s)	Inpainting time (s)
Scene1	1440 × 1056	180	16%	2.35	2.71	67
Scene2	960 × 720	270	9%	2.54	1.42	30
Scene3	960 × 720	225	9%	2.92	1.44	35
Scene4	960 × 720	220	11%	2.68	1.98	41
Scene5	1440 × 1056	480	13%	3.03	2.86	59

on a laptop with an Intel Core i7 2.67GHz and 4Go RAM. As the proposed approach is not multithreaded, it just uses one core. In addition, no optimization was made. Parallelizing spatial inpainting and registration may decrease the computation time.

6.6 Conclusion

Several registration approaches are reported in the literature to align images of the same scene. In this chapter, we reviewed the widely used alignment methods. Feature-based registration method based on homography matrix is one of the most used techniques in video processing applications. This approach performs well with different types of video textures showing robust alignment results with low time consuming algorithm. However, an optimization step is necessary to find the best homography for each pixel making the registration approach time consuming. Furthermore, this kind of homography-based registration approach is not well adapted to the scene structure. To solve these problems, we have presented in this chapter a new homography-based registration approach based on segmented image regions. The proposed approach has been experimented into two main applications namely video inpainting and error concealment. Once the

neighboring frames are aligned using the proposed registration approach, inpainting step is then performed using the energy cost proposed in chapter 6 which is globally minimized. The proposed image registration method has drastically reduced the complexity compared to existing methods while providing high quality results.

Chapter 7

Conclusion and Perspectives

In this manuscript, we have placed the video inpainting problem into two main applications: object removal and error concealment. Both of them have the same aim which is efficiently recover space-time holes in video sequences. However, they differ in some constraints. In object removal, the hole form correspond to the object to be removed from the scene. Such hole can be large enough to make producing good quality of inpainting a challenging task. Additionally, one can uses all frames in the video to find the best pixel values that help to correctly recover the missing holes. On the other hand, in error concealment, only the available GOP images should be considered. Besides, the missing blocks which can be randomly spread in the image should be recovered with ensuring the tradeoff between the computational time and the inpainting quality.

7.1 Summary of thesis contributions

To provide an efficient inpainting framework that can be used in both applications, we studied two categories of video inpainting methods. First, we have extended a fast examplar-based video inpainting method into several directions. The extension concerns two main steps of the method which are the patches matching and the sampling of missing pixels.

Regarding the patches sampling step, instead of recovering each target patch in the hole using the estimate provided by a simple Template Matching (TM) approach, a new method based on neighbor embedding techniques have been proposed in Chapter 3. The idea here is to formulate the recovery of unknown pixel values in a patch as a least-squares problem that can be solved using several methods. To make advantages of different techniques, three estimates are computed for each target patch using TM and two neighbor embedding techniques. Then, only the best estimation according to the template (known pixels) is kept.

This chapter has also proposed an application of the video inpainting approach in a context of error concealment. For this aim, a pre-processing step of estimating the missing motion vectors corresponding to lost/erroneous blocks have been proposed. The matching patches are then searched in a motion compensated window which helps to drastically reduce the complexity of the algorithm and limit artefacts due to errors in similarities between patches. The computational time has been further improved by using a multiresolution approach.
Then, Chapter 4 focus on the patch matching step of the examplar-based algorithm. An analysis of well-known similarity metrics usually considered to compute the degree of matching between patches have been proposed. Subjective tests as well as the objective measures of several patches extracted from various natural images showed that the SSD which is the most widely used metric does not provide accurate decision results.

In summary, the proposed contributions of this part helps to provide a fast and efficient inpainting approach able to correctly recover holes corresponding to removed objects or error/lost blocks in static camera scenes by using an efficient combination of motion information and texture redundancies in video sequences. However, the proposed approach still suffers from accuracy and does not provide high quality results especially for large holes due to the lack of global coherency constraint. This case of inpainting large holes in complex scenes have been addressed in Chapter 5. A video inpainting approach based on the optimization of a energy function have been proposed to solve this problem. The proposed energy function helps to constraint the global consistency of inpainted region providing visually pleasing results for both object removal and error concealment applications.

Finally, the problem of inpainting free moving camera sequences have been tackled in Chapter 6. Before inpainting each target frame, its neighboring images have to be first aligned with reference to the target. The proposed image registration approach based on segmented homogeneous regions showed better results than classical homography alignment methods which are usually based on an energy optimization technique.

7.2 Perspectives

Regarding, the neighbor embedding techniques in Chapter 3, the proposed method for video inpainting can be improved in many directions. For instance, several variants of NMF and LLE methods can be found in the literature. These techniques aim to solve the same optimization problem with different algorithms

and constraints. Therefore, a performance analysis of these techniques should provides the better adapted method for video inpainting problem. Additionally, the analysis of similarity metrics shows that the SSD is not the adequate measure for patches matching computation. We think that this analysis can also be extended by proposing a new measure based on the visual coherence which helps to provide better inpainting quality. Besides, the examplar-based algorithm considered here, is based on the matching between 2D patches. A better approach may be to consider space-time patches which should help to limit artefacts by enforcing the temporal coherence.

Furthermore, the proposed graph-based inpainting approach presented in Chapter 6 can be also extended to inpaint missing holes corresponding to either background or moving objects in the scene. The guiding term based on spatial inpainting can be also improved by using space-time offsets which should provide more accurate initialization. The complexity of the approach can be also reduced by improving the NNF searching algorithm and considering a multiresolution approach.

Finally, we think that image registration approach based on both optical flow and homography matrix may provide better results. Such technique helps to avoid the limits of homography-based approaches which fail with textured scenes. Application of the proposed registration approach can be also used for others video processing of moving camera sequences e.g. background segmentation, stabilization, superresolution and so on.

Bibliography

[1] M. Granados, J. Tompkin, K. Kim, O. Grau, J. Kautz, and C. Theobalt, "Background inpainting for videos with dynamic objects and a free-moving camera," in *European Conf. on Computer Vision (ECCV)*, pp. 682–695, 2012. vii, 1, 9, 26, 78, 79, 82, 83, 84, 107, 108, 111, 113, 114, 115, 116, 117

[2] A. Criminisi, P. Pérez, and K. Toyama, "Object removal by examplar-based image inpainting," in *Int. Conf. on Computer Vision and Pattern Recognition (CVPR)*, pp. 721–728, 2003. viii, 19, 20, 25, 34, 35, 39, 50, 53, 69, 71, 72

[3] K. A. Patwardhan, G. Sapiro, and M. Bertalmio, "Video inpainting under constrained camera motion," *IEEE Trans. on Image Proc. (TIP)*, vol. 16, no. 2, pp. 545–553, 2007. 1, 9, 16, 25, 26, 35, 39, 106

[4] T. Shih, N. Tan, J. Tsai, and H.-Y. Zhong, "Video falsifying by motion interpolation and inpainting," in *Int. Conf. on Computer Vision and Pattern Recognition (CVPR)*, pp. 1–8, 2008. 1, 9, 25

[5] Y. Matsushita, E. Ofek, W. Ge, X. Tang, and H.-Y. Shum, "Full-frame video stabilization with motion inpainting," *IEEE Trans. on Pattern Analysis and Machine Intelligence (PAMI)*, pp. 1150–1163, 2006. 1, 9, 25

[6] T. Shih, N. Tang, and J.-N. Hwang, "Exemplar-based video inpainting without ghost shadow artifacts by maintaining temporal continuity," *IEEE Trans. Circuits Syst. Video Techn. (TCSVT)*, pp. 347–360, 2009. 1, 9, 25

[7] Y. Wexler, E. Shechtman, and M. Irani, "Space-time completion of video," *IEEE Trans. on Pattern Analysis and Machine Intelligence (PAMI)*, pp. 463–476, 2007. 1, 9, 24, 26

[8] K. A. Patwardhan, G. Sapiro, and M. Bertalmio, "Video inpainting of occluding and occluded objects," in *IEEE Int. Conf. on Image Proc. (ICIP)*, vol. 2, pp. 69–72, 2005. 4, 24, 35, 39, 50, 52, 53

[9] R. Szeliski, "Image alignment and stitching: A tutorial," in *Handbook of Mathematical Models in Computer Vision*, pp. 273–292, 2005. 7, 98

[10] R. I. Hartley and A. Zisserman, *Multiple View Geometry in Computer Vision*. Cambridge University Press, second ed., 2004. 7, 98, 100, 101, 103

[11] M. Granados, J. Tompkin, K. Kim, O. Grau, J. Kautz, and C. Theobalt, "How not to be seen - object removal from videos of crowded scenes," in *EUROGRAPHICS*, vol. 31, pp. 219–228, 2012. 9, 26

[12] Y. Wang and Q. Zhu, "Error control and concealment for video communication: A review," in *Proceedings of the IEEE*, vol. 86, pp. 974–997, 1998. 9, 29, 50

[13] K. Meisinger and A. Kaupp, "Spatio-temporal selective extrapolation for 3-d signals and its applicationsin video communication," *IEEE Trans. on Image Proc. (TIP)*, vol. 16, no. 9, pp. 2348–2360, 2007. 9, 50, 51, 54

[14] W. Lam, A. Reibman, and B. Liu, "Recovery of lost or erroneously received motion vectors," in *IEEE Int. Conf. Acous. Speech Signal Proc. (ICASSP)*, pp. 417–420, 1993. 10, 32

[15] G. Sullivan and R. Baker, "Motion compensation for video compression using control grid interpolation," in *IEEE Int. Conf. Acous. Speech Signal Proc. (ICASSP)*, vol. 4, pp. 2713–2716, 1991. 10

[16] M. E. Al-Mulla, N. Canagarajah, and D. R. Bull, "Error concealment using motion field interpolation," in *IEEE Int. Conf. on Image Proc. (ICIP)*, pp. 512–516, 1998. 10, 32, 48

[17] M. Turkan and C. Guillemot, "Image prediction based on neighbor embedding methods," *IEEE Trans. on Image Proc. (TIP)*, 2011. 11, 40, 41

[18] M. Bertalmio, G. Sapiro, C. Ballester, and V. Caselles, "Image inpainting," in *ACM Trans. on Graphics (SIGGRAPH)*, pp. 417–424, 2000. 15, 16, 17

[19] M. Bertalmio, L. Vese, G. Sapiro, and S. Osher, "Simultaneous structure and texture image inpainting," *IEEE Trans. on Image Proc. (TIP)*, vol. 12, no. 8, pp. 882–889, 2003. 15

[20] C. Guillemot and O. Le Meur, "Image inpainting: Overview and recent advances," *IEEE Signal Proc. Magazine*, Jan. 2014. 16

[21] M. Bertalmio, A. Bertozzi, and G. Sapiro, "Navier-stokes, fluid dynamics, and image and video inpainting," in *Int. Conf. on Computer Vision and Pattern Recognition (CVPR)*, pp. 355–362, 2001. 17, 26

[22] M. Mengyao, C. Oscar, G. Chan, and M.-T. Sun, "Edge directed error concealment," *IEEE trans. On Circuits and Syst. for Video Techn. (CSVT)*, vol. 20, no. 3, pp. 382–395, 2010. 17

[23] T. Chan and J. Shen, "Mathematical models for local deterministic inpaintings," in *Technical Report CAM TR 00-11, UCLA*, 2000. 18

[24] L. Rudin, S. Osher, and E. Fatemi, "Nonlinear total variation based noise removal algorithms," *Phys. D*, vol. 60, pp. 259–268, Nov. 1992. 18

[25] T. Chan and J. Shen, "Local inpainting models and tv inpainting," in *ACM Int. Conf. on Multimedia*, vol. 62, pp. 1019–1043, 2001. 18

[26] A. Levin, A. Zomet, and Y. Weiss, "Learning how to inpaint from global image statistics," in *IEEE Int. Conf. on Computer Vision (ICCV)*, pp. 305–313, 2003. 18

[27] T. Chan and J. Shen, "Non-texture inpainting by curvature-driven diffusions (cdd)," *J. Visual Comm. Image Rep*, vol. 12, pp. 436–449, 2001. 18

[28] A. Efros and T. Leung, "Texture synthesis by non-parametric sampling," in *IEEE Int. Conf. on Computer Vision (ICCV)*, pp. 1033–1038, 1999. 18

[29] P. Harrison, "A non-hierarchical procedure for re-synthesis of complex texture," in *Int. Conf. Central Europe Comp. Graphics, Visual and Comp. Vision (WCSG)*, 2001. 18, 19

[30] M. Ashikhmin, "Synthesizing natural textures," in *ACM Symp. on Interactive 3D Graphics*, pp. 217–226, 2001. 18

[31] L.-Y. Wei, S. Lefebvre, V. Kwatra, and G. Turk, "State of the art in example-based texture synthesis," in *Eurographics 2009, State of the Art Report, EG-STAR*, 2009. 18

[32] H. Yamauchi, J. Haber, and H.-P. Seidel, "Image restoration using multiresolution texture synthesis and image inpainting," in *IEEE Comp. Graphics Int. (CGI)*, pp. 120–125, 2003. 18

[33] C. Fang and J. J.-J. Lien, "Fast image replacement using multi-resolution approach," in *Asian Conf. Comp. Vision (ACCV)*, vol. 3852, pp. 509–520, 2006. 18

[34] O. Le Meur, J. Gautier, and C. Guillemot, "Eamplar-based inpainting based on local geometry," in *IEEE Int. Conf. on Image Proc. (ICIP)*, pp. 3401–3404, 2011. 20

[35] S. D. Zenzo, "A note on the gradient of a multi-image," in *Comput. Vision, Graphics, Image Proc.*, pp. 116–125, 1986. 20

[36] Z. Xu and J. Sun, "Image inpainting by patch propagation using patch sparsity," *IEEE Trans. on Image Proc. (TIP)*, vol. 19, pp. 1153–1165, May 2010. 20

[37] A. Bugeau, M. Bertalmío, V. Caselles, and G. Sapiro, "A comprehensive framework for image inpainting," *IEEE Trans. on Image Proc. (TIP)*, vol. 19, no. 10, pp. 2634–2645, 2010. 21, 23, 63, 64

[38] O. Le Meur and C. Guillemot, "Super-resolution-based inpainting," in *European Conf. on Computer Vision (ECCV)*, pp. 554–567, 2012. 21, 63, 64

[39] J. Bentley, "Multidimensional binary search trees used for associative searching," *Commun. ACM*, vol. 18, pp. 509–517, Sept. 1975. 21

[40] P. Yianilos, "Data structures and algorithms for nearest neighbor search in general metric spaces," in *Proceedings of the Fourth Annual ACM-SIAM Symposium on Discrete Algorithms*, SODA '93, pp. 311–321, 1993. 21

[41] C. Barnes, E. Shechtman, A. Finkelstein, and D. B. Goldman, "Patchmatch: a randomized correspondence algorithm for structural image editing," in *ACM Trans. on Graphics (SIGGRAPH)*, 2009. 21, 24, 44, 83

[42] S. Korman and S. Avidan, "Coherency sensitive hashing," in *IEEE Int. Conf. on Computer Vision (ICCV)*, pp. 1607–1614, 2011. 22

[43] C. Barnes, E. Shechtman, D. Goldman, and A. Finkelstein, "The generalized patchmatch correspondence algorithm," in *European Conf. Computer Vision (ECCV)*, pp. 29–43, 2010. 22

[44] V. Kwatra, A. Schodl, I. Essa, G. Turk, and A. Bobick, "Graphcut textures: image and video synthesis using graph cuts," in *ACM Trans. on Graphics (SIGGRAPH)*, pp. 277–286, 2003. 22

[45] V. Kolmogorov and R. Zabih, "What energy functions can be minimized via graph cuts?," *IEEE Trans. on Pattern Analysis and Machine Intelligence (PAMI)*, pp. 147–159, 2004. 22, 76, 110, 114

[46] X. Chen, Y. Shen, and Y. Yang, "Background estimation using graph cuts and inpainting," in *Graphics Interface*, pp. 97–103, 2010. 22, 78, 79, 83, 84

[47] N. Komodakis and G. Tziritas, "Image completion using global optimization," in *Int. Conf. on Computer Vision and Pattern Recognition (CVPR)*, pp. 442–452, 2006. 22

[48] Y. Pritch, E. Kav-Venaki, and S. Peleg, "Shift-map image editing," in *IEEE Int. Conf. on Computer Vision (ICCV)*, pp. 151–158, 2009. 22, 24

[49] Y. Boykov, O. Veksler, and R. Zabih, "Fast approximate energy minimization via graph cuts," *IEEE Trans. on Pattern Anal. and Mach. Intell.*, vol. 23, pp. 1222Ű–1239, Nov. 2001. 22, 85, 110, 114

[50] Y. Liu and V. Caselles, "Exemplar-based image inpainting using multiscale graph cuts," *IEEE Trans. on Image Proc. (TIP)*, vol. 22, no. 3, pp. 1699–1711, 2013. 22

[51] K. He and J. Sun, "Statistics of patch offsets for image completion," in *European Conf. on Computer Vision (ECCV)*, pp. 16–29, 2012. 22, 83, 84

[52] O. Whyte, J. Sivic, and A. Zisserman, "Get out of my picture! internet-based inpainting," in *British Machine Vision Conference (BMVC)*, 2009. 22, 78, 79, 83, 107, 108

[53] S. Rane, G. Sapiro, and M. Bertalmio, "Structure and texture filling-in of missing image blocks in wireless transmission and compression applications," *IEEE Trans. on Image Proc. (TIP)*, vol. 12, no. 3, pp. 296–303, 2003. 23

[54] J. Jia and C. keung Tang, "Inference of segmented color and texture description by tensor voting," in *IEEE Trans. on Pattern Analysis and Machine Intelligence (PAMI)*, vol. 26, pp. 771–786, 2004. 23

[55] G. Medioni, M.-S. Lee, and C.-K. Tang, "A computational framework for segmentation and grouping," *Elsevier*, 2000. 23

[56] A. Newson, A. Almansa, M. Fradet, Y. Gousseau, and P. Pérez, "Towards fast, generic video inpainting," in *Proceedings of the 10th European Conference on Visual Media Production*, pp. 7:1–7:8, 2013. 24

[57] Y. Shen, F. Lu, X. Cao, and H. Foroosh, "Video completion for perspective camera under constrained motion," in *IEEE Int. Conf. on Image Proc. (ICIP)*, vol. 3, pp. 63–66, 2006. 24

[58] Y. Hu and D. Rajan, "Hybrid shift map for video retargeting," in *Int. Conf. on Computer Vision and Pattern Recognition (CVPR)*, pp. 577–584, 2010. 24, 77

[59] M. Venkatesh, S. Cheung, and J. Zhao, "Efficient object-based video inpainting.," *Pattern Recognition Letters*, pp. 168–179, 2009. 25, 26

[60] C.-H. Ling, C.-W. Lin, C.-W. Su, H.-Y. M. Liao, and Y.-S. Chen, "Video object inpainting using posture mapping," in *IEEE Int. Conf. on Image Proc. (ICIP)*, pp. 2785–2788, 2009. 25

[61] J. Jia, Y.-W. Tai, T. Wu, and C.-K. Tang, "Video repairing under variable illumination using cyclic motions," *IEEE Trans. on Pattern Analysis and Machine Intelligence (PAMI)*, pp. 832–839, 2006. 25

[62] T. Shiratori, Y. Matsushita, X. Tang, and S. Kang, "Video completion by motion field transfer," in *Int. Conf. on Computer Vision and Pattern Recognition (CVPR)*, pp. 411–418, 2006. 25

[63] M. Khansari and M. Vetterli, "Layered transmission of signals over power constrained wireless channels," in *IEEE Int. Conf. on Image Proc. (ICIP)*, vol. 3, pp. 380–383, 1995. 29

[64] J. Wolf, A. Wyner, and J. Ziv, "Source coding for multiple descriptions," *Bell Syst. Tech. J.*, vol. 59, no. 8, pp. 1417–1426, 1980. 29

[65] J. Panyavaraporn and S. Aramvith, "Error resilient framework using one-pass explicit flexible macroblock ordering map generation and error concealment for h.264/avc wireless video communication," *J. Electron. Imaging.*, August 2011. 31

[66] R. Koenen, "Overview of the mpeg-4 standard," tech. rep. 31

[67] C. Zhu, "Rtp payload format for h.263 video streams," tech. rep. 31

[68] M. Ghanbari, "Postprocessing of late cells for packet video.," *IEEE trans. On Circuits and Syst. for Video Techn. (CSVT)*, vol. 6, no. 6, pp. 669–678, 1996. 33

[69] S. Cui, H. Cui, and K. Tang, "Error concealment via kalman filter for heavily corrupted videos in h.264/avc," *Signal Processing: Image Communication*, vol. 28, no. 5, pp. 430–440, 2013. 33

[70] A. Buades, B. Coll, and J. Morel, "A review of image denoising algorithms, with a new one," *SIAM journal on Multiscale Modeling and Simulation*, vol. 4, no. 2, pp. 490–530, 2005. 34, 69

[71] M. Ebdelli, C. Guillemot, and O. Le Meur, "Examplar-based video inpainting with motion-compensated neighbor embedding," in *IEEE Int. Conf. on Image Proc. (ICIP)*, 2012. 35, 53

[72] M. Ebdelli, O. Le Meur, and C. Guillemot, "Loss concealment based on video inpainting for robust video communication," in *European Signal Proc. Conf. (EUSIPCO)*, 2012. 35

[73] D. Donoho and V. Stodden, "When does non-negative matrix factorization give a correct decomposition into parts?," 2003. 41

[74] D. Lee and H. Seung, "Algorithms for non-negative matrix factorization," in *Advances in Neural Information Proc. Syst. (NIPS)*, 2000. 41

[75] M. Ebdelli, O. Le Meur, and C. Guillemot, "Analysis of patch-based similarity metrics: application to denoising," in *IEEE Int. Conf. Acous. Speech Signal Proc. (ICASSP)*, 2013. 61

[76] Z. Wang and A. Bovik, "Mean squared error: love it or leave it? a new look at signal fidelity measures," *IEEE Signal Proc. Magazine*, vol. 26, pp. 98–117, January 2009. 62

[77] Z. Wang, A. Bovik, H. Sheikh, and E. Simoncelli, "Image quality assessment: From error visibility to structural similarity," *IEEE Trans. on Image Proc. (TIP)*, vol. 13, no. 4, pp. 600–612, 2004. 63

[78] S. Cha, "Comprehensive survey on distance/similarity measures between probability density functions," *Int. journal of Mathematical Models and Methods in Applied Sciences*, vol. 1, no. 4, pp. 300–307, 2007. 63

[79] M. Zontak and M. Irani, "Internal statistics of a single natural image," in *Int. Conf. on Computer Vision and Pattern Recognition (CVPR)*, pp. 977–984, 2011. 65

[80] Z. Wang, A. Bovik, H. Sheikh, and E. Simoncelli, "Dominant orientation templates for real-time detection of texture-less objects," in *Int. Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2010. 65

[81] N. Dalal and B. Triggs, "Histograms of oriented gradients for human detection," in *Int. Conf. on Computer Vision and Pattern Recognition (CVPR)*, pp. 886–893, 2005. 65

[82] S. Cohen, "Background estimation as a labeling problem," in *IEEE Int. Conf. on Computer Vision (ICCV)*, vol. 2, pp. 1034–1041, 2005. 78, 79

[83] M. Granados, H.-P. Seidel, and H. Lensch, "Background estimation from non-time sequence images," in *Graphics Interface*, pp. 33–40, 2008. 78

[84] Y. Boykov and V. Kolmogorov, "An experimental comparison of min-cut/max-fow algorithms for energy minimization in vision," *IEEE Trans. on Pattern Analysis and Machine Intelligence (PAMI)*, vol. 26, pp. 1124Ű–1137, Sep. 2004. 85, 110, 114

[85] N. Goyette, P.-M. Jodoin, F. Porikli, J. Konrad, and P. Ishwar, "Changedetection.net: A new change detection benchmark dataset.," in *Int. Conf. on Comput. Vis. and Pattern Recognit. Workshops*, pp. 1–8, 2012. 86

[86] B. Zitová and J. Flusser, "Image registration methods: a survey," *Image and Vision Computing*, vol. 21, pp. 977–1000, 2003. 97

[87] A. Yao and A. Calway, "Robust estimation of 3-d camera motion for uncalibrated augmented reality," tech. rep., 2002. 97

[88] R. Horn and C. Johnson, *Matrix Analysis*. Cambridge University Press, 1990. 100

[89] J. Stoer, R. Bulirsch, R. Bartels, W. Gautschi, and C. Witzgall, *Introduction to numerical analysis*. Springer, 2002. 100

[90] G. Golub and C. F. Van Loan, *Matrix Computations (3rd Ed.)*. Johns Hopkins University Press, 1996. 100

[91] Y. I. Abdel-Aziz and H. M. Karara, "Direct linear transformation from comparator coordinates into object space coordinates in close-range photogrammetry," in *Symposium on Close-Range photogrammetry*, 1971. 102

[92] J. Gao, S. Kim, and M. Brown, "Constructing image panoramas using dual-homography warping," in *Int. Conf. on Computer Vision and Pattern Recognition (CVPR)*, pp. 49–56, 2011. 104

[93] A. Agarwal, C. Jawahar, and P. Narayanan, "A survey of planar homography estimation techniques," tech. rep., 2005. 104

[94] I. Zoghlami, O. Faugeras, and R. Deriche, "Using geometric corners to build a 2d mosaic from a set of images," in *Int. Conf. on Computer Vision and Pattern Recognition (CVPR)*, pp. 420–425, 1997. 104

[95] T. Kadir, A. Zisserman, and M. Brady, "An affine invariant salient region detector," in *European Conf. on Computer Vision (ECCV)*, pp. 228–241, 2004. 104

[96] A. Bartoli, M. Coquerelle, and P. Sturm, "A framework for pencil-of-points structure-from-motion," in *European Conf. on Computer Vision (ECCV)*, pp. 28–40, 2004. 104

[97] C. Harris and M. Stephens, "A combined corner and edge detector," in *Alvey Vision Conf., Univ. Manchester*, pp. 147–151, 1988. 104

[98] B. D. Lucas and T. Kanade, "An iterative image registration technique with an application to stereo vision," in *Int. Joint Conf. on Artificial Intelligence (IJCAI)*, pp. 674–679, 1981. 104

[99] C. Tomasi and T. Kanade, "Detection and tracking of point features," tech. rep., Int. J. of Computer Vision (IJCV), 1991. 104

[100] D. G. Lowe, "Distinctive image features from scale-invariant keypoints," *Int. J. of Computer Vision (IJCV)*, vol. 60, no. 2, pp. 91–110, 2004. 104

[101] H. Bay, T. Tuytelaars, and L. V. Cool, "Speed-up robust features(surf)," *Computer Vision and and Image Understanding*, pp. 346–359, 2008. 104, 114

[102] Y. Ke and R. Sukthankar, "Pca-sift: A more distinctive representation for local image descriptors," in *Int. Conf. on Computer Vision and Pattern Recognition (CVPR)*, pp. 506–513, 2004. 104

[103] K. Mikolajczyk and C. Schmid, "A performance evaluation of local descriptors," *IEEE Trans. on Pattern Analysis and Machine Intelligence (PAMI)*, vol. 27, no. 10, pp. 1615–1630, 2005. 104

[104] M. Fischler and R. Bolles, "Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography," *Commun. of the ACM*, vol. 24, no. 6, pp. 381–395, 1981. 105, 110

[105] J. Steele and W. Steiger, "Algorithms and complexity for least median of squares regression," *Discrete Appl. Math.*, vol. 14, no. 1, pp. 93–100, 1986. 105

[106] O. Chum and J. Matas, "Matching with prosac - progressive sample consensus," in *Int. Conf. on Computer Vision and Pattern Recognition (CVPR)*, pp. 220–226, 2005. 105

[107] M. Srinivasan, S. Venkatesh, and R. Hosie, "Qualitative estimation of camera motion parameters from video sequences," *Pattern Recognition*, vol. 30, no. 4, pp. 593–606, 1997. 106

[108] S.-C. Park, H.-S. Lee, and S.-W. Lee, "Qualitative estimation of camera motion parameters from the linear composition of optical flow," *Pattern Recognition*, vol. 37, no. 4, pp. 767–779, 2004. 106

[109] T. Zhang and C. Tomasi, "Fast, robust, and consistent camera motion estimation," in *Int. Conf. on Computer Vision and Pattern Recognition (CVPR)*, pp. 1164–1170, 1999. 106

[110] S. Park, H. Lee, and S. Lee, "Qualitative estimation of camera motion parameters from the linear composition of optical flow," *Pattern Recognition*, vol. 30, pp. 767–779, 2004. 106

[111] L. Quam, "Hierarchical warp stereo," in *Image Understanding Workshop*, pp. 149–155, 1984. 106

[112] P. Anandan, "A computational framework and an algorithm for the measurement of visual motion," *Int. J. of Computer Vision (IJCV)*, vol. 2, no. 3, pp. 283–310, 1989. 106

[113] J. Bergen, P. Anandan, K. Hanna, and R. Hingorani, "Hierarchical model-based motion estimation," in *European Conf. on Computer Vision (ECCV)*, pp. 237–252, 1992. 106

[114] W.-Y. Lin, S. Liu, Y. Matsushita, T.-T. Ng, and L. F. Cheong, "Smoothly varying affine stitching," in *Int. Conf. on Computer Vision and Pattern Recognition (CVPR)*, pp. 345–352, 2011. 106

[115] Z. Zhou, H. Jin, and Y. Ma, "Plane-based content preserving warps for video stabilization," in *Int. Conf. on Computer Vision and Pattern Recognition (CVPR)*, pp. 2299–2306, 2013. 107, 108, 118

[116] S. Liu, L. Yuan, P. Tan, and J. Sun, "Bundled camera paths for video stabilization," *ACM Trans. Graph.*, vol. 32, no. 4, 2013. 107

[117] R. Toldo and A. Fusiello, "Robust multiple structures estimation with j-linkage," in *Eur. Conf. on Comput. Vis.*, pp. 537–547, 2008. 108

[118] J. Lee and G. Kim, "Robust estimation of camera homography using fuzzy ransac," in *IEEE ICCSA*, 2007. 110, 114

[119] Y. Cheng, "Mean shift, mode seeking, and clustering," *IEEE Trans. on Pattern Analysis and Machine Intelligence (PAMI)*, vol. 17, no. 8, pp. 346–359, 1995. 111, 112

Publications

[1] M. Ebdelli, O. Le Meur and C. Guillemot, Reduced complexity video inpainting: application to object removal and background estimation, *submitted to IEEE Trans. on Image Process.*, Jan. 2014

[2] O. Le Meur, M. Ebdelli and C. Guillemot, Super-resolution-based inpainting and applications, *IEEE Trans. on Image Process. (TIP 2013)*

[3] M. Ebdelli, O. Le Meur and C. Guillemot, Analysis of patch-based similarity metrics: application to denoising, *in Proc. of IEEE Int. Conf. Acous. Speech Signal Process. (ICASSP 2013)*

[4] M. Ebdelli, O. Le Meur and C. Guillemot, Loss Concealment Based on Video Inpainting for Robust Video Communication, *in Proc. of European Signal Processing Conference, (EUSIPCO 2012)*

[5] M. Ebdelli, C. Guillemot and O. Le Meur, Examplar-based video inpainting with motion-compensated neighbor embedding, *in Proc. of IEEE Intl. Conf. on Image Processing, (ICIP 2012)*

Résumé Cette thèse présente des outils d'inpainting vidéo permettant de reconstruire de manière efficace les zones perdues d'une séquence vidéo. Deux catégories d'approches sont particulièrement étudiées. Dans une premiere étape les approches basées sur l'exemple sont considérées. Différentes contributions ont été proposées. Une application des méthodes de neighbor embedding pour l'approximation des pixels perdus dans un exemple est d'abord considérée en utilisant deux méthodes de réduction de dimensionnalité: la factorisation de matrice non négative (FMN) et le locally linear embedding (LLE). La méthode d'inpainting proposée a été ensuite adaptée à l'application de dissimulation des erreurs en utilisant une étape de pré-traitement d'estimation des vecteurs de mouvement perdus. Une approche multiresolution a également été considérée pour réduire la complexité. Les évaluations expérimentales de cette approche démontrent son efficacité dans les applications de suppresion des objects et de dissimulation des erreurs. Une deuxième catégorie de méthodes de vidéo inpaintinting a été par la suite étudiée en utilisant une approche basée sur l'optimisation globale d'une fonction d'énergie exprimant la cohérence spatio temporelle de la région reconstruite. Enfin, le problème d'inpainting des vidéos capturées par des caméras en mouvement a été étudié. L'alignement des images en utilisant une homographie par région montre de meilleure performances que les méthodes classiques d'alignement par optimisation d'une homography par pixel.

Abstract This thesis presents video inpainting tools to efficiently recover space-time holes in different kinds of video sequences. Two categories of video inpainting approaches are particularly studied. The first category concerns exemplar-based approach. Several contributions have been proposed for this approach. Neighbor embedding techniques have been proposed for patch sampling using two data dimensionality reduction methods: non-negative matrix factorization (NMF) and locally linear embedding (LLE). The proposed framework have been also adapted to the error concealment application by using a preprocessing step of motion estimation. A multiresolution approach has been considered to reduce the computational time of the method. The experimental evaluations demonstrate the effectiveness of the proposed video inpainting approach in both object removal and error concealment applications. The video inpainting problem has been also solved using a second approach based on the optimization of a well-defined cost function expressing the global consistency of the recovered regions. The camera moving videos has later been takled by using a region-based homography. The neighboring frames in the sequence are aligned based on segmented planar regions. This method has been shown to give better performance compared to classical optimized pixel-based homography.