Image registration : algorithms and applications

Raj Kumar Gupta

2013

Raj Kumar Gupta. (2013). Image registration : algorithms and applications. Doctoral thesis,
Nanyang Technological University, Singapore.

<https://hdl.handle.net/10356/54860>

<https://doi.org/10.32657/10356/54860>

IMAGE REGISTRATION: ALGORITHMS AND APPLICATIONS

RAJ KUMAR GUPTA

School of Computer Engineering

A Thesis submitted to the Nanyang Technological University
in fulfillment of the requirement for the degree of

Doctor of Philosophy

2013

Abstract

Image registration is the task of determining positions of corresponding points in two images. A fast and accurate image registration plays an important role in many computer vision and graphics problems. This thesis presents fast and robust pixel and feature-based image registration algorithms and explores new applications in computer vision and graphics. It is divided into two parts. In the first part, we propose a fast and accurate pixel-based image registration algorithm that registers a pair of stereo images to extract the three-dimensional information of the scene. The algorithm finds an appropriate match for each reference image pixel in the target image to compute dense depth maps and works very well in case of repetitive patterns, object boundaries, as well as in occluded and non-textured image regions without increasing computational overhead significantly. The three-dimensional information obtained using pixel-based registration is then used to recognize complex human activities. A sequence of depth maps is represented as a sequence of codewords that are learnt to discriminate between activities. These discriminative sub-sequences are then used to classify and localize an activity in a given test video. To demonstrate the efficiency of these algorithms, we provide experimental results on standard datasets which are publicly available. In the second part, we present a feature-based registration method that registers different image features to find most similar images from very large-scale image datasets. The proposed system retrieves and ranks database images based on their similarity to any given input image. By registering different image features, the retrieval and indexing of the relevant images is improved. We use these similar images to solve

an ill-posed computer graphics problem, viz., colorization of grayscale images. The algorithm uses similar color images as the reference images and transfers color information to a given input grayscale image using a graph-based optimization method. The algorithm works at the resolution of superpixels and uses local properties for proper color transfer. Our use of superpixels reduces the complexity of our algorithm significantly and it also empowers the colorizations to exhibit a much higher extent of spatial consistency in the colorization as compared to those using independent pixels. We evaluate these approaches on a wide variety of images. Both qualitative and quantitative analyses have been used to demonstrate the greater effectiveness of the proposed approaches.

Acknowledgements

Foremost, I would like to express my sincere gratitude to my advisor Prof. Deepu Rajan for the continuous support of my PhD study and research, for his patience, motivation, enthusiasm, and immense knowledge. His guidance helped me in all the time of research and writing of this thesis. Besides him, I would like to thank Prof. Cho Siu-Yeung David for his support during my initial years of PhD study. I could not have imagined having better advisors and mentors for my PhD study.

I would like to acknowledge the financial, academic and technical support of the Nanyang Technological University and its staff, particularly in the award of a NTU Research Scholarship that provided the necessary financial support for this research. I would also like to thank Institute for Infocomm Research (I^2R) for their collaborations on different research projects.

I thank my fellow labmates in ForSe lab for many enlightening chats: Alex Yong-Sang Chia, Arfika Nurhudatiana, Deepak Subramanyam, Dilip Kumar Prasad, Le Ha Than, Pravin Kakkar, Tang Chao Ying and Zhang Hengyi. I thank Christina Lee for providing such a wonderful environment to work. I faithfully appreciate her help offered over the last four years.

I would like to thank Amit Sachan, Arun Kumar, AV Subramanyam, Bappaditya Mandal, Chan Kin Sun Frodo, Derrick Chan Wei Shih, Ee Sin Ng, Eng How Lung, Fahad Hameed Ahmad, Huang Zhiyong, Hui Yan Cheah, Li Xiao Jie, Liu Di, Liu Yan Zhu, Manish Narwaria,

Rubha Shri Narayanan, Steve Lin, Shaojie Zhuo, Shah Atiqur Rahman, Wei Hua and Yian Ling for their help during these years. I really appreciate their support.

Last, but by no means least, I would like to thank my family members, especially my parents and my friends Alex Yong-Sang Chia, Dilip Kumar Prasad, Krishna Agarwal, Phi Minh Thuy for their support and encouragement to persue this degree. Without them, I would not have finished the degree.

Contents

List of Figures	vi
List of Tables	ix
1 Introduction	1
1.1 Image Registration	1
1.1.1 Problem Definition	3
1.1.2 Components of Image Registration	4
1.1.3 Classification of Registration Methods	6
1.2 Motivation	8
1.3 Objective	8
1.4 Contributions	9
1.5 Thesis Outline	10
2 Pixel-based Registration in Stereo Images	12
2.1 Introduction	12
2.2 Related Work	15
2.3 Proposed Algorithm	19
2.3.1 Initial Matching	21
2.3.2 Unreliable Pixel Detection	23
2.3.3 Disparity Interpolation	24
2.3.4 Disparity Refinement	25
2.4 Experimental Results and Analysis	27
2.4.1 Quantitative Results	27
2.4.2 Qualitative Results	31

CONTENTS

2.5	Conclusion	37
3	Human Activity Detection using Depth Images	43
3.1	Introduction	43
3.2	Related Work	45
3.3	Depth image segmentation	48
3.3.1	Surface normals extraction	50
3.3.2	Spatio-gram computation and matching	51
3.3.3	Segmentation results	54
3.4	Codebook of human poses	54
3.4.1	Computing pose descriptors	56
3.4.2	Spatial-based clustering to learn codewords	57
3.5	Discriminative subsequences of codewords	59
3.5.1	Aligning xSC to a test window codewords	62
3.5.2	Learning discriminative xSC	64
3.6	Activities classification and localization	68
3.6.1	Classification results	68
3.6.2	Localization results	76
3.7	Conclusion	76
4	Feature-based Registration for Similar Image Search	79
4.1	Introduction	79
4.2	Related Work	81
4.3	Proposed System for Image Retrieval	84
4.3.1	Automatic Image Segmentation	85
4.3.2	Shape-based Registration	86
4.3.3	Feature-based Registration	88
4.3.4	Combining Similarity Metrics	91
4.4	Experimental Results	92
4.4.1	Qualitative Evaluation	92
4.4.2	Quantitative Evaluation	97
4.5	Limitations	99
4.6	Conclusion	99

5	Image Colorization using Similar Images	100
5.1	Introduction	100
5.2	Related Work	103
5.3	Algorithm	105
5.3.1	Internet image selection	106
5.3.2	Super-pixel representation	107
5.3.3	Graph-based Optimization	108
5.3.4	Weight Sampling and Result Selection	110
5.4	Experimental Results	113
5.4.1	User Study	116
5.5	Conclusion	122
6	Conclusions and Future Works	125
6.1	Conclusions	125
6.2	Future Works	127
	Publications	129
	References	131

List of Figures

1.1	Applications of image registration	2
1.2	Image registration	4
1.3	Image registration steps	5
1.4	Thesis outline	10
2.1	Stereo images	13
2.2	Multiple matching windows	16
2.3	Adaptive binary window	18
2.4	Block diagram	20
2.5	Problems in disparity selection	22
2.6	Unreliable pixel detection	24
2.7	Disparity interpolation	25
2.8	Algorithm computation time	28
2.9	Quantitative comparison with other algorithms	29
2.10	Error for different window size	32
2.11	Intermediate results	33
2.12	Results on the Middlebury 2003 dataset	35
2.13	Results on our dataset	36
2.14	Streaking artifacts	37
2.15	Results on the Middlebury 2005 & 2006 dataset	38
2.16	Results on the Middlebury 2005 & 2006 dataset - <i>cont'd</i>	39
2.17	3D Reconstruction	40
2.18	3D Reconstruction - <i>cont'd</i>	41
3.1	Silhouettes ambiguities in poses	44

LIST OF FIGURES

3.2	Template-based approach	46
3.3	Problem with depth segmentation	49
3.4	Surface normals	50
3.5	Spatioogram matching	53
3.6	Results of depth image segmentation	55
3.7	Pose descriptor	56
3.8	Pose descriptor	57
3.9	Image clustering	58
3.10	Cluster analysis	60
3.11	Cluster medoids	61
3.12	Sequence matching	65
3.13	Sequence alignment	66
3.14	Stereo image dataset	69
3.15	Confusion matrix for stereo image dataset	70
3.16	Activity dataset	71
3.17	Classification scores and confusion matrix	73
3.18	Example training sequence	76
3.19	Key frames from test sequence	77
3.20	Activity localization results	78
4.1	Images on Internet	80
4.2	System pipeline	84
4.3	Automatic image segmentation	85
4.4	Shape registration	87
4.5	Shape matching results	87
4.6	Intensity histogram computation	88
4.7	Texture feature representation	90
4.8	Dense SIFT feature representation	91
4.9	Results of similar image search	93
4.10	Comparison with other methods	94
4.11	Comparison with other methods - <i>cont'd</i>	95
4.12	Comparison with other methods - <i>cont'd</i>	96
4.13	Quantitative Comparison	98

LIST OF FIGURES

5.1	Image colorization	101
5.2	Scribble-based colorization	103
5.3	Example-based colorization	104
5.4	System pipeline	106
5.5	Color transfer process	107
5.6	Image oversegmentation	108
5.7	Color transfer between images	109
5.8	Results with different weight settings	111
5.9	Results with different weight settings - <i>cont'd</i>	112
5.10	Hierarchical result selection	114
5.11	Graphical user interface	114
5.12	Comparison with <i>intrinsic colorization</i>	115
5.13	Comparison with other methods	117
5.14	Comparison with other methods - <i>cont'd</i>	118
5.15	Colorization with different Internet examples	119
5.16	Colorization results	120
5.17	Colorization results - <i>cont'd</i>	121
5.18	Example images shown during user study	122
5.19	Colorization using different keywords	123

List of Tables

2.1	Comparison of the proposed method with other real-time algorithms listed on Middlebury evaluation table for absolute disparity error > 1	30
2.2	Comparison of the algorithm speed (MDE/s) and the implementation platform.	30
2.3	Quantitative results obtained with and without using small correlation window W_s by the proposed algorithm.	31
3.1	<i>Precision</i> and <i>Recall</i> scores of <i>Naive Classifier (multi-class support vector machine)</i> , <i>One-level MEMM</i> model, <i>Two-level MEMM</i> model, and <i>Our method</i> in each environment.	75
4.1	Computation time required by different algorithms to retrieve similar images for a given input image from a small database of 1000 images.	98
5.1	Correspondence between the colorfulness attribute and the colorfulness metric.	113
5.2	Compares the user study results obtained by using different colorization methods. The cell values represents the number of times the subjects classified the artificially colorized images as real color images.	122

LIST OF TABLES

Chapter 1

Introduction

1.1 Image Registration

Image registration or image alignment is the process of determining the positions of corresponding points in two images. These images may be taken from different viewpoints, at different resolutions, and at different times or in different environments. Image registration has received significant attention of researchers from different fields due to its diverse and wide range of applications in the areas of remote sensing (e.g., multispectral classification, integrating information into geographic information systems (GIS)), computer vision (e.g., stereo matching, object recognition) and medical image analysis (e.g., monitoring tumor growth, combining computer tomography (CT) and nuclear magnetic resonance (NMR) data).

By registering two images, the three-dimensional information of a scene can be determined, changes in the scene can be identified, objects in an image can be recognized, and the fusion of multi-modality information also becomes possible. Figure 1.1 shows applications of two-dimensional image registrations. The first row shows two color images of a scene taken from different perspectives and the three-dimensional reconstruction of the scene. The three-dimensional reconstruction has been performed by finding the displacement between positions of the corresponding pixels in both color images. In the second row, Magnetic resonance imaging (MRI) and Single-photon emission computed tomography (SPECT) images are fused together for better visualization and interpretations of these images.

1. Introduction

Figure 1.1: Applications of image registration - Different applications of image registration: (a) viewpoint registration for 3D reconstruction from stereo images, (b) multimodel registration to fuse medical images for better visualization, (c) template matching for object recognition and, (d) temporal registration for urban growth monitoring.

The third row demonstrates another type of image registration where the correspondence has been extended to the object level for object recognition. Here, the registration problem becomes more difficult since both images contain different instances of the same object category. In the last row, image registration has been used for a remote sensing application to identify changes in urban areas over the time.

Image registration can be carried out either using pixel properties or using features of regions. In pixel-based registration, we use stereo image datasets to establish the correspondence between images taken from different viewpoints. For feature-based registration, we use large image datasets (e.g. Internet images) to perform an object level registration to find and align similar instances of objects in two different scenes. In this thesis, our primary goal is to develop fast, robust and computationally efficient image registration algorithms and explore their new application areas. Along with these registration algorithms, this thesis also presents two such applications where the proposed image registration algorithms play a vital role to solve real-world computer vision and computer graphics problems.

1.1.1 Problem Definition

Image registration involves finding the mapping between two images both spatially and with respect to intensity as shown in Figure 1.2. Given two images I_1 and I_2 that are to be registered. The mapping between these images can be denoted as:

$$I_1(x, y) = g(I_2(f(x, y))) \quad (1.1)$$

where f is a transformation that maps spatial coordinates (x, y) to new spatial coordinates (x', y') ,

$$(x', y') = f(x, y) \quad (1.2)$$

and g is a radiometric transformation. The process of image registration essentially involves the estimation of the optimal spatial and intensity transformations f and g by maximizing an alignment measure (or, minimizing a misalignment measure) so that the images are matched properly. The intensity transformation (g) is not always needed for image registration.

1. Introduction

Figure 1.2: Image registration - Definition of image registration

Alignment Measure The similarity between two images can be determined using different alignment measures. The *sum of squared differences* (SSD) is one of the most common similarity measure used by many registration algorithms [1, 2]. SSD is calculated by subtracting pixels within a square neighborhood between the reference image I_1 and the target image I_2 followed by the aggregation of squared differences and is given as:

$$SSD = \sum_{(i,j) \in \eta} [I_1(x+i, y+j) - I_2(x'+i, y'+j)]^2, \quad (1.3)$$

where η is the square neighborhood that has been used for matching. The other popular alignment measures are: *sum of absolute difference* (SAD) [2, 3], *normalized cross correlation* (NCC) [4, 5], *mutual information* (MI) [6, 7] etc.

1.1.2 Components of Image Registration

In general, the majority of image registration algorithms consist of the following four components [8, 9] (see Figure 1.3):

- **Feature Detection:** To register two images, image features such as edges, corners, closed-boundary regions, contours, etc are manually or, preferably, automatically detected. Pixel-based image registration algorithms do not comprise of this step.

Figure 1.3: Image registration steps - Basic steps of image registration process.

- **Feature/Region Matching:** In this step, the correspondence between the reference image features/regions and the target image features/regions is established. To find the correspondence, similarity measures along with spatial relationship are used.
- **Transform Model Estimation:** With the help of feature correspondence established between images during feature matching, the type and parameters of transformation functions, aligning the target image with the reference image, are estimated.
- **Image Resampling & Transformation:** By using the transformation model estimated in the previous step, the target image is resampled to the geometry of the reference image.

1. Introduction

1.1.3 Classification of Registration Methods

There is no unified method or general theory that has been established for image registration. Depending on the data and the problem to be solved, each application area has developed its own approach and implementation which exploits image pixels or features or a combination of both to establish the correspondence between two images [10, 11, 12, 13]. However, we can broadly divide registration methods into four categories [8, 9, 14]: (1) viewpoint registration, (2) template matching, (3) multimodal registration, and (4) temporal registration. Each of these categories along with their typical applications is discussed below.

Viewpoint registration The objective of viewpoint registration algorithms is to register images of the same scene taken from different viewpoints [11, 15, 16, 17]. To find the correspondence between two images, this type of method often uses assumptions about viewing geometry and surface properties to minimize the search space. These methods exploit image pixels or features or a combination of both to establish the correspondence between two images. Typical applications of viewpoint registration are to provide a larger field of view [11, 15], three-dimensional scene reconstruction [16, 17] to capture the actual shape and appearance of real objects with the help of stereo images, etc. Figure 1.1(a) shows such an example where first two images¹ (of the same scene) are taken from slightly different viewpoints. With the help of viewpoint registration, we can compute the actual shape of the objects and their distance from the camera as shown in the last image. The gray value of a pixel represents its distance from the camera.

Template matching Template matching algorithms are used to find a reference pattern in an image [18, 19, 20]. These algorithms mostly perform feature-based image registration and often use known properties of objects with pre-selected image features for high-level matching. Template registration algorithms find applications in various areas such as character recognition [21], signature verification [22], automatic quality inspection [23], scene interpretation, content-based

¹Source: <http://vision.middlebury.edu/stereo/data/>

or object searching [19, 24], map updating etc. An example of such image registration has been shown in Figure 1.1(c) where the correspondence has been extended to the object level. The reference and target images contain different object instances from same object category. The image registration algorithms find or align the object present in the target image using the template shown in the reference image. Finding such correspondence between images is a difficult task due to intra-class variations, occlusion, clutter background and 3D pose changes.

Multimodal registration Multimodal registration algorithms often use pixel and/or feature based registrations to find the correspondence among images of a scene acquired by using different sensors [25, 26, 27]. Images acquired from different sensors display complementary and shared information about the object in images with different intensity maps. The primary goal of multi-modal image registration is to achieve enhanced and detailed image information by integrating the images obtained from various kinds of sensors. For example, in medical imaging [28], the CT and MRI images of the brain capture a better view of bones and soft tissues, respectively. By integrating both of these images, better visualization of the anatomical structure of the brain can be obtained. Figure 1.1(b) presents such an example where first two images show an orthogonal cross-section of a magnetic resonance (MR) brain image and Single-photon emission computed tomography (SPECT) brain image, respectively. MR images visualize the anatomy well while SPECT images show the function well. By registering these two images, it is possible to anatomically locate brain regions of abnormal function. The image obtained after such registration is shown in the last column.

Temporal registration Temporal registration algorithms are used to register images of same scene taken under different conditions or at different times [29, 30]. These algorithms also use pixel and/or feature based registrations to find the correspondence between images. Temporal registration methods have various applications in remote sensing (e.g. urban growth monitoring and planning, natural resource monitoring etc.) and medical imaging (e.g. detecting and monitoring tumor growth, Digital Subtraction Angiography (DSA) etc). Cloud removal is

1. Introduction

another application where the observations of several days can be used to create cloud-free data. Figure 1.1(d) shows remotely sensed imageries of an urban area taken in year 2003 and 2004, respectively. By registering these two remotely sensed imageries, the changes in an urban area can be determined as shown in the last column.

1.2 Motivation

Image registration is vital for a wide variety of applications and over the years, a broad range of algorithms have been developed to address various types of data and applications. These methods vary in their robustness, complexity and sophistication. The recent advancements in computational and storage capacities of computers have also fueled interest in registration algorithms and opened the doors for new application areas in computer vision [15, 31, 32, 33, 34] and computer graphics [35, 36, 37]. Although the last decade has seen a lot of progress in image registration techniques, the new applications which are computationally very expensive, demand accurate and real-time computation of data [31, 32] or require processing of very large image datasets [15, 34, 35, 36, 37] limit the applicability of existing image registration methods. Along with the computational power, the computational complexity of new applications is also increasing proportionally. These challenging applications will always be constrained by the limitation of physical resources, such as space, time and bandwidth.

1.3 Objective

This thesis focuses on development of pixel and feature-based image registration algorithms and explores new applications in computer vision and graphics. The two main objectives of the thesis are:

- to develop a pixel-based image registration algorithm that registers a pair of stereo images to extract three-dimensional shape information and to apply the algorithm for classification and localization of human activities. A pixel-based approach will be useful here to compute dense disparity maps.

- to explore the role of feature-based image registration in improving the performance of image retrieval of perceptually similar images in response to a query image and to use retrieval results to solve a grayscale image colorization problem. Due to the variations in size, shape, and position of objects in natural images, a feature-based approach will be useful.

1.4 Contributions

In this thesis, our contributions are two fold. We present algorithms of pixel and feature-based image registration along with their applications in areas of computer vision and graphics. Our main contributions are:

- A pixel-based image registration algorithm that registers a set of stereo images to extract the three-dimensional scene information. We use this three-dimensional scene information to recognize and localize human activities from a sequence of depth images.
- A novel depth image segmentation method to segment the objects that are very near to the image background by using only their depth information. This is used in the human activity recognition task mentioned above. We also propose a spatial-based hierarchical clustering approach that computes more compact and less noisy image clusters as compared to other state-of-the-art clustering approaches.
- A system that performs feature-based registration to retrieve most similar images from very large-scale image datasets (e.g. Internet images) and aligns and ranks these retrieved images based on their similarity to any given input image.
- An application that uses similar images to colorize grayscale images. We propose a superpixel based feature extraction and optimization technique to reduce the computational complexities of the algorithms.

Figure 1.4: Thesis outline - This thesis presents algorithms and applications of pixel-based and feature-based image registrations.

1.5 Thesis Outline

As shown in the figure 1.4, this thesis is organized as follows:

Chapter 2. Pixel-based Registration in Stereo Images: We introduce a fast and computationally efficient pixel-based image registration algorithm to register a pair of stereo images. This chapter provides an overview of existing algorithms, discusses the proposed algorithm and evaluates its performance on standard datasets. The qualitative and quantitative comparisons with existing state-of-the-art real-time stereo matching algorithms have been done to demonstrate the effectiveness of the proposed method.

Chapter 3. Human Activity Detection using Depth Images: We present an application of stereo image registration where we use the three - dimensional scene information to identify the human activities. This chapter provides an overview of the related work followed by the proposed algorithm to identify the human activities. We also present novel algorithms for depth images segmentation and clustering. To evaluate the performance of the proposed algorithm, this chapter also presents the experimental results on our own as well as the standard human activity detection dataset.

Chapter 4. Feature-based Registration for Similar Image Search: In this chapter, we present a similar image retrieval system that performs feature-based registration to retrieve most similar images from very large-scale image datasets (e.g. Internet images) and aligns and ranks these retrieved images based on their similarity to any given input image. A detailed comparison with other state-of-the-art algorithms have been made to show the performance of the proposed algorithm.

Chapter 5. Image Colorization using Similar Images: We present an application that uses similar images obtained after performing feature-based image registration to solve an ill-posed computer graphics problem which involves colorization of grayscale images. In this chapter, we used the similar images retrieved from the Internet image dataset to colorize the gray images. A user study along with a detailed comparison with other interactive and automatic colorization techniques have been performed to demonstrate the greater effectiveness of the proposed algorithm.

Chapter 6. Conclusions and Future Works: We summarize our work and discuss about the significance of the proposed algorithms along with further research directions not addressed in this thesis.

Chapter 2

Pixel-based Registration in Stereo Images

In this chapter, we present a fast and computationally efficient pixel-based image registration algorithm to register a pair of stereo images. We provide an overview of existing algorithms, discuss the proposed algorithm and evaluate its performance on standard datasets. Qualitative and quantitative comparisons with existing state-of-the-art real-time stereo matching algorithms are done to demonstrate the overall performance and efficiency of the proposed algorithm.

2.1 Introduction

Depth perception is one of the important tasks in many computer vision and computer graphics systems such as image-based rendering [38], robot navigation in the real environment [39, 40], object detection [40], human computer interaction (to detect three-dimensional motion information of a user) [41] etc. A fast and accurate matching at object boundaries is necessary for proper rendering and reconstruction of an object in a virtual environment. By registering a pair of stereo images (pair of images taken from slightly different viewpoints as shown in Figure 2.1), the depth (or disparity) of various points in a scene relative to the position of a camera can be determined. The 3D reconstruction problem can be viewed as an application of image registration [14] which consists of registering two images that have been taken from slightly different viewpoints by finding a

Figure 2.1: Stereo images - A pair of stereo images that have been captured from different point of views.

set of points in one image that can be identified as the same points in another image.

A large number of algorithms have been proposed to compute dense disparity map. A detailed overview on these stereo matching algorithms can be found in [42, 43, 44]. In general, stereo matching algorithms can be divided into two categories: local methods and global methods.

Local algorithms are statistical methods and are usually based on correlation. Local stereo matching algorithms can be subdivided into two broad categories: area-based algorithms and feature-based algorithms. Area-based algorithms use neighboring pixels of a given pixel to find the suitable match with another image by using the intensity values of the pixels. These algorithms mostly use normalized cross-correlation (NCC) [2] or sum of absolute differences (SAD) [45, 46] or sum of squared differences (SSD) [1, 10, 47, 48] technique during the window matching process. The performance of area-based algorithms is highly influenced by the shape and size of the image region used during the matching process. Feature-based algorithms rely on feature extraction and match local cues (e.g. edge, corners). Although, these algorithms work very fast, they generate sparse disparity maps.

Global algorithms make explicit smoothness assumption and then solve it through various optimization techniques [49, 50, 51, 52, 53, 54]. Most of the global algorithms are based on energy minimization [49, 51, 52, 54]. Global algo-

2. Pixel-based Registration in Stereo Images

gorithms give very good results but these methods are computationally expensive which makes them impractical for real-time systems. In the past few years, many efforts have been made to develop real-time algorithms for disparity estimation. A large number of these algorithms are based on dynamic programming [50]. They perform scanline-based optimization for all scanlines in the image independently. While dynamic programming produces better results at occluded image regions, the inconsistency between scanlines leads to the well-known streaking artifacts. This problem has been solved recently by multiple-label dynamic programming [53, 55]. However, it still lacks accuracy. Felzenszwalb and Huttenlocher [49] proposed an efficient belief propagation algorithm that uses a hierarchical approach to reduce the computation time and memory usage of belief propagation approach. Despite the reduction, the algorithm still requires a lot of memory to form different hierarchical levels.

In this chapter, a fast correlation-based stereo matching method is presented that computes accurate disparity map from a stereo pair of images in real-time. The proposed algorithm uses two correlation windows (one large and one small size) to compute the disparity map. While large correlation window gives good results at non-textured image regions, the small window improves the performance at depth discontinuities. Unlike the previous algorithms [1, 10, 48, 56, 57] that evaluate multiple matching windows at each image pixel, here we have shown that the use of two matching windows is enough to achieve accurate disparity estimations. Rather than evaluating the second correlation window at each image pixel, we evaluate it near the depth discontinuities for a selected disparity values only. By restricting the search space, we are able to achieve good performance without increasing the algorithm’s computation time significantly. To demonstrate the performance of the proposed algorithm, we have evaluated our algorithm on several standard datasets [58] along with our own dataset.

The rest of the chapter is organized as follows: In Section 2.2, we briefly cover the related literature; Section 2.3 describes the proposed algorithm. Section 2.4 contains the qualitative and quantitative results on the Middlebury datasets (2003, 2005 and 2006) and a detailed comparison with other real-time stereo matching algorithms. The last section concludes this chapter.

2.2 Related Work

Correlation-based methods find the corresponding match by measuring the similarity between two image areas. Many common correlation-based methods use Cross Correlation (CC) or the Sum of Squared or Absolute differences (SSD or SAD). The performance of these methods is strongly influenced by the size and shape of the matching image area. Usually, rectangular matching windows are used to achieve high computational performance [45, 47, 48, 56, 59]. The size of the matching window determines the number of pixels to be used for correlation. For reliable disparity estimation, the matching window must be large enough to cover enough intensity variations, but small enough to cover only the pixels having the same disparity value. This requirement raises the need for different shape and size of the windows at different pixels within the same image as no fixed window size works well. While a large size window blurs the object boundaries, a small size window results are unreliable in low textured image regions. The pixels near to the disparity discontinuities require windows of different size and shape to avoid crossing the disparity.

Kanade and Okutomi [1] proposed an adaptive window-based method which starts with an initial estimation of the disparity map and updates it iteratively for each point by choosing the size and shape of a window till it converges. It uses intensity and disparity variance to choose the window with the least uncertainty. This method is sensitive to the initial disparity estimations and the iterative nature of this method increases the computational overhead significantly. Binaghi *et al.* [60] used the zero mean normalized cross correlation (ZNCC) as matching cost and used a neural network (NN) model to decide the proper window shape and size for each support region. The least-mean-square delta rule has been used for training. The results obtained by this method are satisfactory, but due to the use of neural network to learn the windows of proper shape and size, the computational complexity of this method increases significantly.

Mordohai and Medioni [61] proposed a complex area-based method that utilizes both monocular and binocular cues. An initial matching was performed by using a combination of normalized cross correlation techniques which requires the evaluation of multiple different size windows during the matching. To select the

2. Pixel-based Registration in Stereo Images

Figure 2.2: Multiple matching windows - Multiple windows used by Fusiello *et al.* [10] for matching.

correct match at each image pixel, they used the tensor voting framework. The matches are then grouped into smooth surfaces and the disparities for the unmatched pixels are assigned so as to ensure smoothness in terms of both surface orientation and color. While this method achieves good accuracy, the evaluation of multiple windows and surface similarity step to group the matches into a smooth surface increase the computational complexity of this method.

Muhlmann *et al.* [62] used the sum of absolute differences (SAD) correlation measure in RGB color space for a fixed size rectangular window. They used the left to right consistency and uniqueness constraints and applied a fast median filter to the results. While they achieved high speed, the quality of disparity maps greatly affected by the size of the matching window. Boykov *et al.* [63] proposed a variable window algorithm which computes variable size matching windows based on connected components of constant disparity. The maximum computational complexity of different steps of this algorithm is $O(wdN)$ where w , d and N are the window size, maximum disparity and the number of image pixels respectively, which makes it suitable for a real-time implementation. However, this method suffers from a different systematic error. In some cases, it increases the object size by including nearby low texture image regions.

Fusiello *et al.* [10] proposed a simple multiple window approach. For each pixel and disparity, the correlation with a number of windows (see Figure 2.2) is performed and it retains the disparity with the best correlation value. The

evaluation of multiple windows to find the correct disparity value makes this algorithm computationally very expensive. Hirschmuller *et al.* [59] proposed a multiple window-based approach that uses different size windows, mainly focused on reducing the errors at depth discontinuities. The algorithm uses a border correction filter to improve matches at object borders. Many other multiple window approaches have been proposed which use multiple windows of different sizes or use windows of different orientations to compute the matching cost [48, 56, 57]. Although these approaches use simple window matching approach which has the computational complexity of $O(wdN)$, the use of multiple matching windows of different sizes at each image pixel increases the computational cost of these methods significantly.

Di Stefano *et al.* [46] proposed a method based on the uniqueness constraint which rejects previous matches as soon as better ones are detected. In contrast to bidirectional matching algorithms their method performs only single matching phase and achieves almost similar performance. While their method is computationally very efficient and produces dense disparity maps, it fails near depth discontinuities and blurs the object boundaries. Zach *et al.* [64] proposed a hierarchical disparity estimation algorithm that uses a fix size matching window at each hierarchical level. While their method achieved very high frame rate, it performed poor near the depth discontinuities. Zhao and Taubin [65] focused only on foreground moving objects and used background modeling and dense matching with multi-resolution adaptive windows to compute the disparity maps. Tombari *et al.* [66] used of a segmentation-based costs aggregation strategy to overcome the problem near the depth discontinuities.

Veksler [67] proposed an algorithm which chooses the appropriate window shape by optimizing over a large class of compact window by using the minimum ratio cycle algorithm. The window cost is computed as the average window error and bias to larger windows. While this method works very well, it is not efficient enough for a real-time system. In [68], Veksler introduced another approach that computes the correlation values for several different window sizes at the pixel of interest and selects the window size with least matching errors. However, this algorithm needs a number of user defined parameters for the matching cost computation.

2. Pixel-based Registration in Stereo Images

Figure 2.3: Adaptive binary window - An example window used by [69] for matching. (a) shows the pixel (highlighted in blue) for which the adaptive binary window has been computed, and (b) shows adaptive binary window used for matching.

Yoon and Kweon [70] proposed a locally adaptive support weight approach which computes the support weights for each pixel in the support window based on their color dissimilarity and the spatial distance from the center pixel. These weights regulate the pixel's influence in the matching process. In another paper, the same authors [71] proposed a pre-processing step for correspondence search in the presence of specular highlights where they compute specular-free two-band images for given input images. By using these images with their previous method, they further improved the accuracy of disparity maps. Psota *et al.* [72] proposed an iterative refinement on the results of adaptive support weight approach to further improve the accuracy of the depth maps. Rather than just considering the color difference and the distance to the center pixel, Hosni *et al.* [73] used a geodesic approach to compute the support weights which enforce the pixels connectivity. The pixels that have low geodesic distance from the matching window's center are assigned high support weights and therefore large influence in the matching process. While these approaches give very good results, they are computationally expensive and their performance degrade in the presence of image noise. The computational complexities of these methods are $O(w^3 dN)$ and the reported computation time [70, 73] of these algorithms on a fast machine (AMD 2700+ machine with support window size 35×35) is around one minute which makes them impractical for real-time systems. Yu *et al.* [74] reduced the compu-

tation time of adaptive-weight aggregation methods by using an exponential step information propagation which propagates the information from long distance pixels within a few iterations. In another attempt, Hosni *et al.* [75] significantly reduced the computation time by using the guided filter [76].

Gupta and Cho [69] used local color information to compute adaptive binary windows (Figure 2.3) of different shapes and sizes for matching. To speed up the algorithm, rather than computing the disparity value for each image pixel, it assigns the same disparity value to all pixels that participated during matching. The algorithm works very well at depth discontinuities, however the computational cost of this algorithm increases significantly in dense textured image regions.

Ogale and Aloimonos [77] considered the shape of the objects depicted and demonstrated the importance of the vertical and horizontal slanted surfaces. Rather than using the standard uniqueness constraint, the authors referred to pixels with a uniqueness constraint that belong to line segments along a scanline. Thus, their method performs interval matching instead of pixel matching. They obtained a stretching factor by computing the slants of the surfaces along a scanline and then the matching is performed based on the absolute intensity difference. While they achieve very good performance, their algorithm is computationally very expensive. Humenberger *et al.* [78] proposed a census-based algorithm that uses a sparse census mask which reduces the computation time significantly. Zhang *et al.* [79] further improved the runtime and accuracy of the algorithm by using mini-census transform and the cross-based cost aggregation.

2.3 Proposed Algorithm

As discussed in the previous section, a fixed size correlation window fails to perform both in low textured image regions as well as at object boundaries. To deal with this problem, a number of approaches have been proposed that use multiple windows. In this section, we show that the two correlation windows are sufficient to compute reliable disparity maps. Here, we proposed an algorithm that uses one large and one small size correlation windows to compute the disparity map. Figure 2.4 illustrates the block diagram of the proposed algorithm. The algorithm

2. Pixel-based Registration in Stereo Images

consists of the following four processing modules: (1) Initial matching; (2) Unreliable pixel detection; (3) Disparity interpolation; (4) Disparity refinement. In the initial matching step, we compute the initial disparity map by using two different sizes of correlation windows W_l and W_s , respectively. In the unreliable pixel detection step, we use left-right consistency check to identify unreliable pixels. The left-right consistency check enforces the uniqueness constraint and identifies those pixels which have unreliable disparity. In the disparity interpolation step, we estimate the disparity values for unreliable pixels identified by left-right consistency check [47]. In the final step, we refine the disparity map by using the reference image to improve the accuracy at depth discontinuities. We assume rectified stereo image pair as an input i.e. the epipolar lines [80] are aligned with the corresponding scanlines.

Figure 2.4: Block diagram - Basic steps of the proposed algorithm.

2.3.1 Initial Matching

To compute the initial disparity map, we use the sum of absolute difference (SAD) based matching approach by using a large correlation window W_l . The matching cost $C(x, y, d)$ of pixel (x, y) for disparity value d is given as follows:

$$C(x, y, d) = \sum_{i=-\omega/2}^{i=\omega/2} \sum_{j=-\omega/2}^{j=\omega/2} |I_l(x+i, y+j) - I_r(x+i, y+j-d)| + \xi, \quad (2.1)$$

where $I_l(x, y)$ and $I_r(x, y)$ are the intensities of the pixel in left and right image, respectively. ω represents the size of the matching window W_l . Figure 2.5 shows the SAD characteristics under different scenario of matching costs. In Figure 2.5(a), only one minimum matching cost can be found so the disparity can be determined uniquely as d . However, if more than one minimum matching cost is computed by the matching window as shown in Figure 2.5(b), which is common in non-textured image regions, it becomes very hard to determine the correct disparity values of the pixels. We use the disparity values of neighboring pixels to determine the disparity of such pixels. We propose a penalty term ξ based on gradient and the disparity value of neighboring pixel in the same scanline to estimate the disparity value in non-textured image regions accurately. The penalty term ξ is given as:

$$\xi = T(|d - d'|) \left(1 - \frac{|I_l(x, y) - I_l(x, y')|}{255} \right), \quad (2.2)$$

d' is the disparity of the left or the right neighboring pixel (x, y') in the same scanline and T is the smoothing parameter. The value of ξ changes according to the change in image gradient. It increases in low gradient image regions (e.g. low or non-textured areas) and decreases in high gradient image regions (e.g. depth discontinuities)

$$1 \geq \left(1 - \frac{|I_l(x, y) - I_l(x, y')|}{255} \right) \geq 0. \quad (2.3)$$

Figure 2.5(c) demonstrates the SAD values computed for different disparity values in a non-textured region of the Tsukuba image [58] at point (205, 230). We can observe that there is ambiguity due to several local minima. This problem can be solved by using gradient and disparity values of the neighboring pixels.

2. Pixel-based Registration in Stereo Images

Figure 2.5: Problems in disparity selection - In (a), the disparity can be easily determined as d due to unique minimum value. It becomes ambiguous in case of multiple minima's as shown in (b). (c) shows the matching cost calculated at point (205,230) of the Tsukuba image [58]. (d) displays the matching cost calculated by using the penalty term ξ for the same image point. The disparity can be uniquely determined in (d).

Figure 2.5(d) demonstrates the matching cost computed by using equation (2.1). We can see that the disparity value can be easily determined now by using *winner-takes-all* approach. The matching costs C_l and C_r are computed for both left and right neighbors of a pixel respectively. The disparity of a pixel (x, y) can be computed as:

$$d_c(x, y) = \min \left(\arg \min_d C_l(x, y, d), \arg \min_d C_r(x, y, d) \right). \quad (2.4)$$

After computing the disparity values using the large correlation window, we use a small correlation window W_s near the depth discontinuities. The use of a large correlation window blurs the object boundaries and the actual positions of these boundaries are usually within the distance of half the size of the correlation window [59]. As compared to the larger windows, the smaller windows do not overlap the depth discontinuities to the same extent. Hence, the use of a smaller correlation window helps us to reduce this problem. Here, we compute the disparity values of all such pixels that reside near the depth discontinuities within the range of half the size of large correlation window W_l by using a small correlation window. While computing the disparity of such pixels, we restrict the evaluation of the small correlation window only within those disparity values that are carried by neighboring pixels. The disparity of such pixels can be computed as:

$$d_c(x, y) = \arg \min_{d \in N} C(x, y, d), \quad (2.5)$$

where $d_c(x, y)$ is the disparity of the pixel (x, y) and N represents the disparity values of the neighboring pixels. Since we are computing the disparity of the pixels that reside near the depth discontinuity regions, we compute the cost $C(.)$ without using the penalty term ξ as mentioned in Equation (2.1). Restricting the disparity search within the disparity values of the neighboring pixels not only helps to avoid false local minima, but it also speeds up the matching process.

2.3.2 Unreliable Pixel Detection

The left-right consistency check is a very effective way to detect unreliable pixels. The left-right consistency check is based on uniqueness constraint [81] that assumes the one-to-one mapping between stereo image points. We compute left

2. Pixel-based Registration in Stereo Images

and right initial disparity maps by using the method described in Section 2.3.1 by choosing the left and right image as reference image, respectively. For each pixel of the left disparity map, we check whether it carries the same disparity value as its matching point in the right disparity map. A valid correspondence should match in both directions. A simple test of left-right cross checking can be written as:

$$|d_l(x, y) - d_r(x, y'')| < 1, \quad (2.6)$$

where (x, y) and (x, y'') are the correspondence pair of pixels and $d_l(x, y)$ and $d_r(x, y'')$ are left and right disparities for the points (x, y) and (x, y'') , respectively. All the pixels that fail to satisfy Equation (2.6) are marked as unreliable pixels. Figure 2.6 shows such pixels that have been detected as unreliable in the Tsukuba image.

Figure 2.6: Unreliable pixel detection - Unreliable pixels (shown in black) detected by left-right consistency check.

2.3.3 Disparity Interpolation

In section 2.3.2, the left-right consistency check filters out occluded pixels as well as unreliable pixels. In this step, we assign new disparity values to all such pixels with the help of their reliable neighboring pixels. For each unreliable pixel, we search for pixels with reliable disparity value in its 8 neighboring pixels. In absence of any reliable pixels, we increase the search area until we find any reliable pixel. For all these reliable pixels, we compute the distance between the intensities

of unreliable pixel and its reliable neighbor in the reference image. The distance between two points p and q can be calculated as:

$$D(p, q) = \sqrt{(r_p - r_q)^2 + (g_p - g_q)^2 + (b_p - b_q)^2}, \quad (2.7)$$

where r_p, g_p, b_p and r_q, g_q, b_q are the intensity values of points p and q respectively. We assign the disparity value of the reliable pixel which has the minimum distance from unreliable pixel in reference image as shown in Figure 2.7.

Figure 2.7: Disparity interpolation - Disparity interpolation to unreliable pixels. (a) shows the pixel p that has unreliable disparity value, (b) shows the neighboring pixel q that has similar intensity value as of p but contains reliable disparity value.

2.3.4 Disparity Refinement

To remove outliers for each color segment, the plane fitting method is widely used [82, 83, 84, 85]. It is assumed that the neighboring pixels which have the same intensity values will also have the same disparity values. While this method increases the accuracy of the algorithm, it requires the color segmented image as an input which makes it computationally expensive. Here, we use the disparity refinement approach that has been proposed in [69]. The algorithm (see *Algorithm 1*) uses the color information of the reference image to refine the disparity map without performing any image segmentation. The algorithm uses the reference image color information to update the disparity of pixel p with the minimum disparity value between pixel p and the most similar pixel to p in a small one-dimensional window.

2. Pixel-based Registration in Stereo Images

Algorithm 1 Disparity map refinement by using the color information of the reference image.

```
1: for each scanline in reference image do
2:   for all  $p$  in scanline do
3:      $index \leftarrow 0$ ;
4:      $mdif \leftarrow \infty$ ;
5:     for  $q$  from  $(p - w)$  to  $(p + w)$  do
6:       if  $p \neq q$  then
7:          $dif \leftarrow d(p, q)$ ;
8:         if  $mdif > dif$  then
9:            $index \leftarrow q$ ;
10:           $mdif \leftarrow dif$ ;
11:        end if
12:      end if
13:    end for
14:     $disp(p) \leftarrow \min(disp(p), disp(index))$ ;
15:  end for
16: end for
```

2.4 Experimental Results and Analysis

To demonstrate the effectiveness of the proposed algorithm, we have applied it on several standard datasets along with our own dataset. The Middlebury stereo benchmark dataset [58] has been used to evaluate the performance of the proposed algorithm. In our experiment, the small window size W_s is chosen as 3×3 for all test images while the large window size W_l is set to 9×9 . For all the experiments shown in this chapter, we fixed the value of T as 8. The proposed algorithm has been implemented using C language and compiled with GNU gcc compiler performing under a 3 GHz computer running Linux operating system. We used the sliding window approach proposed in [2] to speed up the matching cost computation of large correlation window W_l . The serial CPU implementation of the proposed algorithm achieved more than 8 million disparity evaluations per second while the parallel CPU implementation was able to achieve more than 10 million disparity evaluations per second. However, the computation time of the algorithm depends on the window size.

2.4.1 Quantitative Results

The maximum computational complexity of different steps of the proposed algorithm is $O(wdN)$ where w , d and N are the window size, maximum disparity and the number of pixels in an image. Figure 2.8 shows the computation time (in milliseconds) of the proposed algorithm for different window sizes. The computation time is estimated for the serial CPU implementation of the proposed algorithm on the Tsukuba image. To compute the algorithm computation time, for a fixed window size and a stereo pair, we run the algorithm 100 times and then take an average of total execution time. It has been shown that the increase in window size also increases the computation time of the proposed algorithm. However, the real-time performance can be achieved for a large range of matching windows of different size. The quantitative performance of the proposed method for standard Middlebury dataset (Tsukuba, Venus, Teddy and cones) is summarized in Table 2.1. The values shown in Table 2.1 represent the percentage of bad pixels with an absolute disparity error greater than one for different regions: they are non-occluded (*nocc*), whole image (*all*) and pixels near discontinuities

2. Pixel-based Registration in Stereo Images

(*disc*). The last column of the table shows the overall performance of the algorithm for all four images. In the table, the results of the proposed algorithm is listed as *Results-1*. *Results-2* shows the results with the same parameters without performing the disparity refinement (Section 2.3.4). As shown in Table 2.1, the proposed algorithm is ranked at third place out of eight local real-time stereo matching algorithms. Table 2.2 compares the algorithm speed in terms of million disparity estimations per second (*MSD/s*) with other competitive real-time algorithms. Here, we report the computation time of the parallel CPU implementation of our algorithm which can be easily speed up by 10x to 15x with the help of GPU [86]. It can be also seen that the FPGA implementations are very fast but generate less competitive results in terms of quality. Note that the proposed algorithm uses the rectangular windows for matching which also makes it suitable for FPGA implementation.

Figure 2.8: Algorithm computation time - Shows the computation time of the proposed algorithm for different window sizes W_l (7×7 to 21×21) on the Tsukuba image (image size 384×288 with 16 disparity labels).

Figure 2.9 compares the performance of the proposed algorithm with other fixed size window-based correlation algorithms. It shows the percentage error in whole image (*all*) computed on the Tsukuba image. For the Tsukuba image pair,

Figure 2.9: Quantitative comparison with other algorithms - Compares the performance of the proposed algorithm with other correlation-based algorithms. The graph shows the percentage error in whole image (*all*) on the Tsukuba image.

we compared the proposed algorithm with standard correlation-based algorithms (SAD, SSD and NCC) and other window-based fast stereo matching algorithms such as multi-window (MW-SAD) [10], SAD with border correction (SAD+BC) and SAD with border correction and error filtering (SAD+BC+EF) [59], SAD with systematic error reduction (SAD+SER) and SAD with systematic error reduction and error filtering (SAD+SER+EF) [45]. These algorithms use different parameters and different number of windows of different sizes; so to make a fair comparison between them, for each of these algorithms, the results are computed for the best parameter combinations (i.e. which gives the lowest error) reported by their corresponding authors. It is shown that the proposed algorithm yielded an error of 3.08% which is superior to the other fixed size window-based correlation algorithms.

Figure 2.10 demonstrates the performance of the proposed algorithm with different window sizes. While the small window size W_s is fixed to 3×3 , we change the size of the first window (described as large window W_l in the proposed algorithm) used during initial matching operation. The window size W_l has been

2. Pixel-based Registration in Stereo Images

Table 2.1: Comparison of the proposed method with other real-time algorithms listed on Middlebury evaluation table for absolute disparity error > 1 .

Algorithm	Tsukuba			Venus			Teddy			Cones			%
	<i>nocc</i>	<i>all</i>	<i>disc</i>	<i>nocc</i>	<i>all</i>	<i>disc</i>	<i>nocc</i>	<i>all</i>	<i>disc</i>	<i>nocc</i>	<i>all</i>	<i>disc</i>	Error
RTBFV[87]	1.71	2.22	6.74	0.55	0.87	2.88	9.90	15.0	19.5	6.66	12.3	13.4	7.65
RTABW[69]	1.26	1.67	6.83	0.33	0.65	3.56	10.7	18.3	23.3	4.81	12.6	10.7	7.90
<i>Results-1</i>	<i>2.25</i>	<i>3.08</i>	<i>11.6</i>	<i>0.92</i>	<i>1.31</i>	<i>7.53</i>	<i>10.7</i>	<i>15.7</i>	<i>23.6</i>	<i>8.25</i>	<i>13.5</i>	<i>16.6</i>	<i>9.59</i>
RTCensus[78]	5.08	6.25	19.2	1.58	2.42	14.2	7.96	13.8	20.3	4.10	9.54	12.2	9.73
RTGPU[88]	2.05	4.22	10.6	1.92	2.98	20.3	7.23	14.4	17.6	6.41	13.7	16.5	9.82
DCBGrid[89]	5.90	7.26	21.0	1.35	1.91	11.2	10.5	17.2	22.2	5.34	11.9	14.9	10.9
<i>Results-2</i>	<i>2.80</i>	<i>4.20</i>	<i>13.4</i>	<i>1.64</i>	<i>2.44</i>	<i>14.8</i>	<i>11.0</i>	<i>16.4</i>	<i>24.9</i>	<i>8.89</i>	<i>14.6</i>	<i>17.7</i>	<i>11.1</i>
SAD+IGMCT[90]	5.81	7.14	22.6	2.61	3.33	25.3	9.79	15.5	25.7	5.08	11.5	15.0	12.5
SSD+MF[42]	5.23	7.07	24.1	3.74	5.16	11.9	16.5	24.8	32.9	10.9	19.8	26.3	15.7

Table 2.2: Comparison of the algorithm speed (MDE/s) and the implementation platform.

Algorithm	Speed (MDE/s)	Implementation
RTBFV[87]	100.9	GPU
RTABW[69]	7.8	CPU
<i>Our method</i>	<i>10.1</i>	<i>CPU</i>
RTCensus[78]	45	FPGA
RTGPU[88]	42.1	GPU
DCBGrid[89]	135	GPU
SAD+IGMCT[90]	1080	FPGA
SSD+MF[42]	20	CPU

2.4 Experimental Results and Analysis

Algorithm	Tsukuba			Venus			Teddy			Cones			% Error
	<i>nocc</i>	<i>all</i>	<i>disc</i>	<i>nocc</i>	<i>all</i>	<i>disc</i>	<i>nocc</i>	<i>all</i>	<i>disc</i>	<i>nocc</i>	<i>all</i>	<i>disc</i>	
With window W_s	2.25	3.08	11.6	0.92	1.31	7.53	10.7	15.7	23.6	8.25	13.5	16.6	9.59
Without window W_s	3.22	4.25	17.1	1.61	2.16	15.2	11.3	16.3	27.5	8.42	13.9	19.6	11.7

Table 2.3: Quantitative results obtained with and without using small correlation window W_s by the proposed algorithm.

changed from 7×7 to 21×21 pixels. The graphs show the percentage error in non-occluded, whole image and near depth discontinuities for Tsukuba, Venus, Teddy and Cones images on using different window sizes are shown. The error graphs show that unlike other window-based approaches, the proposed algorithm is not very sensitive to matching window size and the change in the size of matching window does not affect the performance of the algorithm significantly. On increasing the size of matching window, the error increases linearly for *Tsukuba*, *Teddy* and *Cones* images while it becomes constant for *Venus* image due to less amount of depth discontinuities as compared to other images.

Table 2.3 demonstrates the quantitative results obtained with and without using small correlation window W_s by the proposed algorithm. It can be seen that the use of small correlation window W_s improves the accuracy significantly.

2.4.2 Qualitative Results

Figure 2.11 shows the results of the proposed algorithm (a) without using small correlation window W_s and the disparity refinement step, (b) without using the disparity refinement step, (c) without using small correlation window W_s and (d) with all four steps on the Tsukuba image. The results show that the use of small correlation window W_s and disparity refinement step improves the accuracy at object boundaries (see Figure 2.11(d)).

Figure 2.12 shows the qualitative results for standard Middlebury dataset. The first and second rows show the left images and the corresponding ground truth images, respectively. Third row shows the disparity maps computed by using the proposed algorithm. The fourth and fifth rows show the error maps

2. Pixel-based Registration in Stereo Images

Figure 2.10: Error for different window size - Shows the percentage error in non-occluded (*nocc*), whole image (*all*) and near depth discontinuities (*disc*) for different window sizes W_l (7×7 to 21×21) for all four images (Tsukuba, Venus, Teddy and Cones).

Figure 2.11: Intermediate results - Shows the results of the proposed algorithm for (a) without using small correlation window W_s and the disparity refinement step (b) without using the disparity refinement step (c) without using small correlation window W_s and (d) with all four steps on the Tsukuba image.

2. Pixel-based Registration in Stereo Images

and signed disparity errors for all four images, respectively. These error maps have been calculated by using Middlebury evaluation system for the error threshold value set as one. The black pixels in the error maps represent the error in non-occluded image regions while the gray pixels represent the errors in occluded regions. The white image area represents the correct matches.

Figure 2.13 shows frames from a video that have been extracted from the stereo vision system installed in our lab. The first and second rows show the left and the right images respectively while the third row shows the corresponding disparity maps obtained using the proposed algorithm. The disparity range of these images is around 60 pixels. The Middlebury datasets contain images that have been taken in ideal lighting environment, but the conditions in real environment may be quite different. The disparity maps in Figure 2.13 show that the proposed algorithm works well even in environments with low illumination.

Figure 2.15 and Figure 2.16 further show the qualitative results for other Middlebury dataset images. These test images are taken from both Middlebury 2005 and 2006 datasets. These datasets consist of variety of image features i.e. complex geometry (*dolls*, *Moebius* and *Reindeer*), repetitive patterns (*Aloe* and *Cloth* images) and non-textured image regions (*Wood1*, *Books* and *Medi1*). The images in these datasets also have large disparity ranges (more than 60 pixels), resulting in large occlusions. Due to large occlusions, repetitive patterns and higher percentage of untextured surfaces, the new Middlebury datasets are much more challenging as compared to the standard stereo benchmark dataset which contains images such as *Teddy* and *Cones*. The disparity maps shown in these figures have some streaking artifacts to the left side of the image. These image regions are occluded and the pixels in these image regions have no corresponding match in the right image as shown in Figure 2.14. We have propagated the disparity in such image regions from their right neighboring pixels.

Figure 2.17 and Figure 2.18 show the 3D reconstruction of a few images (*Tsukuba*, *Teddy*, *Cone*, *Rock2*, *Dolls*, *Moebius*, *Baby1*, *Baby2*, *Baby3*, *Cloth1*, *Cloth2*, and *Cloth3*) that have been taken from new Middlebury datasets, by using the disparity maps computed using the proposed algorithm. The experimental results clearly show that the proposed algorithm works very well in case of repetitive patterns, object boundaries, as well as in occluded and non-textured

Figure 2.12: Results on the Middlebury 2003 dataset - Results on the Middlebury data set (Tsukuba, Venus, Teddy and Cones). The first row shows the left images, the second row shows the ground truth disparity map, the third row shows the results obtained by using our algorithm and the fourth and fifth rows show the bad pixel maps and signed disparity error for error threshold value 1.0, respectively.

2. Pixel-based Registration in Stereo Images

Figure 2.13: Results on our dataset - Shows few images frames extracted from stereo vision system running in our lab. The first and second rows show the left and the right images while the third row shows the corresponding disparity map.

Figure 2.14: Streaking artifacts - Streaking artifacts occurred due to the occluded image regions.

image regions. These experimental results are obtained by taking same parameters (window size and constant value T) for all the images.

2.5 Conclusion

In this chapter, we present a fast and computationally efficient algorithm that computes dense disparity maps by registering pairs of stereo images. The algorithm uses two correlation windows (one large and one small size) to compute the dense disparity map. While the large correlation window gives good results at non-textured image regions, the small window improves the performance at depth discontinuities. Unlike the previous algorithms which evaluate multiple matching windows at each image pixels, here we have shown that the use of two matching windows is enough to achieve accurate disparity estimations. Rather than evaluating the second correlation window at each image pixel, we evaluate it near the depth discontinuities for a selected disparity values only. By restricting the search space, we are able to achieve good performance without increasing the algorithm's computation time significantly. The algorithm uses simple mathematical operations and can be easily implemented on GPU. An unoptimized CPU implementation of the proposed algorithm computes dense disparity maps at a speed of more than 10 frames per seconds in the case of 320×240 sized image pair with disparity value 16. The computation time can be significantly reduced by hardware specific optimization or GPU implementation of the algo-

2. Pixel-based Registration in Stereo Images

Figure 2.15: Results on the Middlebury 2005 & 2006 dataset - Results on the new Middlebury dataset. The first row in each column represents the left image, second row shows the ground truth image and the third row shows our results.

Figure 2.16: Results on the Middlebury 2005 & 2006 dataset - *cont'd* - Results on the new Middlebury dataset. The first row in each column represents the left image, second row shows the ground truth image and the third row shows our results.

2. Pixel-based Registration in Stereo Images

(a) Tsukuba

(b) Teddy

(c) Cone

(d) Rock2

(e) Dolls

(f) Moebius

Figure 2.17: 3D Reconstruction - Shows the 3D reconstruction of the left images by using the disparity maps computed by using the proposed algorithm.

Figure 2.18: 3D Reconstruction - *cont'd* - Shows the 3D reconstruction of the left images by using the disparity maps computed by using the proposed algorithm.

2. Pixel-based Registration in Stereo Images

rithm. The proposed method can be used in real-time applications to reconstruct the three-dimensional structures with high accuracy at object boundaries.

Chapter 3

Human Activity Detection using Depth Images

In this chapter, we present an application of stereo image registration where we use the three-dimensional scene information (a sequence of depth maps) to identify the human activities. The proposed algorithm does not require any markers attached to the human subject and recognizes various human activities by using the three-dimensional information only. This chapter provides an overview of the related work that has been done in this area followed by the proposed algorithm to identify the human activities. We also present new algorithms for depth image segmentation and clustering. To evaluate the performance of the proposed algorithm, we show the experimental results on our own as well as on standard *human activity detection* dataset.

3.1 Introduction

Robust classification of human activities find uses in a wide array of applications like video-based surveillance, action/event centric video retrieval and patient monitoring systems, and has attracted strong interest from the research community. Majority of these activity recognition methods focus mainly on gray or RGB video sequences, and exploit spatio-temporal interest points or silhouettes as classification cues. Here, a significant challenge is to sieve out features from the video sequence that are representative of the activity, in which cluttered

3. Human Activity Detection using Depth Images

Figure 3.1: Silhouettes ambiguities in poses - (a) shows the silhouettes ambiguity where different poses having different hand positions generate similar silhouettes. This ambiguity can be resolved using the three-dimensional shape information as shown in (b).

or moving backgrounds, illumination changes and camera motions easily corrupt these features. Additionally, given that these features incorporate only spatial x - y and temporal information, they often fail to fully resolve the silhouette ambiguity in poses under self occlusion [91].

There is an emerging consensus that depth information is a more reliable recognition cue for classifying human activities because of two main reasons: First, it is color/texture/intensity invariant and hence is robust towards appearance variations of the humans performing the action. Second, unlike gray/color images acquired with traditional 2D cameras which provides spatial x - y image information, depth images affords another important spatial dimension (z -component) which can be exploited to help resolve silhouettes ambiguities in poses as shown in Figure 3.1.

In this chapter, we present a method which harnesses only depth images to recognize very complex human activities such as brushing teeth, cooking-chopping and cooking-stirring. Our method does not demand knowledge of the body joint locations, but only requires a segmentation mask of the human silhouette to be available. We propose an algorithm to extract the human silhouette from depth

images. A scale and depth invariant descriptor has been developed to represent the 3D pose of these segmented silhouette. We extract the descriptors from training images depicting a wide array of poses, and cluster these descriptors to find a set of compact clusters. Here, we propose a spatial based clustering step to ensure cluster members are well represented by its medoids, and consider each medoid as a codeword. Collectively, these codewords represent the range of unique body poses that are exhibited by the humans in the training images.

Given a training depth video sequence (and its activity label), we segment the human from the video sequences, and represent the sequence of segmented bodies as a sequence of codewords. Discriminative subsequence of codewords that model unique poses and the distinguishing temporal ordering of these poses are then learned with a boosting framework. Each discriminative subsequence has a variable number of codewords, and is termed in this chapter as *x-subsequence-codewords* or *xSC*. Both the *xSC*, its corresponding window size and weight are learned automatically, and represent the most discriminative subsequences for the activity class. At test time, we adopt a sliding window approach to align each *xSC* to overlapping windows of the test sequence. Each *xSC* acts as a classifier and we learn a boosted ensemble of *xSC* to give a confidence score for the activity label of the test sequence. We evaluated our method on the Human Activity Detection database [92, 93] which comprises twelve challenging human activities performed by four different humans, and obtain superior classification results as compared to existing methods. Additionally, we also applied our method on the task of activity detection (i.e. localize all instances of a targeted activity from a video sequence). Accurate localizations of targeted activity are demonstrated.

3.2 Related Work

There is a vast literature of research work on human activity recognition [94, 95, 96, 97]. Majority of these works focus on the use of gray/color videos, and the use of depth images for activity recognition has attracted interest only in recent years. Here, we first discuss the more representative recognition works which exploit gray/color videos, before discussing those that exploit depth images.

3. Human Activity Detection using Depth Images

Figure 3.2: Template-based approach - uses (a) motion energy image, and (b) motion history image to recognize the actions.

There are two main branches of human activity recognition works, those employing a holistic representation and those using a part-based representation. In an activity recognition using holistic representation, Bobick *et al.* [98] developed the motion energy and motion history images (MHI) (Figure 3.2) to encode short spans of motions efficiently. Hu moments [99] extracted from these images are then exploited as activity descriptors. Efros *et al.* [100] exploited optical flow measurements to compute spatio-temporal descriptors, and demonstrated the power of these descriptors in recognizing activities seen in low resolution videos. Ahad *et al.* [101] used these flow measurements to solve the issue of self-occlusion in a MHI approach. Ali and Shah [102] computed a number of kinematic features from the optical flow which include vorticity, divergence, symmetry and gradient tensor features. They used principal component analysis to determine dominant kinematic modes.

Yilmaz and Shah [103] tracked moving 2D contours to generate a 3D spatio-temporal volume, and extracted differential geometry features from the surfaces of the volumes to recognize various activities. Although their method achieved very good activity recognition accuracy, it demands robust tracking of the contours and fails when contours cannot be tracked (e.g. under self-occlusion). Parameswaran *et al.* [104] used 3D joint locations to model the geometry of an action, and represented an activity in terms of its static canonical poses and dynamic trajectories

in 2D invariance space. A limitation of this work is the need for accurate 3D joint locations.

Motivated by the success in the object classification domain, part based representations for activity recognition have gained strong traction in the research community. Most of these methods employ sparse interest point descriptors such as [105, 106, 107] to recognize human actions. For example, Schuldt *et al.* [108] represented each video sequence by a bag-of-word representation, in which they employed STIP features [105] as the underlying features. A support vector machine is then learned for classifying test video sequences. Very good results have been demonstrated on videos depicting simple actions like running and walking. Oikonomopoulos *et al.* [109] detected spatio-temporal features by extending the work on 2D salient point detection by Kadir and Brady [110] in the temporal direction. The main drawback of these methods is the relatively small number of stable interest points. This issue is addressed by Dollar *et al.* [106] by applying separate linear filters in the spatial and temporal dimensions to detect interest points that have local maxima value in both dimensions. A classifier is then learned in a similar way as [108] to recognize a test activity. Gilbert *et al.* [111] searched spatio-temporal corners and determined the relative spatial arrangement of all other corners in the frame which generates extremely large number of features. To find the discriminative features from these features, they used data mining techniques to discriminatively select those feature combinations that are informative of a class. Liu *et al.* [112] used PageRank algorithm to find discriminative features. The above methods use a histogram representation of features and ignore the temporal ordering between features. To preserve the temporal ordering, Nowozin *et al.* [107] improved on the histogram representation and proposed a sequential representation to preserve the temporal ordering between words. The PrefixSpan subsequence mining algorithm is exploited to find discriminative representation of an activity. Their method does not model the global geometry of human body joints when performing an activity, and instead considers them as a simple bag of features.

The above methods exploit appearance (color/texture) information as the key recognition cues and are often easily corrupted by variations in illumination and appearance of the human body. More importantly, such features encode the

3. Human Activity Detection using Depth Images

spatial (x-y) information, and cannot fully resolve silhouette ambiguity seen in self-occlusion [91]. To overcome these problems, recent works focus on the use of depth images to recognize human activities. Li *et al.* [113] proposed the first work that use only depth information to recognize the human activities. Here, they sampled a bag of 3D points from depth images, and project these points into a 2D space. An action graph is constructed from the training points to encode the actions to be recognized. Although their method achieves good recognition accuracy, the use of 2D projections of key poses can lead to sub-optimal feature representations. Sung *et al.* [92, 93] directly used the orientations and 3D joint locations to learn a model based on Maximum-Entropy Markov Model (MEMM) [114]. Their method demands accurate detection of joint locations, which limits the applicability of their method.

3.3 Depth image segmentation

The segmentation of moving objects from a depth image is an important problem. The standard approaches that have been developed for RGB images can also be used for depth images [115, 116, 117, 118, 119, 120]. However, these approaches often fail to extract those objects that are very close to the background (i.e. person lying on the ground/bed, objects very near to wall etc) and require additional information (i.e. color) for proper segmentation. Figure 3.3 shows such an example where the person is lying on the ground. Since the object is very near to the background, the object boundaries become ambiguous in the depth map as shown in Figure 3.3(b). This edge map has been obtained by applying Canny edge detector on depth image. Figure 3.3(c-d) show the segmentation results obtained after applying a popular background subtraction method [119] on both color and depth images, respectively. It can be seen that the algorithm works very well with color image (shown in Figure 3.3(c)) but fails to extract the object completely when applied on depth image (Figure 3.3(d)).

To address this issue, we propose a method that computes the spatiograms [121] of the depth image along with the projections of its surface normals in xy , yz and zx planes for both images (background image and object+background image) and

Figure 3.3: Problem with depth segmentation - (a) Depth image where the object is very near to the background. (b) edge map extracted from the depth image (a). It can be seen that the algorithm fails to detect the object boundaries. (c-d) background subtraction results obtained by applying [119] on both color and depth images, respectively. On using the depth image, the algorithm fails to extract the object completely. (e) segmentation mask obtained by using the proposed method which is able to extract the object completely (similar to (c)) with the help of depth image only.

3. Human Activity Detection using Depth Images

then compares these spatiograms to compute the similarity scores [122]. These similarity scores are used to segment the object as shown in Figure 3.3(e).

Figure 3.4: Surface normals - Visualization of surface normals computed on (a) background depth image and (b) the object+background depth image (shown in Figure 3.3(a)) from two different viewpoints for better visualization.

Figure 3.4 visualizes the surface normals computed for both depth images from two different viewpoints. Figure 3.4(a) shows the surface normals computed for the background image, while Figure 3.4(b) shows the surface normals computed on Figure 3.3(a).

3.3.1 Surface normals extraction

We briefly review the method to extract surface normals as described in [123]. Let the position vectors of points in an image are given as a function of (s, t) coordinates of the parametric space (image coordinates). If $\vec{X}_s$ and $\vec{X}_t$ are the partial derivatives of $\vec{X}$ with respect to s and t , then:

$$\vec{X} = \begin{pmatrix} x(s, t) \\ y(s, t) \\ z(s, t) \end{pmatrix}; \quad \vec{X}_s = \begin{pmatrix} x_s(s, t) \\ y_s(s, t) \\ z_s(s, t) \end{pmatrix}; \quad \vec{X}_t = \begin{pmatrix} x_t(s, t) \\ y_t(s, t) \\ z_t(s, t) \end{pmatrix}. \quad (3.1)$$

For each of the functions $x(s, t)$, $y(s, t)$, and $z(s, t)$ a local least squares polynomial function fit is evaluated. This polynomial function is then differentiated to compute the derivatives $\vec{X}_s$ and $\vec{X}_t$. Given a function $f(s, t)$, the derivatives are evaluated as [124]:

$$f_s = D_s * S * f \quad \text{and} \quad f_t = D_t * S * f \quad (3.2)$$

where S is the operator to smooth the data, $*$ the convolution operator, and D_s and D_t are given as:

$$[D_s] = \vec{C}_0 \vec{C}_1^T \quad \text{and} \quad [D_t] = \vec{C}_1 \vec{C}_0^T \quad (3.3)$$

where

$$\vec{C}_0 = \frac{1}{7}[1 \ 1 \ 1 \ 1 \ 1 \ 1 \ 1]^T \quad \text{and} \quad \vec{C}_1 = \frac{1}{28}[-3 \ -2 \ -1 \ 0 \ 1 \ 2 \ 3]^T. \quad (3.4)$$

We can compute the normals by computing the cross product of the partial derivatives $\vec{X}_s$ and $\vec{X}_t$ as:

$$\hat{n} = \frac{\vec{X}_s \times \vec{X}_t}{\|\vec{X}_s \times \vec{X}_t\|}. \quad (3.5)$$

Figure 3.4 visualizes the surface normals computed for both depth images (background and object+background images) from two different viewpoints. Figure 3.4(a) shows the surface normals computed for the background image, while Figure 3.4(b) shows the surface normals computed on Figure 3.3(a). For each of these surface normals, the projections onto the xy -plane, the yz -plane, and the zx -plane are used to compute the spatiograms.

3.3.2 Spatiogram computation and matching

Birchfield and Rangarajan [121] introduced spatiogram as a generalization of the popular concept of histograms by adding spatial layout information to the feature distribution information of a histogram. Hence, as compared to histograms, spatiograms are also able to capture higher-order spatial moments of each attribute bin. A second-order spatiogram model stores the histogram of local features along with the mean (μ) and covariance (Σ) of the spatial position of all pixels that fall into each histogram bin.

3. Human Activity Detection using Depth Images

To compute the histogram for an image of N pixels, the histogram bin count n_b of bin b can be written as:

$$n_b = C \sum_{i=1}^N \delta_{ib}, \quad (3.6)$$

where C is the normalizing constant and

$$\delta_{ib} = \begin{cases} 1 & \text{if } i^{th} \text{ pixel falls in } b^{th} \text{ bin} \\ 0 & \text{otherwise.} \end{cases} \quad (3.7)$$

For each bin, the spatial mean (μ_b) and covariance (Σ_b) can be computed as:

$$\mu_b = \frac{1}{\sum_{j=1}^N \delta_{jb}} \sum_{i=1}^N X_i \delta_{ib}, \quad (3.8)$$

$$\Sigma_b = \frac{1}{\sum_{j=1}^N \delta_{jb}} \sum_{i=1}^N (X_i - \mu_b)(X_i - \mu_b)^T \delta_{ib}, \quad (3.9)$$

where $X_i = [x, y]^T$ is the spatial position of pixel i . We scale the coordinates of an image into the $[-1, +1]$ range. The spatiogram $S = \{n, \mu, \Sigma\}$ is computed for the depth image and for all three projections of its surface normals in xy -plane, yz -plane and zx -plane.

After computing the spatiograms for both background image and object + background image, we compare the spatiograms of each plane to find the similarity between these images. We use the method proposed by Conaire *et al.* [122] to measure the similarity between two spatiograms at each image pixel. The similarity (ρ) between two spatiograms, $S = \{n, \mu, \Sigma\}$ and $S' = \{n', \mu', \Sigma'\}$, can be computed as:

$$\rho = \sum_{b=1}^B \sqrt{n_b n'_b} [8\pi |\Sigma_b \Sigma'_b|^{1/4} N(\mu_b; \mu'_b, 2(\Sigma_b + \Sigma'_b))], \quad (3.10)$$

where $N(X; \mu, \Sigma)$ represents a normalized Gaussian evaluated at X . The similarity maps ρ_d , ρ_{xy} , ρ_{yz} and ρ_{zx} are computed for the depth image and all three projections of its surface normals in xy , yz and zx planes as shown in Figure 3.5(a-d), respectively. The dissimilarities between two images (background image and

3.3 Depth image segmentation

object+background image) have been represented by the darkness of the pixels in different image regions. These similarity maps are then combined together to detect the object position as follows:

Figure 3.5: Spatiogram matching - Similarity scores computed after comparing the spatiograms of (a) the depth image (ρ_d) and (b-d) projections of its surface normals in xy -plane (ρ_{xy}), yz -plane (ρ_{yz}) and zx -plane (ρ_{zx}), respectively. (e) shows the average similarity score (ρ_{xyz}) of the surface normal projections. (f) segmentation mask calculated using the spatiograms (a) and (e).

$$mask = \begin{cases} 1 & \min(\rho_d, \rho_{xyz}) \leq th \\ 0 & \text{otherwise.} \end{cases} \quad (3.11)$$

where th is the threshold and the value of ρ_{xyz} is

$$\rho_{xyz} = \frac{1}{3}(\rho_{xy} + \rho_{yz} + \rho_{zx}) \quad (3.12)$$

3. Human Activity Detection using Depth Images

To generate the final segmentation mask, we perform some morphological operations to fill the holes and smoothen the object boundaries.

3.3.3 Segmentation results

In this section, we compare the performance of the proposed algorithm with a popular background subtraction method [119]. To compute the similarity score at pixel p , we compute the spatiograms using neighboring pixels of p in both images and then find the similarity between these two spatiograms using equation (3.10). In our experiments, we fix the neighborhood size and number of bins to 15×15 and 24, respectively. The threshold value th (in equation (3.11)) has been chosen empirically and was fixed to 0.86 in all our experiments.

Figure 3.6 shows the segmentation results obtained by using the proposed algorithm and [119]. Here, we use the depth images as an input for the proposed method while we compare with Zivkovic’s method [119] that is applied on both color and depth images, respectively. The input depth images are shown in Figure 3.6(a). Figure 3.6(b-c) show the segmentation masks obtained by applying [119] on color and depth images, respectively. The segmentation masks obtained by our method have been shown in Figure 3.6(d). It can be seen that the proposed method is able to extract the complete objects from the depth images that are very close to the background.

3.4 Codebook of human poses

We build a codebook of body poses from training images, and represent a human pose shown in a video frame by its most similar codeword. Here, explicit effort is made to learn codewords which have coherent appearances with a large number of poses (i.e. can represent most poses well), but have substantial different appearances from other codewords (i.e. effectively covers the range of poses shown in training set). To learn this codebook, we segment human body from the training depth images, and represent each segmented silhouette by a scale, depth and translation invariant descriptor. We apply a spatial-based clustering

Figure 3.6: Results of depth image segmentation - (a) input depth image, (b-c) segmentation mask obtained after applying standard background subtraction method [119] on color and depth images, respectively. (d) segmentation mask obtained by using our algorithm.

3. Human Activity Detection using Depth Images

method to these descriptors, and consider the medoid of each resulting clusters as a codeword.

3.4.1 Computing pose descriptors

We extract descriptors from training images depicting a wide array of body poses. To ensure a descriptor is translation and scale invariant, we segment the human from the background (which is easily achieved by using the method discussed in previous section), and resize the segmented body to a canonical size (we used a size of 280×280 pixels in our experiments) prior to computing the descriptor. To describe the segmented body, a straightforward option would be a histogram of normalized depth values computed over the body, which is a simple bag of features representation. However, we can empower the descriptor with greater discriminative potential by encoding the spatial layout of the normalized depth values. Here, we define a dense grid of size 28×28 over the bounding box of the segmented body (as shown in Figure 3.7), and compute the average normalized depth value for each cell in the grid. Additionally, we also compute the fraction of the body that is within each cell in the grid as shown in Figure 3.8. The former incorporates depth information into the descriptor, while the latter encodes the spatial layout of the pose. The concatenation of these two values across all cells yields the pose descriptor.

Figure 3.7: Pose descriptor - The first dimension of pose descriptor contains an average normalized depth value for each histogram bin.

Figure 3.8: Pose descriptor - The second dimension of pose descriptor contains the filling area by body parts as highlighted in green color for each histogram bin.

3.4.2 Spatial-based clustering to learn codewords

We cluster the set of descriptors that are extracted from the training set, and compute the medoid of each cluster as a codeword. Here, we seek codewords which are sufficiently flexible such that each codeword can well represent poses that have similar appearances, but yet are substantially distinct from each other to ensure they cover the possible variations of poses. One approach to clustering is to consider each descriptor in its entirety during the clustering step. However, such an approach considers every dimension in the descriptor *simultaneously*. Consequently, sharp differences between two descriptors in a few dimensions may be mitigated by small differences in other dimensions, so that dissimilar poses are grouped together into the same cluster.

To improve the quality of the clustering, we propose a spatial-based clustering step which explicitly incorporates pose information during clustering. Here, we divide the bounding box of the segmented body into four quadrants (see Figure 3.9), and cluster the descriptors based on their values within each quadrant separately. Specifically, we first apply clustering on the descriptor values corresponding to cells in the first quadrant to find sub-clusters whose members have similar poses in the first quadrant of the segmented body. The clustering step

3. Human Activity Detection using Depth Images

Figure 3.9: Image clustering - Spatial-based clustering step. We divide the bounding box of a segmented body into four quadrants as shown in (a). Clustering proceeds in the form of the hierarchical tree shown in (b), in which we apply clustering on descriptor values from the cells corresponding to a quadrant at each level of the tree. Notice that members in each sub-cluster have similar sub-poses (shown color coded in (b)) within the processed quadrants.

3.5 Discriminative subsequences of codewords

continues in the form of a hierarchical tree, in which each subsequent node of the tree corresponds to a sub-cluster formed from the previous clustering step, and we cluster on descriptor values from cells of quadrant i at level i of the tree. Consequently, each leaf node of the tree includes descriptors which have similar values in every quadrant, and hence are more likely to have similar poses. This improves on the clustering step. In our work, we apply the very powerful and robust mean-shift clustering to the set of descriptors. Given that we apply mean-shift clustering on separate dimensions of the descriptors (as opposed to the entire dimensions collectively), and each application of mean-shift at a node of the tree is evaluated on sub-clusters with fewer members than its parent node, clusters can thus be efficiently found.

We apply the above method on the Human Action dataset [92, 93], and plot a histogram of the Euclidean distances between the cluster medoids (i.e. codewords) and its members (intra-cluster distances) in Figure 3.10(a). For comparison, we show the histogram obtained by direct application of mean-shift clustering on the descriptor values in Figure 3.10(b). Note that while both possess the same number of clusters, the proposed method finds much more compact clusters as shown by the smaller mean value (a t-test shows that this smaller mean value to be significant, $p = 10^{-15}$). Figure 3.10(c) shows the histogram of the Euclidean distances between the cluster medoids found by the spatial-clustering method (inter-cluster distances), and Figure 3.10(d) shows those obtained with direct application of mean-shift clustering. The proposed method finds codewords that are substantially distinct from each other, as shown by the higher mean Euclidean distance between codewords ($p = 10^{-12}$). We show codewords selected by our method in Figure 3.11. As observed, salient and unique poses exhibited by the human, e.g. sitting, standing and drinking water are well represented.

3.5 Discriminative subsequences of codewords

In this section, we detail our method to learn discriminative subsequences of video codewords (termed xSC in this chapter) that are unique to a targeted activity. Inference at test time for the targeted activity is achieved by aligning these xSC to the test video codeword sequence with a sliding window approach, and computing

3. Human Activity Detection using Depth Images

Figure 3.10: Cluster analysis - Comparison of clustering quality. Histogram of the Euclidean distances between codewords and its members obtained by our method and by mean-shift are shown in (a) and (b) respectively. Histograms of the Euclidean distances between codewords obtained by our method and by mean-shift are shown in (c) and (d) respectively. Notice that the proposed method finds more compact clusters, while learning codewords which have more diverse appearances. T-tests show the improvements to be significant.

Figure 3.11: Cluster medoids - Poses corresponding to medoids (i.e. code-words) obtained for the Human Action dataset [92, 93].

3. Human Activity Detection using Depth Images

the weighted sum of the confidence score at each window. Here, we first detail the learning of these activity-specific subsequence of codewords (termed in the following as xSC) before discussing how they are exploited at test time.

We seek a discriminative xSC which models both the distinguishing poses of a human when performing an activity, as well as the distinctive temporal ordering of these poses to reliably recognize the activity. The former incorporates pose/appearance features as recognition cues, while the latter encodes important temporal information that are unique to the activity. The alignment of these discriminative xSC to a window of a test video sequence can thus be exploited to localize the activity within the test sequence with a sliding window approach. A key challenge here however is on the *alignment* of a xSC to the test window. Specifically, a same/different person may exhibit poses at test time that are slightly different from those seen during training, even when performing the same activity. Hence, the test video will be represented by a codeword sequence that is vastly different from the xSC learned during training. More importantly, a person may perform a same activity with different speeds during training and testing. Such temporal variations are typically inconsistent across the activity (e.g. faster actions in initial phase of the activity, and slower actions thereafter). This could make temporal ordering of the codewords in a xSC a potentially less reliable recognition cue. Here, we first describe our alignment method which addresses these two issues in a unified framework, before discussing how an ensemble of boosted xSC are learned.

3.5.1 Aligning xSC to a test window codewords

Motivated by recent successes of DNA sequences alignment in the bioinformatics field [125], we formulate the problem of video codewords alignment as a DNA sequence alignment problem where the sequence of codewords of an activity can be viewed as a *codon* in a DNA sequence. The mismatches that occur due to the variations in an activity performed by different persons, can be viewed as *point mutations* which can be handled by introducing *indels* i.e. insertion or deletion of mutations. Here, for a given discriminative xSC and a test window of codewords, we compute the optimum alignment between the two sequences by

3.5 Discriminative subsequences of codewords

the semi-global sequence alignment method [126]. Unlike the global alignment method, the semi-global alignment method ignores leading and trailing gaps in the sequence alignment process and is well suited for our problem in which a xSC may be longer than the test window codewords. Additionally, unlike local sequence alignment which finds *subsequences* of xSC within the test window, semi-global alignment searches for the presence of the *entire* xSC within the test window and hence fully exploit the discriminative potential of the xSC . In the following, we first describe the semi-global sequence alignment method, before addressing how this method gives robust alignment under pose and temporal variations in the video sequence.

Let P denote a window of codewords, $P = \{p_1, p_2, \dots, p_m\}$, where p_i is the i^{th} codeword in the sequence, and let $Q = \{q_1, q_2, \dots, q_n\}$ denote a discriminative xSC . The optimum semi-global alignment between P and Q can be obtained by first computing a $(m+1) \times (n+1)$ score matrix M , where $M(1, 1) = M(i, 1) = M(1, j) = 0$, $i = 1 \dots m$, $j = 1 \dots n$, and

$$M(i+1, j+1) = \min \begin{cases} M(i, j) + d(p_i, q_j) \\ M(i, j+1) + g \\ M(i+1, j) + g \end{cases} \quad (3.13)$$

Here, $d(p_i, q_j)$ denotes the Euclidean distance between the pose descriptors of codewords p_i and q_j , and g is a gap penalty which represents the cost of inserting a gap in the alignment. In all experiments, we set g to be equal to the standard deviation of the distances computed between every codeword. Given matrix M , we identify the entry $M(x, y)$ in the last row/column which has the minimum value as ϕ . It denotes the misalignment score between the two sequences. The optimum alignment is then found by keeping track of the elements that contribute the minimum distance at each step i.e. backward transversal of a path from $M(x, y)$ to $M(1, 1)$. As illustration, Figure 3.12(a) shows an example xSC that is learned for the open-bottle activity, and its alignment to a positive window, where the xSC and window codewords corresponds to different persons. Misalignment score ϕ obtained for the alignment is given in the same figure. Additionally, we also show an example of the alignment of the same xSC to a random negative window

3. Human Activity Detection using Depth Images

(drinking-water activity) in Figure 3.12(b). As observed, a higher misalignment score is obtained when xSC is paired to a negative window.

The formulation of the sequence alignment in the above share similarities to the classical time warping, but has an important difference in that time warping considers the distances of vector pairs taken from a common k -dimensional feature space, whereas sequence alignment considers the distances of codewords taken one each from each sequence [127]. Here, insertion and deletion of gaps in the alignment is similar to lengthening (slowing-down) and shortening (speeding-up) of the activity in the video sequence, and empowers the alignment procedure to be robust to temporal variations of an activity. Additionally, we compute the optimum alignment based on the pose differences between the codewords (rather than demanding the exact matching of codewords). This affords us much flexibility when aligning sequences with slight pose variations. In this aspect, we attempt to strike a winning trade-off: exploit the appearance and temporal orderings of the codewords encoded in a xSC to empower it with strong discriminative potential to recognize an activity, and leveraging on an alignment framework which affords an xSC with sufficient flexibility to align to test sequences with slight pose and temporal variations.

3.5.2 Learning discriminative xSC

We learn a boosted ensemble of xSC from positive and negative training sequences for each target activity. These xSC are aligned to test video codeword sequence with a sliding window approach at inference, where weighted sum of the confidence score at each window is used as the overall confidence measure for the targeted activity at the window. We first present our method to learn a single discriminative xSC , before discussing how a boosted ensemble of discriminative xSC is learned.

We require a discriminative xSC to align well to positive window codewords but not to the negative. In this work, we exploit validation video sequences for discriminative learning of xSC . Specifically, we select a random subsequence of codewords from a positive training codeword sequence as a candidate xSC , and align it to positive and negative validation codeword sequences with a sliding

(a)

(b)

Figure 3.12: Sequence matching - Alignment of example positive and negative window codewords on the same xSC that is learned for the open-container activity. Vertical lines between codewords denote the alignment of the codewords, while horizontal lines denote the gaps which are inserted to obtain the optimum alignment. Misalignment scores (i.e. ϕ computed by the semi-global alignment method) are shown for each alignment.

3. Human Activity Detection using Depth Images

Figure 3.13: Sequence alignment - Aligning a candidate xSC to a video codeword sequence. We show the window by the blue rectangle, and also depict the anchoring frame for the window. Candidate xSC assigns a misalignment score to each image frame of the video codeword sequence at the anchoring frame of the window.

3.5 Discriminative subsequences of codewords

window approach. Rather than assigning a fixed size window for every candidate xSC , we pair each candidate xSC with a window of random size, and anchor each window to a video frame in the validation sequence (see Figure 3.13). The anchoring frame gives the reference for the current window. The use of sliding windows enables our method to not only recognize human activities (i.e. classification task), but more importantly empower it with an ability to find a targeted activity within a video sequence (i.e. localization task). We consider the misalignment score (defined as ϕ above) for each window as a feature value of the window, where the class label for the window is +1 if its corresponding anchoring image frame is from a positive validation sequence, and -1 otherwise. The set of feature values and class labels, together with the weights of the windows (initialized according to the number of positive and negative validation video frames) are then used to learn a weak decision stump [128] which maximizes the weighted accuracy on the set of validation windows. The prediction of the decision stump classifier based on the learned xSC at a window w is,

$$xSC[w] = \begin{cases} +1 & \text{if } \phi < \tau \\ -1 & \text{otherwise} \end{cases}, \quad (3.14)$$

where τ is a learned parameter. We learn an ensemble of discriminative xSC by AdaBoost [129], where each boosting round outputs an xSC , its weight, window size, and a corresponding decision stump. All xSC learned by AdaBoost are then combined to form a boosted ensemble, where the overall confidence measure of an activity in a window w is

$$H[w] = \sum_{j=1} \alpha_j \times xSC_j[w], \quad (3.15)$$

with α_j as the weight of xSC_j and is learned during boosting. In this chapter, we pick candidate sXC by selecting random subsequences from positive training data, and assign its window size a random value between 75% to 125% of its length. We sample windows of the validation sequences (both positive and negative) at every frame, and apply 300 boosting rounds to learn the boosted ensemble.

3.6 Activities classification and localization

In this section, we describe how the learned boosted ensembles are used to recognize human activities (i.e. classification task), and how they are exploited to find human activities within a test video (i.e. localization task). The only image information used by the ensemble is the depth image information.

To classify the activity depicted in a test video, we learn a boosted ensemble for each activity, and capitalize on these boosted ensembles to recognize the activity-label of the test video. Specifically, we apply the boosted ensemble $H_i[\cdot]$ which was learned for the i^{th} activity to the test video in a sliding window approach. The sliding step is the same as that used for training. Each window in the test video is assigned a score based on eq. (3.15), and we consider the median score as the classification confidence of the test video as activity i . To localize the i^{th} activity, we apply the boosted ensemble $H_i[\cdot]$ on all windows of the test video, and employ the powerful mean shift mode estimation technique [130] on the confidence measure of each window, similar to [131]. Mean shift models the non parametric distribution with the kernel density estimator,

$$P(\mathbf{w}) \propto \sum_{w_j \in W} H_i[w_j] G\left(\frac{\mathbf{w} - w_i}{b_w}\right) \quad (3.16)$$

where the Gaussian kernel G uses bandwidth b_w . Mean shift efficiently find modes within the video sequence which are then used as the final set of detections. Here, we used the density estimated at each mode as the confidence value of the detection.

3.6.1 Classification results

We evaluate our method on stereo image dataset that has been recorded in our lab by using stereo camera. The dataset contains three activities (*clapping*, *waving* and *move forward*) performed by four persons (three males and one female). To compute the depth maps, we apply stereo matching algorithm proposed in Chapter 2. To extract the foreground object from these depth images, we use the method discussed in Section 3.3. Figure 3.14 shows few examples where each row shows the reference image, corresponding depth map, segmentation mask and

Figure 3.14: Stereo image dataset - Stereo image dataset recorded in our lab by using stereo camera. (a) Reference image, (b) disparity map computed using the algorithm proposed in Chapter 2, (c) segmentation mask computed using the method discussed in Section 3.3, and (d) segmented foreground object.

3. Human Activity Detection using Depth Images

the foreground object extracted using the segmentation mask, respectively. To evaluate the performance of the proposed method, we used leave-one-out cross validation to test each person data whereby the ensemble was trained on the data of three persons and tested on the fourth person. Figure 3.15(a) shows the confusion matrix of the activities listed in Figure 3.15(b). The proposed algorithm is always able to detect the activities *clapping* and *waving* while the activity *move forward* gets confused with *waving* half of the times. This misclassification is due to the hand movements in *move forward* activity which is not always in a straight line while performed by different persons and the precision of pixel-based registration performed on stereo images captured using Point Grey Stereo Camera [132]. Here, the proposed algorithm is able to achieve an accuracy of 83.3% to detect and classify these activities.

		Detected		
Ground truth		1	2	3
	1	1		
	2		1	
	3		0.5	0.5

(a)

ID	Activity
1	Clapping
2	Waving
3	Move Forward

(b)

Figure 3.15: Confusion matrix for stereo image dataset - Confusion matrix and the corresponding list of activities are shown in (a) and (b), respectively.

To evaluate the performance on publicly available dataset, we apply our technique on the challenging Human Activity dataset [92, 93] and compare against the best (to our knowledge) classification results obtained so far on the dataset. The only image information used by our method is the depth images and we adhere to the same evaluation protocols as those used by other methods. This dataset comprises 12 challenging activities performed by four persons (two males and two females). We list these activities in Figure 3.16(a), and show example depth images depicting each activity in Figure 3.16(c). We note that humans in

3.6 Activities classification and localization

ID	Activity
1	Brushing teeth
2	Rinsing mouth with water
3	Wearing contact lenses
4	Working on computer
5	Cooking-chopping
6	Cooking-stirring
7	Talking on the phone
8	Drinking water
9	Opening container
10	Talking on couch
11	Relaxing on couch
12	Writing on whiteboard

(a)

(b)

(c)

Figure 3.16: Activity dataset - Human Activity dataset [92, 93]. (a) List of activities. (b) Segmented human from (c) obtained by thresholding. IDs given in (b) and (c) indicates the activities listed in (a).

3. Human Activity Detection using Depth Images

the datasets are quite a distance away from the background. Moreover, since the background images are also not available with the dataset, we can not use our proposed method for segmentation. Hence, we apply a simple depth threshold value to segment the human from the background. In our work, we use the very efficient Otsu’s method [133] to learn an optimum depth threshold value to extract the foreground human. Figure 3.16(b) shows segmented humans automatically segmented from the corresponding images in Figure 3.16(c).

In Table 3.1, we compare the performance with *Naive Classifier (multi-class support vector machine)*, *One-level MEMM*, and *two-level MEMM*[93] with our method. Following [92, 93], we divide the activities into five different environments: office, kitchen, bedroom, bathroom, and living room. Based on the environments, three to four common activities are identified from given twelve unique activities (see Table 3.1). We test our method under two settings.

In the *new-person* setting, we evaluate the boosted ensemble learned for each activity on video sequences in which the ensemble has not previously seen the person carrying out the activity. We used leave-one-out cross validation to test each person data whereby the ensemble was trained on the data of three persons and tested on the fourth person. To learn an ensemble for an activity-class, we pick the positive sequence of one person (randomly chosen) as the training sequence, and use the sequences (both positive and negative) of the other two persons for validation. To evaluate the ensemble, we apply it on data of the person not used for training/validation, in which all activities of the person are used for testing. In this aspect, each activity is searched for in video sequences of *every* activity class. In the *person-seen-before* setting, we evaluate the boosted ensemble on video sequences for which the ensemble has previously seen the person carrying out the activity. Here, we report two-fold cross validation classification accuracy, in which we split the video sequence of each person into two equal halves, used one half for training/validation, and the other for testing.

We first report classification results for the *new-person* setting. Figure 3.17(a) reports the average cross-validation classification confidence of each activity computed across the four persons, and we report confusion matrix of the activities in Figure 3.17(b) irrespective to different environments. Turning to incorrect classifications, it can be seen that activities brushing-teeth and talking-phone are

3.6 Activities classification and localization

Figure 3.17: Classification scores and confusion matrix - Cross fold classification scores for each activity are shown in (a) and (c), and corresponding confusion matrix in (b) and (d) respectively. Results for *new-person* setting are shown in (a) and (b), and *person-seen-before* setting in (c) and (d). Indices in first row and column of each table corresponds to those in Figure 3.16 (a).

3. Human Activity Detection using Depth Images

often misclassified. This is due to the similar body poses for both activities (see Figure 3.16) which is further complicated by little hand motions in both activities. Consequently, our method which exploits the appearance and temporal cues fails to achieve accurate classification on these activities. On the other hand, our method attain high accuracies for activities which have unique poses (e.g. working-on-computer) and unique temporal pose ordering (e.g. rinsing-mouth). Here, we use the precision and recall measures to evaluate the performance of the proposed algorithm. Our algorithm is able to detect and classify the activities performed in different environments with an overall average precision/recall measure of 78.1%/75.4% which is better than the best result of 67.9%/55.5% currently obtained by Sung *et al.* [93]. It has to be mentioned that [93] exploits very accurate pose information obtained with robust tracking of body joints locations, while our method uses silhouettes that are (crudely) segmented from depth images. This further demonstrate the power of our learned ensemble to recognize complex activities.

Figure 3.17(c) reports the average cross-validation classification confidence for the *person-seen-before* setting, and Figure 3.17(d) reports the confusion matrix. It can be seen from both tables that the boosted ensemble has sharp improvement in performance. This is not surprising, since this experiment setting allows the boosted ensemble to directly exploit the unique characteristic of the person performing the activity to learn the xSC , but does highlight the capability of our method to hone in on these characteristics. Here, our algorithm is able to detect and classify the activities performed in different environments with an overall average precision/recall measure of 94.6%/93.1% which is better than 84.7%/83.2% obtained by Sung *et al.* [93] for the same setting. To our knowledge, this is the best result published so far for this dataset.

Location	Activity	new-person								person-seen-before							
		Naive Classifier		One-layer MEMM		Skel. + Skel HOG		Our Method		Naive Classifier		One-layer HEMM		Skel. + Skel HOG		Our Method	
		Prec	Rec	Prec	Rec	Prec	Rec	Prec	Rec	Prec	Rec	Prec	Rec	Prec	Rec	Prec	Rec
bathroom	rinsing mouth	77.7	49.3	71.8	63.2	51.1	51.4	100	100	73.3	49.7	70.7	53.1	61.4	70.9	100	100
	brushing teeth	64.5	20.5	83.3	57.7	88.5	55.3	100	75.0	81.5	65.1	81.5	75.6	96.7	77.1	100	100
	wearing contact lens	82.0	89.7	81.5	89.7	78.6	88.3	80.0	100	87.8	71.9	87.8	71.9	79.2	94.7	100	100
	Average	74.7	53.1	78.9	70.2	72.7	65.0	93.3	91.6	80.9	62.2	80.0	66.9	79.1	90.9	100	100
bedroom	talking on phone	82.2	32.6	82.0	32.6	63.2	48.3	60.0	75.0	70.2	67.2	70.2	69.0	88.7	90.8	100	62.5
	drinking water	19.2	12.1	19.1	12.1	70.0	71.7	50.0	50.0	64.1	31.6	64.1	39.6	83.3	81.7	80.0	100
	opening container	95.6	65.9	95.6	65.9	95.0	57.4	100	75.0	48.7	52.3	48.7	54.8	93.3	77.4	88.8	100
	Average	65.6	36.9	65.6	36.9	76.1	59.2	70.0	66.7	61.0	50.4	61.0	54.5	88.4	83.3	89.6	87.5
Kitchen	cooking (chopping)	33.3	56.9	33.2	57.4	45.6	43.3	60.0	75.0	78.9	28.9	78.9	29.0	70.3	85.7	100	100
	cooking (strring)	44.2	29.3	45.6	31.4	24.8	17.7	50.0	50.0	44.6	45.8	44.6	45.8	74.3	47.3	100	100
	drinking water	72.5	21.3	71.6	23.9	95.4	75.3	75.0	75.0	52.2	51.5	52.2	52.4	88.8	86.8	100	100
	opening container	76.9	6.20	75.8	6.20	91.9	55.2	100	75.0	17.9	62.4	17.9	62.4	91.0	77.4	100	100
	Average	56.8	28.4	56.6	29.7	64.4	47.9	71.2	68.7	48.4	47.2	48.4	47.4	81.1	74.3	100	100
living room	talking on phone	69.7	0.90	83.3	25.0	51.5	48.5	50.0	75.0	34.1	67.7	34.1	67.7	88.8	90.6	100	75.0
	drinking water	57.1	53.1	52.8	55.8	54.3	69.3	66.7	50.0	80.2	48.7	71.0	53.8	80.2	82.6	80.0	100
	talking on couch	71.5	35.4	57.4	91.3	73.2	43.7	100	50.0	91.4	50.7	91.4	50.7	98.8	94.7	100	62.5
	relaxing on couch	97.2	76.4	95.8	78.6	31.3	21.1	80.0	100	95.7	96.5	95.7	96.5	86.8	82.7	72.7	100
	Average	73.9	41.5	72.3	62.7	52.6	45.7	74.2	68.7	75.4	65.9	73.1	67.2	88.7	87.7	88.2	84.4
office	talking on phone	60.5	31.0	60.6	31.5	69.4	48.2	60.0	75.0	80.4	52.2	80.4	52.2	87.6	92.0	100	75.0
	writing on whiteboard	47.1	73.3	45.2	74.1	75.5	81.3	100	100	42.5	59.3	42.5	59.3	85.5	91.9	100	100
	drinking water	41.1	12.4	51.2	23.2	67.1	68.8	66.7	50.0	53.4	36.7	53.4	36.7	82.3	81.5	80.0	100
	working on computer	93.5	76.8	93.5	76.8	83.4	40.7	100	100	89.2	69.3	89.2	69.3	89.5	93.8	100	100
	Average	60.5	48.4	62.6	51.4	73.8	59.8	81.7	81.2	66.4	54.4	66.4	54.4	86.2	89.8	95.0	93.7
Overall Average		66.3	41.7	67.2	50.2	67.9	55.5	78.1	75.4	66.4	56.0	65.8	58.1	84.7	83.2	94.6	93.1

Table 3.1: Precision and Recall scores of Naive Classifier (multi-class support vector machine), One-level MEMM model, Two-level MEMM model, and Our method in each environment.

3. Human Activity Detection using Depth Images

3.6.2 Localization results

In this section, we demonstrate our method to localize a targeted activity from a test video sequence. Since the object is very near to the background (lying on the ground), we use the proposed method to extract the foreground. Here, we define the targeted activity as the kicking motion of the leg (which simulates epileptic seizure events), and depict such an activity in Figure 3.18. We test our method on a test sequence which shows a person performing the targeted activity at three separate times within the sequence. Specifically, this activity were performed at frames f_{175} to f_{280} , f_{676} to f_{847} and f_{1120} to f_{1325} . Frames f_1 to f_{174} , f_{281} to f_{675} , f_{848} to f_{1119} and f_{1326} to f_{1478} comprise random activities performed by the person.

Figure 3.18: Example training sequence - Example sequence of training images used to localize the targeted *leg-kicking* activity from a test video sequence.

We show key frames from each interval in Figure 3.19. We learn a boosted ensemble of xSC for the target activity, and plot the detection confidence for each sliding window in Figure 3.20(a). Detections obtained by mean shift mode estimation corresponds to the three peaks in the graph. We show these detections in Figure 3.20(b), where the fourth frame in each detection corresponds to the detected modes. As observed, these detections accurately localize the leg-kicking activity, and demonstrate the activity localization capability of our method.

3.7 Conclusion

We have proposed an application of stereo image registration where we exploit the three-dimensional scene information (depth images) alone for activity classification and localization. Towards this end, we proposed an algorithm to extract the human body from depth images. We used a descriptor to represent segmented human body, and exploited a spatial-based clustering method to find unique human poses (i.e. codewords). Given a training video, we represent the video as a

sequence of codewords and learn subsequences of codewords (termed xSC) that model both the discriminative poses of a human when performing an activity, as well as the distinctive temporal ordering of these poses. We identify such a discriminative xSC as one which aligns well only in positive test videos. Here, we formulate the problem of alignment of xSC with a test window codewords as the DNA alignment problem, and exploit the semi-global alignment method to find the optimum alignment. The insertion/deletion of gaps in the alignment represent expansion/compression of the videos, and affords the alignment to be robust towards temporal variations in the activity. At the same time, robustness to pose variations is achieved since the optimum alignment is computed from the pose differences of the codewords (rather than demanding the exact matching of codewords). Activity classification and localization results on test videos demonstrate the effectiveness of our approach, in which for the Human Activity dataset, we achieved improvement over the best results published so far for the dataset.

Figure 3.19: Key frames from test sequence - Key frames from different test sequence intervals belonging to target and random activities are shown here for illustration.

3. Human Activity Detection using Depth Images

(a)

(b)

Figure 3.20: Activity localization results - (a) Plot of detection confidence against frame number obtained by applying the ensemble of the test video using the sliding window approach. (b) Detections obtained by our method, where each row correspond to a detection and number indicates frame index. Red bounding boxes within segmented body correspond to detected modes with neighboring frames visualized alongside detected modes.

Chapter 4

Feature-based Registration for Similar Image Search

In this chapter, we present an image retrieval system that performs feature-based registration to retrieve most similar images from very large-scale image datasets (e.g. Internet images). The proposed approach leverages on a rich set of image features to find similar objects present in the scene and aligns them with the input image irrespective their position in the target image. A detailed comparison with other state-of-the-art algorithms has been made to demonstrate the effectiveness of the proposed system.

4.1 Introduction

The last decade has witnessed tremendous growth in computational power and storage capacity which has fueled the interest of many computer vision and computer graphics researchers in data-driven approaches [15, 35, 37, 134, 135, 136, 137]. However, the process of locating a desired image in a large and varied collection of images often becomes a source of considerable frustration. For example, recently Internet has emerged as a huge database for many data driven applications. A popular image sharing website *Flickr* is reported to host more than 6 billion of images [138] (See Figure 4.1). Another popular social networking website *Facebook* has database of more than 100 billion images (by summer 2011) and is increasing every month by 6 billion images [139]. Retrieval of any

4. Feature-based Registration for Similar Image Search

Figure 4.1: Images on Internet - Images available on a image sharing website Flickr.

meaningful information from such a huge and ever increasing database is, in itself a challenging task.

Traditionally, there are two frameworks that are used to retrieve similar images: text-based approaches and content-based approaches. Text-based approaches can be traced back to the early years of computer vision [140]. In text-based approaches, the images are manually annotated by text descriptors, which are then used by a database management system (DBMS) to perform image retrieval. While these approaches can be used for small image databases, it becomes impractical in the case of large image datasets due to the vast amount of labor required in manual image annotation. More importantly, annotation inaccuracy arises due to the subjectivity of human perception [141, 142]. To overcome these issues, content-based image retrieval was introduced in the early 1980s. In these approaches, instead of using manual annotations by text descriptors, images are indexed by their own visual content, such as color and texture [143, 144, 145, 146]. These content-based techniques are receiving increasing attention due to their great potential in many real-world applications, and hence become a part of some search engines such as *Google* and *Bing*. Apart from providing better image search experience to the user, similar image search has also been shown to be helpful for solving classical computer vision problems such as image categorization, image manipulation etc [135, 137].

Retrieval of similar images can also be viewed as an image registration problem where the database corresponds to a set of transformed versions of the query image [147]. The best match is expressed as a partial re-indexing of the database images in decreasing order based on their similarity to the query image. By registering different image features, the retrieval and indexing of the relevant images can be improved significantly.

In this chapter, we propose a content-based image retrieval system that uses very simple, yet surprisingly effective approach to find the images that contains similar objects and align them with any given input image. Here, we do not propose any new feature representations or image descriptors but use existing images features such as shape, intensity, texture distributions and dense SIFT features to represent the given input image. To find similar images from an image dataset, we register these image features with database images and rank these images based on their similarity to given input image. The improvement in the precision of retrieved images as a result of image registration is illustrated by comparing the results of our system with existing state-of-the-art algorithms and Google's *similar image search*.

The rest of the chapter is organized as follows: In Section 4.2, we briefly cover the related literature; Section 4.3 describes the proposed image retrieval system. Section 4.4 presents the experimental results and compares the performance of the proposed system with existing state-of-the-art algorithms. The last section concludes this chapter.

4.2 Related Work

There is a vast literature of research work on searching visually similar images from an image dataset [148, 149, 150, 151]. Image appearance and features (i.e. color, texture, SIFT etc) are the basis of these approaches. These image properties can be either extracted from the entire image or from regions. In general, most of the existing image retrieval methods can be roughly divided into two groups: *appearance-based* approaches and *local feature-based* approaches.

Appearance-based approaches find similarity between two images based on global visual features such as color, texture etc [144, 145, 152]. Color is one of

4. Feature-based Registration for Similar Image Search

the most widely used features in image retrieval due to its simplicity of implementation and insensitivity to image size and orientation [145, 148, 153]. Color histogram [154] is the most commonly used color representation to measure similarity between two images. Since most of the images produce very sparse color histograms, they are very sensitive to noise. To address this issue, Stricker and Orengo [155] used cumulative color histogram. To further improve the accuracy and reduce the histogram quantization effects [156], Shih and Chen [157] used first three color moments (i.e. mean, variance and skewness) for color feature representation. Although color features are simple to calculate and provide reasonable discriminating power, it gives many false positives when searching large image datasets. While texture is not as well-defined as color feature, it provides very important information to describe image contents. Motivated by psychological studies in human visual perception of texture, Tamura *et al.* [158] used visual texture properties such as coarseness, contrast, roughness etc. The performance was further improved by using the Gabor features and wavelet features for texture representation [159, 160]. Smith and Chang [159] used the mean and variance extracted from the wavelet subbands as the texture representation. Ma and Manjunath [160] used the Gabor wavelet transform for the texture image annotation and showed that the Gabor transform results are more close to human vision study results. Dorairaj and Namuduri [152] used both color and texture descriptor to enhance the image retrieval performance. Although not as widely used as color and texture features, shape features are also important image features which provide an efficient and robust role in image retrieval [161, 162, 163]. The earlier approaches suffered due to the geometric problems with shapes such as scale, rotation etc. To address these issues, Belongie *et al.* [164] proposed a new shape descriptor, referred to as *shape context* which is robust to a number of geometric transformations. Chen *et al.* [35] used *shape context* on salient image objects to retrieve the images that contain similar objects. Since the shape provides only the high-level details about the object, these approaches often lead to less relevant images during the similar image search.

Local feature-based approaches have attracted the attention of many researchers in recent years. The features such as *corner points* or *interest points*, traditionally used for stereo matching, are also being used for image retrieval. Tian *et al.* [165]

used wavelet-based *salient points* to retrieve the similar images. Along similar lines, Mikolajczyk and Schmid [166] used scale and affine invariant interest points that can deal with significant affine transformations and illumination changes. Grauman and Darrell [167] improved the matching of locally invariant features by using the earth mover’s distance [168]. The performance of these feature-based approaches significantly depends on the discriminative power of detected points that represent important image regions. Zhu *et al.* [169] proposed a novel Nonrigid Image Matching approach that recovers an explicit mapping between two images. While their approach is very effective in dealing with cases involving viewpoint changes and local deformations, it is not robust to large image changes. For effective representation of an image, Oliva and Torralba [170] proposed the GIST descriptor that uses a set of perceptual dimensions i.e. naturalness, openness, roughness, expansion and ruggedness to represent the spatial structure of an image. Li *et al.* [171] demonstrated the effectiveness of this descriptor to retrieve the same landmarks from a set of images. Hays and Efros [37] used the GIST descriptor to retrieve the similar images from web for scene completion. Sivic and Zisserman [172] proposed a bag-of-features (BOF) based approach that uses the SIFT descriptor [173] which are very discriminant and invariant to local transformations. Moreover, image comparison based on this descriptor is also robust to change in viewpoint, cropping and illumination. Recently, Liu *et al.* [174] used the dense SIFT feature that allows robust matching across different images and locates objects at different parts of the image. To find the correspondence between the images, they adopted the computational framework of optical flow and used SIFT descriptors for pixel-to-pixel matching which increases the computational cost significantly. Hoiem *et al.* [175] proposed a Bayesian approach for object-based image retrieval using relevance feedback. They compute the unconditional probability density of images and use it in a Bayesian framework to find close matches. Tieu and Viola [176] used boosting at query-time to learn query-specific discriminative features. However, these approaches require user guidance and multiple positive query images. Recently, the multi-modal approaches have been used which employ both metadata and image information to retrieve relevant images [177, 178, 179, 180].

4. Feature-based Registration for Similar Image Search

Figure 4.2: System pipeline - Block diagram of the proposed system for similar image search.

Here, we improve image retrieval results with the help of registration of query image with a preliminary set of retrieved images obtained through shape matching. It can be viewed as a form of feedback, which has been widely used in image retrieval literature [181, 182]. Feature-based registration improves performance of the system significantly and enables the system to find relevant images.

4.3 Proposed System for Image Retrieval

Motivated by the success of Chen *et al.*'s method [35] to retrieve the images that contain similar shape objects, here we extend their work to retrieve images where the objects also have similarities in their appearance to a given input image. The first two steps of our proposed system have been adopted from their method. Figure 4.2 shows the block diagram of the proposed system. It consists of the following four processing modules: (1) Automatic image segmentation; (2) Shape-based registration; (3) Feature-based registration; and (4) Aggregation of similarity metrics. As described in [35] salient objects are extracted and affine transformation matrix is computed by registering the shape of the object in the

database image to the shape of input object. This transformation matrix is then used to deform the database image object. We use deformed database object and compute the similarities between the input and database images for local image properties in feature-based registration step. In the last step, we combine the matching results and rank the database images based on their similarity scores.

Figure 4.3: Automatic image segmentation - Demonstrates some examples of the automatic image segmentation process. (a) input image, (b) salient image region, (c) expanded image region on which the Grabcut has been applied, and (d) segmented image region.

4.3.1 Automatic Image Segmentation

To extract objects from images, we use the method of [35] that involves saliency filtering followed by the image segmentation. We use saliency detection algorithm [183] to find the salient image region. The high-saliency region is then expanded by morphological dilation. The Grabcut algorithm [184] is applied on this expanded image region. Sometimes, the expanded image region does not contain the complete object in it. To address this problem, we iteratively apply morphological dilation to expand the image region followed by the Grabcut until

4. Feature-based Registration for Similar Image Search

the segmentation boundary does not change or a maximum number of iterations is reached. Here, we fix the maximum number of iterations to 20. Figure 4.3 shows some examples to demonstrate the automatic segmentation process. Figure 4.3(a) shows the input images. Figure 4.3(b) and (c) show salient image region and expanded region used by the Grabcut algorithm, respectively. The segmented objects are shown in the last column. Our implementation using C language performing under a 3GHz computer running Windows 7 operating system takes around 500 to 600 milliseconds per image.

4.3.2 Shape-based Registration

After segmenting objects from the images, we use the shape feature to find objects that have similar shapes as in the given input template. To find the objects that are similar in terms of shape, we use a popular shape matching algorithm proposed by Belongie *et al.* [164]. The algorithm defines a shape descriptor, the *shape context*, for a point p on the shape as a coarse histogram of the relative coordinates of the remaining shape points. The bins of the histogram are defined by the overlay of concentric shells around the centroid of the model and sectors emerging from the centroid. The process of finding the corresponding points between two shapes consists of a local matching stage and a global matching stage. In the local matching stage, for points p the best matching point q is found on the other shape. The cost of matching C_{pq} two points p and q is given as:

$$C_{pq} = \frac{1}{2} \sum_{k=1}^K \frac{[h_p(k) - h_q(k)]^2}{h_p(k) + h_q(k)}, \quad (4.1)$$

where $h_p(k)$ and $h_q(k)$ denote the K-bin normalized histogram at point p and q , respectively. In the global matching stage, correspondences between similar sample points on the two shapes are found.

To allow for some shape deformation, we apply an affine transformation to register the target object with template object, where the affine matrix is computed from the correspondences of matching shape context descriptors. Figure 4.4(a-b)

4.3 Proposed System for Image Retrieval

show the template and the target shapes, respectively. Figure 4.4(c) demonstrates the correspondence found between these two shapes by performing the shape context matching. Figure 4.4(d) shows the deformed template shape that has been registered to target object. We sum up the shape context matching cost with that of the affine registration to give an overall shape matching score. This overall score is used to rank the segmented objects, and the top N objects are retained for further filtering. In Figure 4.5, we show two examples where the images are displayed based on their shape consistency score.

Figure 4.4: Shape registration - Demonstrates the shape registration process. (a) and (b) show the template and the target shapes, respectively. (c) demonstrates the correspondence found between (a) and (b) by performing the shape context matching. (d) shows the deformed template shape that has been registered to target object.

Figure 4.5: Shape matching results - Demonstrates top five results obtained after the shape matching.

4.3.3 Feature-based Registration

While shape information can be a vital cue to filter away irrelevant objects, the inappropriate objects are likely to remain, e.g. all the top ranked objects in Figure 4.5 that have very similar shape to given template but have very different appearances as compare to the input images. In the following, we present a novel approach to improve image selection based on local information, which helps us to obtain images that are more similar to input template image. Before performing feature-based registration, we deform target images to align them with input reference image using transformation matrix computed above. This alignment improves the matching process and enables us to retrieve relevant images only.

Figure 4.6: Intensity histogram computation - The histogram bins are spanned by the row, column and intensity dimensions for intensity matching.

Intensity We use histogram to represent each database image object. Histograms have been widely used in content-based image retrieval. The main drawback of using histogram as the main feature is the loss of spatial information. To incorporate the spatial information, we use a 3D intensity histogram spanned by the dimensions of row, column and intensity as shown in Figure 4.6. The row and column dimensions of this representation encode the spatial distribution of the intensities. All database objects are registered to the input image by an affine transformation before building their histograms. While computing histogram, it

4.3 Proposed System for Image Retrieval

is important to avoid all boundary artifacts that may abruptly change histogram as a sample intensity shifts from one histogram bin to another. Therefore, we use trilinear interpolation to distribute the value of each sample into adjacent histogram bins. In other words, each contribution into a histogram bin is weighted by $1 - d$ for each dimension, where d is the distance of the pixel to the bin center measured in terms of bin spacing. We compute the distance between two histograms A, B (template and target objects) by the χ^2 distance as follows:

$$D_i(A, B) = \frac{1}{2} \sum_i \frac{(a_i - b_i)^2}{a_i + b_i}. \quad (4.2)$$

The normalization in χ^2 distance reduces the effect of large bins having undue influence. It is shown to be helpful in many cases, where the χ^2 histogram distance outperformed the L_2 norm [164, 185, 186]. In our implementation, we employ ten bins for the row and column dimensions and 64 bins for the intensity dimension.

Texture In addition to intensity, we also exploit texture features to evaluate object similarity. Here, we represent each object by its texton indices. Specifically, we collect a set of 100 random images that comprise a variety of image scenes (both indoor and outdoor) and do not correspond to any one particular object category. On each of these images, we apply Gabor filters with eight orientations varying in increments of $\pi/8$ from 0 to $7\pi/8$ and five exponential scales $\exp(i \times \pi), i = 0, 1, 2, 3, 4$ to compute a 40-dimensional texture feature at each pixel. Figure 4.7 visualizes such a feature representation of an image where we map the top three principal components of texture features to the RGB color space. k -means clustering algorithm is then applied to the set of texture feature vectors that are extracted from all pixels within the 100 random images to obtain a set of K clusters. The cluster centers are taken as texton codewords. In our experiment, we set the value of K as 64.

We convert the images into its texton representation as follows. The 40-dimensional texture feature is extracted from each pixel of an image and is assigned a texton codeword that is closest to it in the Euclidean sense. We assign a texton index to each image pixel. This texton representation of the image is

4. Feature-based Registration for Similar Image Search

Figure 4.7: Texture feature representation - Visualization of texture feature representation of an image.

then used to compute a histogram. To incorporate the spatial information in histogram, similar to intensity histogram computed above, we compute 3D texture histogram spanned by the dimensions of row, column and texon. The histogram is computed in similar manner as we compute for intensity. Here we also use the χ^2 distance to measure the distance between two histograms A and B as follows:

$$D_t(A, B) = \frac{1}{2} \sum_i \frac{(a_i - b_i)^2}{a_i + b_i}. \quad (4.3)$$

Dense SIFT While Gabor features provide an effective representation of texture in an image, it is not discriminative enough to represent all image regions. Here, we adopt dense SIFT features [173] that have strong discriminative ability, in a manner similar to that for texture. Here, we represent each object by its dense SIFT codeword indices. To build this codebook, we use the same set of images as we used to compute the textons. To compute the dense SIFT features at a pixel, we divide its neighborhood pixels into a 4 x 4 cell array. For each of these cells, we quantize the orientation obtained from the SIFT feature into 8 bins to obtain 4 x 4 x 8 = 128 dimensional feature descriptor. Figure 4.8 visualizes such a feature representation of an image where we map the top three principal components of dense SIFT features to the RGB color space. k -means clustering algorithm is then applied to the set of dense SIFT feature vectors that are extracted from all pixels within the 100 random images to obtain a set of K clusters.

4.3 Proposed System for Image Retrieval

The cluster centers are taken as dense SIFT codewords. In our experiment, we set the value of K as 64.

For an image, we first compute the dense SIFT features at each image pixel and then each of the image pixel is represented by the index of the its nearest dense SIFT codeword. The dense SIFT codeword index and the position of the pixel are used to construct a 3D histogram. Similar to the intensity and texton histograms, we use χ^2 distance to measure the distance between two histograms.

Figure 4.8: Dense SIFT feature representation - Visualization of dense SIFT feature representation of an image.

4.3.4 Combining Similarity Metrics

The intensity, texture and dense SIFT similarity scores are then linearly combined as

$$D(A, B) = \omega_i \times D_i(A, B) + \omega_t \times D_t(A, B) + \omega_s \times D_s(A, B) \quad (4.4)$$

where A, B denote an input object and a database object respectively. We denote $D_i(\cdot, \cdot), D_t(\cdot, \cdot), D_s(\cdot, \cdot)$ as the χ^2 distance between their intensity, texture and SIFT histograms. ω_i, ω_t and ω_s are set to 0.10, 0.45 and 0.45 in all our experiments.

4.4 Experimental Results

In this section, we present the experimental results obtained by using the proposed system. To evaluate our system, we build an image dataset by downloading thousands of Internet images using 15 keywords (bird, building, butterfly, castle, cheetah, deer, fish, fox, frog, giraffe, lion, lighthouse, parrot, rooster, zebra) from different image sharing websites [187, 188, 189]. We downloaded 30000 – 40000 of images for each of these keywords (total approx. 550K). To expand the diversity and quantity of downloaded images, we employ Google Translation to translate the keywords into German, Chinese, French, Spanish, Italian and Portuguese, then search with the translated terms as well. For a given input image, we extract the object (as discussed in Section 4.3.1) from the image region specified by the user. We use this segmented object to find images that contain the similar object and align these images with given input image. The database images do not require any user interaction and are processed using the method described in Section 4.3.1.

The proposed algorithm works on very large-scale dataset, so to keep the computational complexity low, here we first perform shape-based registration (requires approx. 200 milliseconds per image) and select a subset of images that have similar shape object as in given input image. The feature-based matching is then performed on these images to retrieve the images that are similar to a given input image. Table 4.1 compares the computation time required by our algorithm with other methods [35, 37, 169]. For this comparison, we use a small database of 1000 images and retrieve top 25 similar images for a given input image by using these methods. Note that we do not consider the time spent in automatic segmentation of database images as it is a one-time operation and the segmentation results can also be used by other query images without any processing overhead.

4.4.1 Qualitative Evaluation

Figure 4.9 shows some results obtained by using the proposed system. Here, we show top four similar images retrieved using our system. For each retrieved result, we also present the corresponding alignment with given input image in subsequent

Figure 4.9: Results of similar image search - Results obtained by using the proposed image retrieval system. For each object, the first row presents top four matches retrieved by using our system and the second row presents the corresponding alignment with given input image for each retrieved result.

4. Feature-based Registration for Similar Image Search

Figure 4.10: Comparison with other methods - Compares the performance of the proposed method with other state-of-the-art methods. Each row presents top four matches obtained by using corresponding methods.

4.4 Experimental Results

Figure 4.11: Comparison with other methods - *cont'd* - Compares the performance of the proposed method with other state-of-the-art methods. Each row presents top four matches obtained by using corresponding methods.

4. Feature-based Registration for Similar Image Search

Figure 4.12: Comparison with other methods - *cont'd* - Compares the performance of the proposed method with other state-of-the-art methods. Each row presents top four matches obtained by using corresponding methods.

row. This alignment has been computed using the corresponding affine transformation matrix obtained during the shape matching step (see Section 4.3.2). The registration of the shapes before matching different image features enables us to find the images that are very similar to the input image. It also makes our system invariant to rotation, scale and viewpoint changes. The results show that proposed method works very well for all such changes and only retrieves the relevant images from dataset.

Since, there are a large number of algorithms that have been proposed for content-based image retrieval, we restrict our performance evaluation to some popular algorithms [35, 37, 169] as well as to Google’s *similar image search* [187]. Figure 4.10, 4.11 and 4.12 compare the performance of our method with these approaches. For the fair comparison with these methods [35, 37, 169], we provide only salient image region to these approaches. To retrieve similar images from Google’s *similar image search*, we upload the input image on Google server [187] and select *Visually similar images* option. It can be seen that the proposed system outperforms these methods and retrieves the images which contain the objects that are perceptually more similar to the given input image in terms of their appearance (shape, colors and patterns).

4.4.2 Quantitative Evaluation

We are addressing here the problem of similar image search in large-scale image datasets (i.e. Internet images) that contain thousands or millions of images. Obtaining quantitative results on a large-scale database is practically impossible because the class labels for these images are very tedious to determine. Similar to Tieu and Viola [176], we perform quantitative evaluation on the top 100 ranked images retrieved by our system as well as other methods on a database of 8000 images. We use three query images from each of the five object classes *Building*, *Butterfly*, *Fox*, *Deer* and *Rooster* and hand-labeled the retrieved images as relevant or not for each class. Then we computed an average precision as the number of images returned increases from 1 to 100 as shown in Figure 4.13 where the x-axis represents the *number of images*, and the y-axis represents the *precision*. It enables us to measure how many of the images returned by these methods satisfy

4. Feature-based Registration for Similar Image Search

	GIST [37]	Sketch2Photo [35]	Our Method	NDK [169]
Time (in Sec.)	382	527	620	925

Table 4.1: Computation time required by different algorithms to retrieve similar images for a given input image from a small database of 1000 images.

Figure 4.13: Quantitative Comparison - Average precision for five object classes (*Building, Butterfly, Fox, Deer and Rooster*) over a database of 8000 images.

our query. Figure 4.13 shows that the proposed system retrieves more relevant images as compare to other methods [35, 37, 169, 187]. Since the measurement of appearance-based similarities to given input image is very subjective and different for each individual user, at the time of hand-labeling images, we restrict our self to object classes rather than their appearance similarities to query image i.e. how many retrieved images contain the same object as query image.

4.5 Limitations

While the shape-based registration step reduces the search space and improves the quality of retrieval results, it also restricts the applicability of the proposed algorithm to the images that have predominant objects in it. The images which do not have any salient object (e.g. landscape images), the shape-based registration cannot be performed and the performance of the proposed algorithm with feature-based matching steps remains comparable to GIST algorithm [37].

4.6 Conclusion

In this chapter, we proposed an image retrieval system that performs feature-based registration to retrieve most similar images from very large-scale image datasets (e.g. Internet images) and ranks these retrieved images based on their similarity to any given input image. The proposed approach leverages on a rich set of image features such as shape, intensity, texture and dense SIFT features to find the similar objects present in the scene and aligns them with the input image irrespective to their position in target image. We compared our approach with other state-of-the-art algorithms and Google’s *similar image search* to demonstrate the effectiveness of the proposed system.

Chapter 5

Image Colorization using Similar Images

In this chapter, we present an application that uses similar images obtained after performing feature-based image registration (discussed in the *previous chapter*) to solve an ill-posed problem which involves the colorization of input gray images. The proposed algorithm leverages the rich image content available on different image sharing websites to increase the visual appeal of the gray image by adding realistic color information. A detailed comparison with other state-of-the-art colorization methods and a user study which involved 30 subjects have been done to demonstrate the greater effectiveness of the proposed algorithm.

5.1 Introduction

The goal of image colorization is to add colors to a gray image such that the colorized image is perceptually meaningful and visually appealing (see Figure 5.1). A key challenge of this problem is that it is under-constrained since there are potentially many colors that can be assigned to the gray pixels of an input image (e.g. leaves may be colored in green, yellow and brown). Hence, there is no one correct solution to the colorization problem and human intervention often plays an important role in the colorization process.

In general, colorization methods can be broadly divided into two main classes: interactive colorization methods and automatic colorization methods. Interac-

Figure 5.1: Image colorization - Process of adding the color information to grayscale image.

tive colorization techniques require a user to manually mark color scribbles on the target image [190, 191, 192, 193, 194, 195, 196]. Colors from these scribbles are then smoothly propagated onto the entire image based on an optimization framework. A key weakness of such methods is that they demand extensive efforts from a user. Additionally, for such methods, colorization quality is strongly dependent on the user supplied color scribbles, and it is often difficult for a novice user to provide sufficiently good color scribbles to achieve desirable colorization results. Automatic colorization methods take a different approach to image colorization [134, 197, 198, 199, 200, 201, 202]. It is also one of the applications presented by Hertzmann *et al.* in [203] where they transferred the color information between the images instead of texture. Specifically, rather than obtaining chromatic values from the user, these methods take a reference color image as input, and transfer colors from the reference image to the target image. While these methods can reduce the extent of user efforts, in many cases, these methods often require careful tuning of a large number of parameter settings to yield satisfactory results. The colorization results also heavily depend on the selection of the reference image used for color transfer. The manual selection of such a suitable reference image which is very similar to input gray image is, in itself, a challenging task.

To colorize grayscale photos with less manual labor, we present an algorithm that takes advantage of the tremendous amount of image data available on the

5. Image Colorization using Similar Images

Internet. The Internet is almost certain to contain images suitable for colorizing a given grayscale input, but finding those images in a sea of photos is a challenging task, especially since search engines often return images with incompatible content. We address this problem with the help of the feature-based registration discussed in previous chapter to identify candidate reference images that are the most compatible with the grayscale target. The user needs only to provide semantic labels and segmentation cues for major foreground objects in the grayscale image, which is more intuitive than previous scribble-based user interaction used by previous colorization techniques. For each foreground object, a multitude of images is downloaded from the Internet using the semantic label as a search term, and our algorithm filters them down to a small number of best matches. To minimize the amount of user input, our algorithm does not require the user to label and segment background regions. Rather, it exploits correlations between the foregrounds and backgrounds of scenes by re-using the images downloaded for the foreground objects, which likely contain some backgrounds that can serve as a reference for background colorization.

We evaluate our method on a diverse range of images comprising of both deformable and rigid foreground objects. Experimental results demonstrate that our method, while straightforward, is sufficiently powerful to yield perceptually meaningful and visually appealing colorizations on these complex images. Additionally, comparison against existing state-of-the-art methods also demonstrates our method to be more effective at colorization. Our experiments and user study demonstrate the greater effectiveness of the proposed algorithm in comparison to previous techniques.

The chapter is structured as follows. Immediately below, we discuss related works. Section 5.3 describes the proposed colorization algorithm. We present colorization results on a wide array of images in Section 5.4. A detailed comparison with existing state-of-the-art colorization methods along with an user study are also given in the same section. Section 5.5 concludes this chapter.

Figure 5.2: Scribble-based colorization - An example of example-based colorization. (a) input gray image, (b) color scribbles provided by the user, and (c) the colorization result [190].

5.2 Related Work

Adding realistic colors to a gray image can improve the photorealism of the image, and has attracted much attention in the research community. Levin *et al.* [190] proposed a simple yet effective colorization algorithm that requires a user to provide color scribbles at various image regions as shown in Figure 5.2. These color scribbles are then propagated automatically to the entire image by a least-squares optimization method. Huang *et al.* [191] improved on this method to reduce color blending at image edges. Szeliski [204] introduced the locally adaptive hierarchical basis function technique to speed up the colorization process. Oscar *et al.* [205] proposed a colorization method based on Bayesian segmentation to improve the performance at image edges.

Yatziv and Sapiro [192] used multiple scribbles to colorize a pixel, where the combination weights are computed by a distance measure between the pixel and the scribble. Lagodzinski and Smolka [206] used the morphological distance transformation which considers the image structures to automatically propagate the color scribbles inserted into the image by the user. While these methods have been shown to achieve good colorization results, a main shortcoming of these previous approaches is that they require a large number of color scribbles on the gray image as input.

To reduce the number of color scribbles, Qu *et al.* [193] and Luan *et al.* [194]

5. Image Colorization using Similar Images

Figure 5.3: Example-based colorization - An example of example-based colorization. (a) input gray image, (b) example image used to transfer the color information, and (c) the colorization result [198].

propagated scribbles that are marked on an image patch to other patches that have similar texture features. While their methods reduce the number of required scribbles, like all other interactive colorization methods, they demand substantial artistic skills of the user to mark appropriate colors on image patches to yield desirable colorizations.

Rather than obtaining colors from a user, Welsh *et al.* [198] obtained colors from a user-supplied reference color image as shown in Figure 5.3. They extracted small image patches at each pixel of the target image and matched these patches to those of the reference image. These matches are then used to directly transfer colors from the reference image to the target image. Their method requires the user to manually mark corresponding regions between the reference and target to yield satisfactory color transfer, and is considerably less intuitive than our method which only needs a user-supplied reference image. Additionally, their method completely ignores spatial information of the pixels and hence their colorizations often yield very weak spatial consistency. Sýkora *et al.* [207] proposed an unsupervised image colorization method that combines image segmentation, patch-based sampling and probabilistic reasoning to colorize already designed black- and-white cartoons. This method performs patch-based sampling to predict the colors at different image points and then spreads these color to all over the image using probabilistic relaxation. This algorithm was mainly designed for frame-by-frame color transfers and greatly limited by the number of local structural matches between the example and target images. The algorithm completely fails when the example and target images are widely dissimilar.

To address the spatial coherency problem, Irony *et al.* [200] proposed a method which used a segmented reference color image as an additional input. Here, they exploit color and texture information from segmented regions of the reference image to automatically segment the target gray image into a set of locally homogeneous image patches. The color values from the segmented regions of the reference image are then transferred to the image patches of the target image as color scribbles. These color scribbles are propagated across the entire image by using Levin's method [190]. Charpiat *et al.* [201] assign colors to a gray image by minimizing an energy function using the graph cut approach. While their algorithm does not require any user intervention during the colorization process, colorization results by their method are heavily influenced by the choice of many parameters settings. Gupta *et al.* [208] proposed a learning-based approach that uses a set of decision trees to learn the relationship between different image features and their corresponding color values. In another work, Gupta *et al.* [209] extracted a set of image features at superpixel level from both input gray image as well as reference color image and then used a cascade feature matching scheme to find the suitable matches. These algorithms work quite well on a broad category of images; however, the colorization heavily depends on the user selection of the reference color images used to transfer/learn the color informations.

Liu *et al.* [134] proposed an example-based colorization technique which is robust to illumination variations in the target and reference images. Their method downloads multiple reference images from the Internet based on user-supplied keywords, and computes precise per-pixel registration between the reference and target images for color transfer. A key weakness of their method is that it is limited to colorizing foreground objects for which an exact replica of the target gray foreground object can be found. Consequently, their method is limited to rigid objects (e.g. landmarks) and is unsuitable for colorizing images which contain deformable objects such as those shown in this paper.

5.3 Algorithm

A block diagram of our system framework is shown in Fig. 5.4. To colorize a grayscale image, the user first segments the foreground object(s) with the powerful

5. Image Colorization using Similar Images

Figure 5.4: System pipeline - Foreground objects are labeled and segmented by the user. Our system searches the Internet to find relevant reference images for colorization, and utilizes these images to colorize the grayscale input image.

and intuitive Lazy Snapping technique Li *et al.* [210], and provides a semantic text label for each object. These semantic text labels are the keywords that describe the objects e.g. rooster, fish, butterfly etc. These are the only inputs required from the user. We use the semantic labels to download a large set of photos from image sharing websites such as Flickr, Google Image Search and Picasa. To expand the diversity and quantity of downloaded images, we employ Google Translation to translate the text labels into German, Chinese, French, Spanish, Italian and Portuguese, then search with the translated terms as well. Among the downloaded images, we find the most suitable reference photos (for both foreground objects and background) by filtering the search results with respect to similarity to the grayscale input. Each scene object and the background is then colorized using the reference images to obtain a diverse set of natural colorization results. The user can efficiently select from among these results with a provided user interface. Figure 5.5 shows a flowchart of the proposed algorithm to transfer the color information.

5.3.1 Internet image selection

To find the suitable foreground images, we use the method that has been discussed in Chapter 4. For background images, our system does not require labeling and segmentation of background regions, but instead capitalizes on the strong correlation that exists between the foreground and background components of a scene [211]. Reference images for the background regions in the input grayscale photo can generally be found among the images that were downloaded for the foreground objects, so we filter these images for suitable reference backgrounds as well. Based

Figure 5.5: Color transfer process - Outline for our method to transfer color information from colored objects to gray input object.

on the user input, the system automatically selects the top few matches for each foreground objects and background image to perform the colorization.

In comparison to labeled foreground objects, close matches for backgrounds are difficult to find, since they tend to exhibit much more diversity. Rather than employ the filtering method used for foreground objects, we utilize the compact GIST descriptor [170], which has been shown to be effective at finding semantically similar scenes such as forests and beaches. We exclude the foreground from the background filtering process by specifically adopting the weighted GIST descriptor [37], where the segmentation mask is used to weight each spatial bin of the GIST descriptor in proportion to the number of valid pixels in it. Filtering is thus performed using the sum of squared differences between the GIST descriptor of the input image and that of the Internet images, weighted by the segmentation masks of both the Internet and input images. Note that the background masks for the Internet images are simply the inverse of those computed for foregrounds during foreground image filtering.

5.3.2 Super-pixel representation

All Internet objects are first normalized to have the same diagonal bounding box length as the input object. We then apply over-segmentation [130] with stringent thresholds (of spatial bandwidth 1 and range bandwidth 3) to break these images into super-pixels. Typically, there are between 1000 to 5000 super-pixels per

5. Image Colorization using Similar Images

Figure 5.6: Image oversegmentation - Algorithm uses the super-pixel representation to transfer the color information.

image. Figure 5.6 shows such an image representation. We describe each super-pixel by (i) the average intensity, (ii) texture, (iii) SIFT feature values and (iv) spatial coordinates of its pixels. We also extract the median Cb-Cr color values from each super-pixel of a reference object.

5.3.3 Graph-based Optimization

We now describe our method to transfer colors from the Internet object to the gray input object. In assigning colors from reference images to the grayscale photo, we aim to optimize the quality and likelihood of correspondences while favoring smoothness/consistency between neighboring super-pixels. We compute a color c_a for each super-pixel P_a of the grayscale input image by minimizing the following energy function:

$$E = \sum_{P_a} E_d(c_a, P_a) + \lambda \sum_{P_b \in N(P_a)} E_s(c_a, c_b, P_a, P_b) \quad (5.1)$$

where $N(P_a)$ denotes the set of neighboring super-pixels of P_a , and λ is a weight parameter $\lambda \in \{0.1, 0.5\}$. The term $E_d(\cdot, \cdot)$ defines the data cost of assigning color c_a to P_a . To evaluate this cost, we first find the Internet object super-pixel of color $c = c_a$ with the most similar intensity to P_a . We then evaluate the intensity

Figure 5.7: Color transfer between images - Color information is transferred at super-pixel level by using an energy minimization framework.

based cost as

$$E_d^{intensity}(c, P_a) = \alpha_i \times \frac{1}{Pr_{intensity}(c, i)} + \beta_i ||i_a - i|| \quad (5.2)$$

where i_a and i are the intensity of P_a and this Internet object super-pixel respectively. The probability $Pr_{intensity}(c, i)$ of an Internet object pixel that has the intensity i and color c , can be computed as:

$$Pr_{intensity}(c, i) = \sum_{P_k \in \Upsilon} \frac{|P_k|}{A}. \quad (5.3)$$

Here, the cost will be smaller if many pixels of the Internet object have color c . The second term $||i_a - i||$ is the difference between their intensities, which favors Internet object super-pixels with an intensity similar to P_a . $|\cdot|$ denotes the number of pixels within a super-pixel, and Υ denotes the set of Internet object super-pixels whose intensity and color values are i and c respectively. The normalization factor A is the total number of pixels in the Internet object. α_i, β_i are normalization coefficients to ensure $1/Pr_{intensity}(c, i)$ and $||i_a - i||$ vary from 0 to 1. Without normalization, there could be one term which suppresses the other term (for example, $1/Pr$ may be much larger than the distance between

5. Image Colorization using Similar Images

intensity). Similarly, we also define

$$E_d^{texture}(c, P_a) = \alpha_t \times \frac{1}{Pr_{texture}(c, t)} + \beta_t \times ||t_a - t||, \quad (5.4)$$

$$E_d^{SIFT}(c, P_a) = \alpha_s \times \frac{1}{Pr_{SIFT}(c, s)} + \beta_s \times ||s_a - s||, \quad (5.5)$$

$$E_d^{spatial}(c, P_a) = ||x_a - x||. \quad (5.6)$$

Here, $Pr_{texture}(c, t)$ (or $Pr_{SIFT}(c, s)$) is the probability of a pixel having color c and texture t (or SIFT value s). The normalization coefficients $\alpha_t, \beta_t, \alpha_s, \beta_s$ are determined in the same way as α_i, β_i . We denote t_a, s_a, x_a as the average texture, SIFT value and spatial position of the super-pixel P_a respectively. Similarly, t, s, x are respectively the texture, SIFT value and position of the Internet object super-pixel of color c with the minimum Euclidean distance to t_a, s_a, x_a . The overall data cost is defined as

$$\begin{aligned} E_d(c, P_a) &= w_1 E_d^{intensity}(c, P_a) + w_2 E_d^{texture}(c, P_a) \\ &+ w_3 E_d^{SIFT}(c, P_a) + w_4 E_d^{spatial}(c, P_a) \end{aligned} \quad (5.7)$$

where $w_1, \dots, w_4$ are the combination weights. The term $E_s(\cdot, \cdot, \cdot, \cdot)$ measures smoothness between two super-pixels and is defined as:

$$E_s(c_a, c_b, P_a, P_b) = \mathcal{F}(a, b) \times ||c_a - c_b|| \quad (5.8)$$

where c_a and c_b are the colors assigned to P_a and P_b respectively, and $\mathcal{F}(a, b) = \exp^{-(w_1||i_a - i_b|| + w_2||t_a - t_b|| + w_3||s_a - s_b||)}$ controls the relative strength of the smoothing term. We use the belief propagation framework [212] to minimize the overall objective function and obtain the color assignments. Here, we would like to find the colors c_a to be assigned to the gray input super pixels P_a such that the above overall energy is minimized.

5.3.4 Weight Sampling and Result Selection

The performance of our method depends heavily on weights $w_1 \dots w_4$, which control the relative importance of different features, and on λ which controls the

extent of smoothing. In practice, we find that fixed weights cannot generate good results for all data. To address this problem, we colorize the object using many different weight settings, cluster the results, and let the user choose one via an intuitive interface.

Figure 5.8: Results with different weight settings - Foreground colorization results obtained by using different weight settings.

We sample each weight value w_i as 0, 10, 20 or 30 and λ as 0.1 or 0.5, which yields 512 different results for each scene object. Prior to clustering, we evaluate the colorfulness of these results using the method proposed by Hasler *et al.* [213]. They used statistical parameters i.e. mean, standard deviation to compute a

5. Image Colorization using Similar Images

Figure 5.9: Results with different weight settings - *cont'd* - Background colorization results obtained by using different weight settings.

colorfulness metric M as follows:

$$\begin{aligned} rg &= R - G \\ yb &= \frac{1}{2}(R + G) - B \\ M &= \sigma_{rgyb} + 0.3 \times \mu_{rgyb}, \end{aligned} \tag{5.9}$$

where σ_{rgyb} and μ_{rgyb} are the standard deviation and the mean value of the pixel cloud. These values are given as:

$$\sigma_{rgyb} = \sqrt{\sigma_{rg}^2 + \sigma_{yb}^2} \quad \text{and} \quad \mu_{rgyb} = \sqrt{\mu_{rg}^2 + \mu_{yb}^2}. \tag{5.10}$$

Table 5.1 summaries the *meaning* of the metric M . We discard images that have colorfulness metric below the *quite colorful*. The remaining images are grouped by k -means clustering (k is set to 5), where image distance is computed by the χ^2 distance of color histograms. From each cluster, we retain the two images that are closest to the Internet color reference image. Figure 5.8 displays examples of retained foreground object colorizations. For background colorization, we apply the same method except with $w_4 = 0$ because the geometric relationship between backgrounds is weak. Figure. 5.9 shows some colorized backgrounds. After colorizing the foreground objects and background, we apply guided image filtering [76] to suppress color noise while preserving color structures.

To compose a diverse set of colorized results, we typically use four to six reference Internet images for each input grayscale image. Colorized foreground objects and backgrounds are then combined to form a set of results. We have

Attribute	M
not colorful	0
slightly colorful	15
moderately colorful	33
averagely colorful	45
quite colorful	59
highly colorful	82
extremely colorful	109

Table 5.1: Correspondence between the colorfulness attribute and the colorfulness metric.

designed a user interface (see Figure 5.11) to help the user to quickly select a desirable result. We apply hierarchical k -means clustering ($k=6$) to the set of colorization results to generate a hierarchical tree of images, where images in the same cluster have similar color. We display six cluster centers to the user at a time. Clicking on a thumbnail brings the user one level down the hierarchical tree, and the six cluster centers at that level will then be shown. With this interface, a user can select a colored photo from among the $O(6^n)$ results in n clicks.

5.4 Experimental Results

We compare our colorization to [134] in Figure 5.12. The reference Internet examples used for our colorization are shown on the right. Our system produces a result similar in quality to that shown in Figure 1(c) of [134], but is capable of colorizing a substantially broader range of imagery. We note that [134] requires the reference Internet object to be identical to the target object for precise per-pixel registration between reference and target objects. Hence, it is limited to colorizing objects like landmark buildings.

In Figures 5.13 and 5.14, we compare our method with [198], [214], [201] and [209]. For all the methods, the foreground objects and background image were colorized separately using the same reference Internet examples obtained by our filtering. The threshold settings for [198, 201, 209, 214] were varied to obtain the best results for these methods. Our method is seen to work well for images

5. Image Colorization using Similar Images

Figure 5.10: Hierarchical result selection - Tree structure to help the user to quickly select the preferred image.

Figure 5.11: Graphical user interface - Our graphical user interface, based on a hierarchical cluster tree, helps the user to quickly select the preferred image.

Figure 5.12: Comparison with *intrinsic colorization* - Compares our method with *intrinsic colorization* [134]. (a) input gray image, (b-c) colorization results obtained with [134] and our method, respectively, and (d) the Internet images used to transfer the colors.

with different amounts of texture, as well as for very colorful objects such as the rooster and parrot. From our observations, the amount of time a user needs to spend on semantic labeling and providing segmentation cues for foreground objects, and selecting a final result from the user interface, is typically about 1-1.5 minutes in total for the images used in this work. The computation time of the proposed algorithm depends on the number of objects in an image that needs to be colorized. Assuming that the color reference images are available, an unoptimized Matlab implementation of the proposed algorithm takes around 15 minutes to colorize an object (contains around 2500 superpixels) on a 1.83 GHz computer running Window 7 operating system. While the superpixel extraction and feature computation steps require only few seconds (approx. 6 - 7 seconds), the majority of computation time is required by the energy minimization step. As after initial inputs from the user, the whole colorization process does not require any user intervention, it can be done offline and the colorization results can be selected by the user later.

Figures 5.16 and 5.17 exhibit several more colorization results from our method. For colorizing input images with multiple objects, we use a semantic label to download Internet images for each object, and utilize these images to colorize

5. Image Colorization using Similar Images

the object. The background is colorized using the collective sets of downloaded Internet images. Figure 5.15 displays different colorizations of an input image using different Internet examples. We show colorization results using different keywords (given at the top of each result) in Figure 5.19. Colorizations obtained with excessively broad keywords are shown on the left, and those obtained with more specific keywords are on the right. It is seen that colorizations are poor with excessively broad keywords. This is due to the lack of relevant Internet images that are downloaded. Taking the input parrot image in the top row of Figure 5.19 as an example, visual inspection yields no useful parrot images from the first 1000 Internet images downloaded by Flickr with the ‘animal’ keyword. Colorizations improved markedly when slightly more compatible keywords are used (*e.g.* using ‘bird’ as the keyword for the parrot image).

5.4.1 User Study

We performed a user study to quantitatively evaluate our method. We chose 120 images from the Internet and randomly converted 30 of them to grayscale (using Matlab’s *rgb2gray* function). The 30 images were then colorized by our method and the methods in [198, 201, 209, 214]. The other 90 images were used for comparison.

We show each subject a set of four different images at a time (as shown in Figure 5.18) for a total of 30 sets, and asked the subject to identify all artificially colorized photos in each set. Beforehand, the subject is told that at most two of the four images in each set are artificially colorized. In each set, the colorized images (if present) are obtained using the same method, to avoid bias against a comparatively weaker method. We perform two tests. In the first test, the subject is given five seconds to view the four displayed images. In the second test, the subject has unlimited time. In total, thirty subjects took part in this study.

Table 5.2 demonstrates the user study results for both the experiments. In the first experiment, the subjects classified results of our system as real 66.59% of the time. This compares favorably to the 48.90% obtained by [198], 40.79% by [214], 32.30% by [201] and 64.90% by [209]. Interestingly, they identified original

Figure 5.13: Comparison with other methods - Comparison with other state-of-the-art colorization methods. The first row shows the input foreground and background images that have been used by all colorization methods. Second row shows input gray image, rows (3-6) colorization results obtained by [198], [214], [201], and [209], respectively. The last row presents our colorization results.

5. Image Colorization using Similar Images

Figure 5.14: Comparison with other methods - *cont'd* - Comparison with other state-of-the-art colorization methods. The first row shows the input foreground and background images that have been used by all colorization methods. Second row shows input gray image, rows (3-6) colorization results obtained by [198], [214], [201], and [209], respectively. The last row presents our colorization results.

Figure 5.15: Colorization with different Internet examples - Demonstrates the colorization results obtained by using different Internet examples. The first row shows the input gray image. The second and third rows show different Internet examples used for foreground and background colorization, respectively. The last row presents the corresponding colorization results.

5. Image Colorization using Similar Images

Figure 5.16: Colorization results - Colorization results obtained by using the proposed algorithm. (a) input gray images, (b) the colorization results, and (c) the Internet images used to transfer the color information where the upper and lower half of the image have been used to colorize the foreground objects and background, respectively.

Figure 5.17: Colorization results - *cont'd* - Colorization results obtained by using the proposed algorithm. (a) input gray images, (b) the colorization results, and (c) the Internet images used to transfer the color information where the upper and lower half of the image have been used to colorize the foreground objects and background, respectively.

5. Image Colorization using Similar Images

Figure 5.18: Example images shown during user study - Examples of images displayed for quantitative evaluation. The third image is artificially colored with our colorization method, while the others are original photos.

Algorithm	Experiment 1	Experiment 2
Welsh <i>et al.</i> [198]	48.90%	45.23%
Tai <i>et al.</i> [214]	40.79%	39.55%
Charpiat <i>et al.</i> [201]	32.30%	27.36%
Gupta <i>et al.</i> [209]	64.90%	62.20%
<i>Our Algorithm</i>	66.59%	64.52%

Table 5.2: Compares the user study results obtained by using different colorization methods. The cell values represents the number of times the subjects classified the artificially colorized images as real color images.

color images as real only 90.57% of the time. We believe this is due to subjects scrutinizing these images to such an extent to believe they were artificially colored. In the second test, the subjects identified our results as real 64.52% of the time, while that by [198], [214], [201] and [209] drop to 45.23%, 39.55%, 27.36% and 62.20% respectively. The original color images were identified as such 90.33% of the time. These results demonstrate the better performance of our method, where even with unlimited viewing time, over half of the colorized images are of sufficient quality to appear real. A t-test shows the comparison results to be statistically significant, ($p < 10^{-6}$).

5.5 Conclusion

We proposed an application that uses similar images obtained after performing feature-based image registration to solve an ill-posed computer graphics problem

Figure 5.19: Colorization using different keywords - Colorization results using different keywords (shown at the top of each result). The colorizations obtained with excessively broad keywords are shown on the left, and those obtained with more specific keywords on the right.

5. Image Colorization using Similar Images

which involves colorization of grayscale images. Colorization of a grayscale photograph often requires considerable effort from the user, either by placing numerous color scribbles over the image to initialize a color propagation algorithm, or by looking for a suitable reference image from which color information can be transferred. Even with this user supplied data, colorized images may appear unnatural as a result of limited user skill or inaccurate transfer of colors. To address these problems, a colorization system that leverages the rich image content on the Internet has been proposed. As input, the user needs only to provide a semantic text label and segmentation cues for major foreground objects in the scene. With this information, the proposed algorithm downloads the images from different photo sharing websites and performs the feature-based registration to obtain suitable reference images that are reliable for color transfer to the given grayscale photo. Different image colorizations are generated from the various reference images, and a graphical user interface is provided to easily select the desired result. Our experiments and user study demonstrate the greater effectiveness of this system in comparison to previous state-of-the-art colorization techniques.

Chapter 6

Conclusions and Future Works

6.1 Conclusions

In this thesis, we have focused on pixel and feature-based images registration algorithms and explored their new application areas. This thesis can be divided into two parts. In the first part, we proposed a pixel-based registration method and its application to solve a computer vision problem. In the second part, we presented a feature-based registration method and its application to solve a computer graphics problem. In this section, we briefly summarize our work and contributions:

In Chapter 2, we proposed a fast and computationally efficient pixel-based image registration algorithm to register a pair of stereo images. The algorithm uses two different sized correlations windows i.e. one large and one small size window to compute dense disparity maps by finding an appropriate match for each reference image pixel in the target image. Unlike previous approaches that used multiple windows at each image pixel, the proposed algorithm evaluates the second correlation window near the depth discontinuities for the selected disparity values only. By restricting the search space, we achieved good performance without increasing the algorithm's computation time significantly. To demonstrate the effectiveness of our algorithm, we evaluated our algorithm on standard Middlebury datasets (2003, 2005 & 2006 dataset). An unoptimized CPU implementation of our algorithm computes dense disparity maps at a speed of more

6. Conclusions and Future Works

than 10 frames per seconds in case of 320×240 sized image pair with maximum disparity value 16.

Chapter 3 discussed an application of stereo image registration where we used the three-dimensional scene information (depth images) to classify and localize the human activities. Firstly, we proposed a novel method to extract the human body from depth images. Then we used a histogram-based descriptor to represent segmented human body, and exploited a spatial-based clustering method to find unique human poses (i.e. codewords). We used codewords to represent a training video as a sequence of codewords and then learned subsequences of codewords that model both the discriminative poses of a human when performing an activity, as well as the distinctive temporal ordering of these poses. We used such discriminative subsequences to classify and localize the human activities by aligning them with test videos. We formulated the problem of alignment of discriminative subsequence with a sequence of codewords representing test video as the DNA alignment problem, and exploited the semi-global alignment method to find the optimum alignment. The insertion/deletion of gaps in the alignment is done to address expansion/compression of the videos, and enable the alignment to be robust towards temporal variations in an activity while performed by different persons. At the same time, robustness to pose variations is also achieved since the optimum alignment is computed from the pose differences of the codewords (rather than demanding the exact matching of codewords). To demonstrate the effectiveness of our approach, we evaluated our algorithm on publicly available *Human Activity Detection* dataset where we achieved improvement over the best results published so far for the dataset.

In Chapter 4, we presented an image retrieval system that performs feature-based registration to retrieve most similar images from very large-scale image datasets and aligns and ranks these retrieved images based on their similarity to any given input image. The proposed approach leverages on a rich set of image features to find the similar objects present in the scene and aligns them with the input image irrespective to their position in target image. As the system deforms the database images before matching to align them with given input image, the proposed system is also scale and rotation invariant. We compared the qualitative and quantitative evaluations to compare the performance of the proposed system

with other state-of-the-art algorithms. The results show that the proposed system retrieves the images that are perceptually more close to given input image.

Chapter 5, presented an application that uses similar images obtained after performing feature-based image registration to solve an ill-posed computer graphics problem which involves colorization of grayscale images. Colorization of a grayscale photograph often requires considerable effort from the user, either by placing numerous color scribbles over the image to initialize a color propagation algorithm, or by looking for a suitable reference image from which color information can be transferred. Even with this user supplied data, colorized images may appear unnatural as a result of limited user skill or inaccurate transfer of colors. To address these problems, a colorization system that leverages the rich image content on the Internet has been proposed. As input, the user needs only to provide a semantic text label and segmentation cues for major foreground objects in the scene. With this information, the proposed algorithm downloads the images from different photo sharing websites and performs the feature-based registration to obtain suitable reference images that are reliable for color transfer to the given grayscale photo. Different image colorizations are generated from the various reference images, and a graphical user interface is provided to easily select the desired result. Our experiments and user study demonstrate the greater effectiveness of this system in comparison to previous state-of-the-art colorization techniques.

6.2 Future Works

Here we list few extensions that can further enhance the performance of the proposed applications and bring them more close to the real world:

RGBD information for Human Activity Detection: In our present work, we only use the depth information to identify the human activities while the color information has been completely ignored. However, the color and the depth images are rather complementary than exclusive. Hence, color can be used as an additional information along with the three-dimensional shape information to

6. Conclusions and Future Works

improve the performance of the proposed approach.

Human-Object Interaction: The object being used to perform an activity also provides an additional information that can be useful to recognize an activity. By automatically recognizing the objects and human-object interactions from the video, the performance of the human activity detection algorithms can be enhanced significantly.

Scene Understanding: As we have shown in our experiments, the performance of the proposed algorithm improves significantly if we divide the activities based on the environment in which they are being performed such as bathroom, bedroom, kitchen etc. An automatic understanding of such contextual information can further improve the accuracy and can help significantly in making more powerful human activity detection algorithms.

Grayscale Image Segmentation: The proposed colorization algorithm requires user to provide a segmentation cue for major foreground objects in the scene. It can be a major source of frustration to the end user if there are many such objects present in the scene. A solution of such problem can make the proposed algorithm more adaptable to the end users. The use of Internet images to guide the image segmentation process can enable us to limit the user interaction to provide the semantic text labels of major foreground objects only. A method to solve this classical computer vision problem will be desirable to many other computer vision and graphics applications as well.

Publications

- **Raj Kumar Gupta**, Chia Yong Sang Alex and Deepu Rajan, "Recognizing Human Activities using Depth Images", *21th Annual ACM International Conference on Multimedia (MM)*, 2013, Spain.
- **Raj Kumar Gupta** and Siu-Yeung Cho, "A Window-based approach for fast stereo correspondence", *IET Computer Vision* 2013.
- **Raj Kumar Gupta**, Chia Yong Sang Alex, Deepu Rajan, Ee Sin Ng and Huang Zhiyong, "Image Colorization using Similar Images", *20th Annual ACM International Conference on Multimedia (MM)*, 2012, Japan.
- **Raj Kumar Gupta**, Chia Yong Sang Alex, Deepu Rajan and Huang Zhiyong, "A learning-based approach for automatic image and video colorization", *Computer Graphics International*, Jun 2012, Bournemouth.
- **Raj Kumar Gupta** and Siu-Yeung Cho, "Stereo Correspondence using Efficient Hierarchical Belief Propagation", *Neural Computing and Applications*, Jan 2012.
- Chia Yong Sang Alex, Shaojie Zhuo, **Raj Kumar Gupta**, Yu-Wing Tai, Siu-Yeung Cho, Tan Ping and Stephen Lin, "Semantic Colorization with Internet Images", *SIGGRAPH Asia 2011 (Proc. ACM Transaction of Graphics, 2011, Vol. 30(6))*, Hong Kong.
- **Raj Kumar Gupta** and Siu-Yeung Cho, "A Color-based approach for disparity refinement", *11th International Conference on Control Automation Robotics & Vision (ICARCV)*, Dec 2010, Singapore.

6. Conclusions and Future Works

- **Raj Kumar Gupta** and Siu-Yeung Cho, "A correlation-based approach for real-time stereo matching", *6th International Symposium on Visual Computing (ISVC)*, Nov 2010, Las Vegas.
- **Raj Kumar Gupta** and Siu-Yeung Cho, "Real-time Stereo Matching using Adaptive Binary Window", *5th International Symposium on 3D Data Processing, Visualization and Transmission (3DPVT)*, May 2010, Paris.

References

- [1] T. Kanade and M. Okutomi, “A stereo matching algorithm with an adaptive window: Theory and experiments,” *IEEE Transactions Pattern Analysis and Machine Intelligence*, vol. 16, no. 9, pp. 1088–1095, 1994. 4, 13, 14, 15
- [2] O. Faugeras, B. Hotz, H. Mathieu, T. Viville, Z. Zhang, P. Fua, E. Thron, L. Moll, and G. Berry, “Real-time correlation-based stereo: algorithm, implementation and applications,” *INRIA Technical Report No. 2013*, 1993. 4, 13, 27
- [3] R. K. Gupta and S.-Y. Cho, “A correlation-based approach for real-time stereo matching,” *International Symposium on Visual Computing*, 2010. 4
- [4] K. Briechle and U. D. Hanebeck, “Template matching using fast normalized cross correlation,” *Proceedings of SPIE: Optical Pattern Recognition*, vol. 4387, pp. 95–102, 2001. 4
- [5] S.-D. Wei and S.-H. Lai, “Fast template matching based on normalized cross correlation with adaptive multilevel winner update,” *IEEE Transaction on Image Processing*, vol. 17, no. 11, pp. 2227–2235, 2008. 4
- [6] G. Egnal and ostas Daniilidis, “Image registration using mutual information,” *Technical Report, University of Pennsylvania*, 2000. 4
- [7] J. P. W. Pluim, J. B. A. Maintz, and M. A. Viergever, “Mutual information based registration of medical images: a survey,” *IEEE Transactions on Medical Imaging*, vol. 22, pp. 986–1004, 2003. 4

REFERENCES

- [8] B. Zitova and J. Flusser, “Image registration methods: a survey,” *Image and Vision Computing*, vol. 21, pp. 977–1000, 2003. 4, 6
- [9] A. Goshtasby, “2-d and 3-d image registration for medical, remote sensing, and industrial applications,” *John Wiley & Sons*, 2005. 4, 6
- [10] A. Fusiello, V. Roberto, and E. Trucco, “Efficient stereo with multiple windowing,” *IEEE Conference on Computer Vision and Pattern Recognition*, pp. 858–863, 1997. 6, 13, 14, 16, 29
- [11] R. Szeliski, “Image alignment and stitching: A tutorial,” *MSR-TR-2004-92, Microsoft Research*, 2005. 6
- [12] B. Rezaie and M. D. Srinath, “Algorithms for fast image registration,” *IEEE Transactions on Aerospace and Electronic Systems*, vol. 20, no. 6, pp. 716–728, 1984. 6
- [13] P. Hellier and C. Barillot, “Coupling dense and landmark-based approaches for nonrigid registration,” *IEEE Transactions on Medical Imaging*, vol. 22, pp. 217–227, 2003. 6
- [14] L. G. Brown, “A survey of image registration techniques,” *ACM Computing Surveys*, vol. 24, no. 4, pp. 325–376, 1992. 6, 12
- [15] N. Snavely, S. M. Seitz, and R. Szeliski, “Photo tourism: Exploring photo collections in 3d,” *ACM Transactions on Graphics*, vol. 25, no. 3, pp. 835–846, 2006. 6, 8, 79
- [16] V. Kolmogorov and R. Zabih, “Multi-camera scene reconstruction via graph cuts,” *European Conference on Computer Vision*, pp. 82–96, 2002. 6
- [17] Y. Taguchi, B. Wilburn, and C. L. Zitnick, “Stereo reconstruction with mixed pixels using adaptive over-segmentation,” *International Conference on Computer Vision and Pattern Recognition*, pp. 1–8, 2008. 6
- [18] D. Keysers, T. Deselaers, and T. M. Breue, “Optimal geometric matching for patch-based object detection,” *Electronic Letters on Computer Vision and Image Analysis*, vol. 6, no. 1, 2007. 6

REFERENCES

- [19] S. Belongie, J. Malik, and J. Puzicha, “Shape matching and object recognition using shape contexts,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 24, no. 24, pp. 509–522, 2002. 6, 7
- [20] A. C. Berg, T. L. Berg, and J. Malik, “Shape matching and object recognition using low distortion correspondences,” *IEEE Conference on Computer Vision and Pattern Recognition*, vol. 1, pp. 26–33, 2005. 6
- [21] F. Al-Omari, “Hand-written indian numerals recognition system using template matching approaches,” *Proceedings of the ACS/IEEE International Conference on Computer Systems and Applications*, 2001. 6
- [22] L. Dong, G. Yun-Jian, and Z. Xue-Yong, “On-line signature verification based on template matching approach and support vector data description,” *International Conference on Computer Application and System Modeling*, vol. 12, 2010. 6
- [23] S. Sassanapitak and P. Kaewtrakulpong, “An efficient translation-rotation template matching using pre-computed scores of rotated templates,” *International Conference on Electrical Engineering/Electronics, Computer, Telecommunications and Information Technology*, vol. 02, pp. 1040–1043, 2009. 6
- [24] F. Jurie and M. Dhome, “Real time robust template matching,” *British Machine Vision Conference*, 2002. 7
- [25] G. Hermosillo, C. Chefd’Hotel, and O. Faugeras, “Variational methods for multimodal image matching,” *International Journal of Computer Vision*, vol. 50, no. 3, pp. 329–343, 2002. 7
- [26] J. Du, S. Tang, T. Jiang, and Z. Lu, “Intensity-based robust similarity for multimodal image registration,” *International Journal of Computer Mathematics*, vol. 38, no. 1, pp. 49–57, 2006. 7
- [27] M. Scharfe, R. Pielot, and F. Schreiber, “Fast multi-core based multimodal registration of 2d cross-sections and 3d datasets,” *BMC Bioinformatics*, pp. 11–20, 2010. 7

REFERENCES

- [28] A. Gholipour, N. Kehtarnavaz, R. Briggs, M. Devous, and K. Gopinath, “Brain functional localization: A survey of image registration techniques,” *IEEE Transactions on Medical Imaging*, vol. 26, no. 4, pp. 427–451, 2007. 7
- [29] N. Ahmed, C. Theobalt, M. Magnor, and H.-P. Seidel, “Spatio-temporal registration techniques for relightable 3d video,” *International Conference on Image Processing*, vol. 2, pp. 501–504, 2007. 7
- [30] Y. Songa, X. Yuana, and H. Xua, “A multi-temporal image registration method based on edge matching and maximum likelihood estimation sample consensus,” *International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, pp. 61–66, 2008. 7
- [31] L. Cai, L. He, Y. Xu, Y. Zhao, and X. Yang, “Multi-object detection and tracking by stereo vision,” *Pattern Recognition*, vol. 43, no. 12, pp. 4028–4041, 2010. 8
- [32] C. L. Zitnick and S. B. Kang, “Stereo for image-based rendering using image over-segmentation,” *International Journal of Computer Vision*, vol. 75, no. 1, pp. 49–65, 2007. 8
- [33] R. Munoz-Salinas, E. Aguirre, M. Garca-Silvente, and A. Gonzalez, “People detection and tracking through stereo vision for human-robot interaction,” *MICAI*, pp. 337–346, 2005. 8
- [34] S. Agarwal, N. Snavely, I. Simon, S. M. Seitz, and R. Szeliski, “Building rome in a day,” *IEEE 12th International Conference on Computer Vision*, vol. 3, pp. 72–79, 2009. 8
- [35] T. Chen, M.-M. Chen, T. Ping, A. Shamir, and S.-M. Hu, “Sketch2photo: Internet image montage,” *ACM Transactions on Graphics*, vol. 28, no. 5, pp. 124:1–124:10, 2009. 8, 79, 82, 84, 85, 92, 97, 98, 99
- [36] A. Y.-S. Chia, S. Zhuo, R. K. Gupta, Y.-W. Tai, S.-Y. Cho, P. Tan, and S. Lin, “Semantic colorization with internet images,” *ACM Transactions on Graphics*, vol. 30, no. 6, pp. 156:1–156:8, 2011. 8

REFERENCES

- [37] J. Hays and A. A. Efros, “Scene completion using millions of photographs,” *ACM Transactions on Graphics (SIGGRAPH 2007)*, vol. 26, no. 3, 2007. 8, 79, 83, 92, 97, 98, 99, 107
- [38] T.-T. Lin, Y.-K. Hsiung, G.-L. Hong, H.-K. Chang, and F.-M. Lu, “Development of a virtual reality gis using stereo vision,” *Computers and Electronics in Agriculture*, vol. 63, no. 1, pp. 38–48, 2008. 12
- [39] D. Murray and C. Jennings, “Stereo vision based mapping and navigation for mobile robots,” *International Conference on Robotics and Automation*, pp. 858–863, 1997. 12
- [40] S. Helmer and D. G. Lowe, “Using stereo for object recognition,” *International Conference on Robotics and Automation*, 2010. 12
- [41] A. S. Malik, T. S. Choi, and H. Nisar, *Depth Map and 3D Imaging Applications: Algorithms and Technologies*. IGI Global, 2012. 12
- [42] D. Scharstein and R. Szeliski, “A taxonomy and evaluation of dense two-frame stereo correspondence algorithms,” *International Journal of Computer Vision*, vol. 47, no. 1, pp. 7–42, 2002. 13, 30
- [43] M. Z. Brown, D. Burschka, and G. D. Hager, “Advances in computational stereo,” *IEEE Transactions Pattern Analysis and Machine Intelligence*, vol. 25, no. 8, pp. 993–1008, 2003. 13
- [44] N. Lazaros, G. C. Sirakoulis, and A. Gasteratos, “Review of stereo vision algorithms: From software to hardware,” *International Journal of Optomechatronics*, vol. 2, pp. 435–462, 2008. 13
- [45] S. Yoon, S.-K. Park, S.-K. Kang, and Y. K. Kwak, “Fast correlation-based stereo matching with the reduction of systematic errors,” *Pattern Recognition Letters*, vol. 26, no. 14, pp. 2221–2231, 2005. 13, 15, 29
- [46] L. D. Stefano, M. Marchionni, and S. Mattoccia, “A fast area-based stereo matching algorithm,” *Image and Vision Computing*, vol. 22, no. 12, 2004. 13, 17

REFERENCES

- [47] P. Fua, “A parallel stereo algorithm that produces dense depth maps and preserves image features,” *Machine Vision and Applications*, vol. 6, no. 1, pp. 35–49, 1993. 13, 15, 20
- [48] S. A. Adhyapak, N. Kehtarnavaz, and M. Nadin, “Stereo matching via selective multiple windows,” *Journal of Electronic Imaging*, vol. 16, no. 1, 2007. 13, 14, 15, 17
- [49] P. F. Felzenszwalb and D. P. Huttenlocher, “Efficient belief propagation for early vision,” *IEEE Conference on Computer Vision and Pattern Recognition*, vol. 1, pp. 261–268, 2004. 13, 14
- [50] S. Birchfield and C. Tomasi, “Depth discontinuities by pixel-to-pixel stereo,” *International Journal of Computer Vision*, vol. 35, no. 3, pp. 269–293, 1999. 13, 14
- [51] Y. Boykov, O. Veksler, and R. Zabih, “Fast approximate energy minimization via graph cuts,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 23, no. 11, pp. 1222–1239, 2001. 13
- [52] J. Sun, N.-N. Zheng, and H.-Y. Shum, “Stereo matching using belief propagation,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 25, no. 7, pp. 787–800, 2003. 13
- [53] R. Sara, “How to teach stereoscopic matching?” *ELMAR*, pp. 445–453, 2010. 13, 14
- [54] R. K. Gupta and S.-Y. Cho, “Stereo correspondence using efficient hierarchical belief propagation,” *Neural Computing and Applications*, 2012. 13
- [55] A. Criminisi, A. Blake, C. Rother, J. Shotton, and P. H. Torr, “Efficient dense stereo with occlusions for new view-synthesis by four-state dynamic programming,” *International Journal of Computer Vision*, vol. 71, no. 1, pp. 89–110, 2007. 14
- [56] M. Okutomi, Y. Katayama, and S. Oka, “A simple stereo algorithm to recover precise object boundaries and smooth surfaces,” *International Journal of Computer Vision*, vol. 47, no. 1, pp. 138–144, 2002. 14, 15, 17

REFERENCES

- [57] J. Jeon, C. Kim, and Y.-S. Ho, “Sharp and dense disparity maps using multiple windows,” *Lecture Notes Computer Science*, vol. 2532, pp. 207–225, 2002. 14, 17
- [58] [Http://vision.middlebury.edu/stereo](http://vision.middlebury.edu/stereo). 14, 21, 22, 27
- [59] H. Hirschmuller, P. R. Innocent, and J. G. u, “Real-time correlation-based stereo vision with reduced border errors,” *International Journal of Computer Vision*, vol. 47, pp. 229–246, 2002. 15, 17, 23, 29
- [60] E. Binaghi, I. Gallo, G. Marino, and M. Raspanti, “Neural adaptive stereo matching,” *Pattern Recognition Letters*, vol. 25, pp. 1743–1758, 2004. 15
- [61] P. Mordohai and G. Medioni, “Stereo using monocular cues within the tensor voting framework,” *IEEE Transactions Pattern Analysis and Machine Intelligence*, vol. 28, no. 6, pp. 968–982, 2006. 15
- [62] K. Muhlmann, M. Dennis, H. Jurgen, and M. Reinhard, “Calculating dense disparity maps from color stereo images, an efficient implementation,” *International Journal of Computer Vision*, vol. 47, no. (1-3), pp. 79–88, 2002. 16
- [63] Y. Boykov, O. Veksler, and R. Zabih, “A variable window approach to early vision,” *IEEE Transactions Pattern Analysis and Machine Intelligence*, vol. 20, no. 12, pp. 1283–1294, 1998. 16
- [64] C. Zach, K. Karner, and H. Bischof, “Hierarchical disparity estimation with programmable 3d hardware,” *Proceedings of International Conference in Central Europe on Computer Graphics, Visualization and Computer Vision*, pp. 275–282, 2004. 17
- [65] Y. Zhao and G. Taubin, “Real-time stereo on gpgpu using progressive multi-resolution adaptive windows,” *Image and Vision Computing*, vol. 29, pp. 420–432, 2011. 17
- [66] F. Tombari, S. Mattoccia, L. D. Stefano, and E. Addimanda, “Near real-time stereo based on effective cost aggregation,” *International Conference on Pattern Recognition*, pp. 1–4, 2008. 17

REFERENCES

- [67] O. Veksler, “Stereo matching by compact window via minimum ratio cycle,” *IEEE International Conference on Computer Vision*, vol. 1, pp. 540–547, 2001. 17
- [68] —, “Fast variable window for stereo correspondence using integral images,” *IEEE Conference on Computer Vision and Pattern Recognition*, vol. 1, pp. 556–561, 2003. 17
- [69] R. K. Gupta and S.-Y. Cho, “Real-time stereo matching using adaptive binary window,” *3D Data Processing, Visualization and Transmission*, 2010. 18, 19, 25, 30
- [70] K.-J. Yoon and I. S. Kweon, “Adaptive support-weight approach for correspondence search,” *IEEE Transactions Pattern Analysis and Machine Intelligence*, vol. 28, no. 4, pp. 650–656, 2006. 18
- [71] —, “Correspondence search in the presence of specular highlights using specular-free two-band images,” *Proceedings of 7th Asian Conference on Computer Vision*, pp. 761–770, 2006. 18
- [72] E. T. Psota, J. Kowalczyk, J. Carlson, and L. C. Prez, “A local iterative refinement method for adaptive support-weight stereo matching,” *International Conference on Image Processing, Computer Vision, and Pattern Recognition*, 2011. 18
- [73] A. Hosni, M. Bleyer, M. Gelautz, and C. Rhemann, “Local stereo matching using geodesic support weights,” *International Conference on Image Processing*, 2009. 18
- [74] W. Yu, T. Chen, and J. C. Hoe, “Real time stereo vision using exponential step cost aggregation on gpu,” *International Conference on Image Processing*, pp. 4281–4284, 2009. 18
- [75] A. Hosni, M. Bleyer, C. Rhemann, M. Gelautz, and C. Rother, “Real-time local stereo matching using guided image filtering,” *IEEE International Conference on Multimedia and Expo*, pp. 1–6, 2011. 19

REFERENCES

- [76] K. He, J. Sun, and X. Tang, “Guided image filtering,” *European conference on Computer vision*, pp. 1–14, 2010. 19, 112
- [77] A. S. Ogale and Y. Aloimonos, “Shape and the stereo correspondence problem,” *International Conference on Computer Vision*, vol. 65, no. 3, pp. 147–162, 2005. 19
- [78] M. Humenberger, C. Zinner, M. Weber, W. Kubinger, and M. Vincze, “A fast stereo matching algorithm suitable for embedded real-time systems,” *Computer Vision and Image Understanding*, vol. 114, no. 11, pp. 1180–1202, 2010. 19, 30
- [79] L. Zhang, K. Zhang, T. S. Chang, G. Lafruit, G. K. Kuzmanov, and D. Verkest, “Real-time high-definition stereo matching on fpga,” *Proceedings of the 19th ACM/SIGDA international symposium on Field programmable gate arrays*, pp. 55–64, 2011. 19
- [80] A. Fusiello, E. Trucco, and A. Verri, “A compact algorithm for rectification of stereo pairs,” *Machine Vision and Applications*, vol. 12, pp. 16–22, 2000. 20
- [81] D. Marr and T. Poggio, “A computational theory of human stereo vision,” *Proc. of the Royal Society of London, Series B*, vol. 240, no. 1156, pp. 301–328, 1979. 23
- [82] A. Klaus, M. Sormann, and K. Karner, “Segment-based stereo matching using belief propagation and a self-adapting dissimilarity measure,” *International Conference on Pattern Recognition*, pp. 15–18, 2006. 25
- [83] G. Wang and D. Forsyth, “Object image retrieval by exploiting online knowledge resources,” *IEEE Conference on Computer Vision and Pattern Recognition*, 2008. 25
- [84] M. Bleyer, C. Rother, and P. Kohli, “Surface stereo with soft segmentation,” *IEEE Conference on Computer Vision and Pattern Recognition*, 2010. 25

REFERENCES

- [85] X. Mei, X. Sun, M. Zhou, S. Jiao, H. Wang, and X. Zhang, “On building an accurate stereo matching system on graphics hardware,” *International Conference on Computer Vision Workshop*, 2011. 25
- [86] CUDA C Programming Guide, NVIDIA Corporation. 28
- [87] K. Zhang, J. Lu, G. Lafruit, R. Lauwereins, and L. V. Gool, “Real-time accurate stereo with bitwise fast voting on cuda,” *International Conference on Computer Vision Workshops*, pp. 794–800, 2009. 30
- [88] M. Gong and Y.-H. Yang, “Near real-time reliable stereo matching using programmable graphics hardware,” *IEEE Conference on Computer Vision and Pattern Recognition*, vol. 1, pp. 924–931, 2005. 30
- [89] C. Richardt, D. Orr, I. Davies, A. Criminisi, and N. A. Dodgson, “Real-time spatiotemporal stereo matching using the dual-cross-bilateral grid,” *European Conference on Computer Vision*, vol. 6313, pp. 510–523, 2010. 30
- [90] K. Ambrosch and W. Kubinger, “Accurate hardware-based stereo vision,” *Computer Vision and Image Understanding*, vol. 114, no. 11, pp. 1303–1316, 2010. 30
- [91] J. Shotton, A. Fitzgibbon, M. Cook, T. Sharp, M. Finocchio, R. Moore, A. Kipman, and A. Blake, “Real-time human pose recognition in parts from single depth images,” *IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1297–1304, 2011. 44, 48
- [92] J. Sung, C. Ponce, B. Selman, and A. Saxena, “Human activity detection from rgbd images,” *AAAI workshop on Pattern, Activity and Intent Recognition*, 2011. 45, 48, 59, 61, 70, 71, 72
- [93] —, “Unstructured human activity detection from rgbd images,” *IEEE International Conference on Robotics and Automation*, pp. 842–849, 2012. 45, 48, 59, 61, 70, 71, 72, 74

REFERENCES

- [94] D. M. Gavrilu, “The visual analysis of human movement: A survey,” *Computer Vision and Image Understanding*, vol. 73, pp. 82–98, 1999. 45
- [95] P. Turaga, R. Chellappa, V. S. Subrahmanian, and O. Udrea, “Machine recognition of human activities: A survey,” *IEEE Trans. on Circuits and Systems for Video Technology*, vol. 18, no. 11, pp. 1473–1488, 2008. 45
- [96] R. Poppe, “A survey on vision-based human action recognition,” *Image and Vision Computing*, vol. 28, pp. 976–990, 2010. 45
- [97] J. Aggarwal and M. Ryoo, “Human activity analysis: A review,” *ACM Computing Surveys*, vol. 43, no. 3, pp. 16:1–16:43, 2011. 45
- [98] A. F. Bobick and J. W. Davis, “The recognition of human movement using temporal templates,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 23, pp. 257–267, 2001. 46
- [99] M.-K. Hu, “Visual pattern recognition by moment invariants,” *IRE Transactions on Information Theory*, vol. 8, 1962. 46
- [100] A. A. Efros, A. C. Berg, E. C. Berg, G. Mori, and J. Malik, “Recognizing action at a distance,” *International Conference on Computer Vision*, pp. 726–, 2003. 46
- [101] M. A. R. Ahad, T. Ogata, J. K. Tan, H. Kim, and S. Ishikawa, “Motion recognition approach to solve overwriting in complex actions,” *International Conference on Automatic Face and Gesture Recognition*, pp. 1–6, 2008. 46
- [102] S. Ali and M. Shah, “Human action recognition in videos using kinematic features and multiple instance learning,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 32, pp. 288–303, 2010. 46
- [103] A. Yilmaz and M. Shah, “Actions sketch: A novel action representation,” *IEEE Conference on Computer Vision and Pattern Recognition*, vol. 1, pp. 984–989, 2005. 46

REFERENCES

- [104] V. Parameswaran and R. Chellappa, “View invariance for human action recognition,” *International Journal of Computer Vision*, vol. 66, pp. 83–101, 2006. 46
- [105] I. Laptev, “On space-time interest points,” *International Journal of Computer Vision*, vol. 64, pp. 107–123, 2005. 47
- [106] P. Dollar, V. Rabaud, G. Cottrell, and S. Belongie, “Behavior recognition via sparse spatio-temporal features,” *2nd Joint IEEE International Workshop on Visual Surveillance and Performance Evaluation of Tracking and Surveillance*, pp. 65–72, 2005. 47
- [107] S. Nowozin, G. Bakir, and K. Tsuda, “Discriminative subsequence mining for action classification,” *International Conference on Computer Vision*, pp. 1–8, 2007. 47
- [108] C. Schuldt, I. Laptev, and B. Caputo, “Recognizing human actions: A local svm approach,” *International Conference on Pattern Recognition*, pp. 32–36, 2004. 47
- [109] A. Oikonomopoulos, I. Patras, and M. Pantic, “Spatiotemporal salient points for visual recognition of human actions,” *IEEE Transactions On Systems Man And Cybernetics (SMC) Part B: Cybernetics*, vol. 36, no. 3, pp. 710–719, 2006. 47
- [110] T. Kadir and M. Brady, “Scale saliency: a novel approach to salient feature and scale selection,” *International Conference on Visual Information Engineering*, pp. 25–28, 2003. 47
- [111] A. Gilbert, J. Illingworth, and R. Bowden, “Scale invariant action recognition using compound features mined from dense spatio-temporal corners,” *European Conference on Computer Vision*, pp. 222–233, 2008. 47
- [112] J. Liu, J. LUO, and M. Shah, “Recognizing realistic actions from videos in the wild,” *IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1–8, 2009. 47

REFERENCES

- [113] W. Li, Z. Zhang, and Z. Liu, “Action recognition based on a bag of 3d points,” *IEEE Conference on Computer Vision and Pattern Recognition*, pp. 9–14, 2010. 48
- [114] A. McCallum, D. Freitag, and F. C. N. Pereira, “Maximum entropy markov models for information extraction and segmentation,” *International Conference on Machine Learning*, pp. 591–598, 2000. 48
- [115] C. R. Wren, A. Azarbayejani, T. Darrell, and P. Pentland, “Pfinder: Real-time tracking of the human body,” *IEEE Transactions Pattern Analysis and Machine Intelligence*, vol. 19, no. 7, pp. 780–785, 1997. 48
- [116] G. Gordon, T. Darrell, M. Harville, and J. Woodfill, “Background estimation and removal based on range and color,” *IEEE Conference on Computer Vision and Pattern Recognition*, vol. 2, pp. 2459–2466, 1999. 48
- [117] C. Stauffer and W. E. L. Grimson, “Adaptive background mixture models for real-time tracking,” *IEEE Conference on Computer Vision and Pattern Recognition*, vol. 2, pp. 2246–2253, 1999. 48
- [118] A. Monnet, A. Mittal, N. Paragios, and V. Ramesh, “Background modeling and subtraction of dynamic scenes,” *International Conference on Computer Vision*, pp. 1305–1312, 2003. 48
- [119] Z. Zivkovic, “Improved adaptive gaussian mixture model for background subtraction,” *International Conference on Pattern Recognition*, pp. 28–31, 2004. 48, 49, 54, 55
- [120] P. Spagnolo, T. Orazio, M. Leo, and A. Distanto, “Moving object segmentation by background subtraction and temporal analysis,” *Image and Vision Computing*, vol. 24, no. 5, pp. 411–423, 2006. 48
- [121] S. T. Birchfield and S. Rangarajan, “Spatiograms versus histograms for region-based tracking,” *IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1158–1163, 2005. 48, 51

REFERENCES

- [122] C. O’Conaire, N. E. O’Connor, and A. F. Smeaton, “An improved spatiogram similarity measure for robust object localisation,” *IEEE International Conference on Acoustics, Speech, and Signal Processing*, pp. 15–20, 2007. 50, 52
- [123] B. Sabata, F. Arman, and J. K. Aggarwal, “Segmentation of 3d range images using pyramidal data structures,” *International Conference on Computer Vision*, pp. 662–666, 1990. 50
- [124] P. J. Besl, *Surface in Range Image Understanding*. Springer-Verlag New York, Inc. New York, 1989. 51
- [125] S. Kumar and A. Filipski, “Multiple sequence alignment: In pursuit of homologous dna positions,” *Genome Research*, pp. 127–135, 2007. 62
- [126] T. Attwood and D. Parry-Smith, *Introduction to Bioinformatics*. Addison Wesley Longman, 1999. 63
- [127] J. Aach and G. M. Church, “Aligning gene expression time series with time warping algorithms,” *Bioinformatics*, vol. 17, no. 6, pp. 495–508, 2001. 64
- [128] W. Iba and P. Langley, “Induction of one-level decision trees,” *Proceedings of the Ninth International Workshop on Machine Learning*, pp. 233–240, 1992. 67
- [129] Y. Freund and R. E. Schapire, “A decision-theoretic generalization of on-line learning and an application to boosting,” *Journal of Computer and System Sciences*, vol. 55, pp. 119–139, 1997. 67
- [130] D. Comaniciu and P. Meer, “Mean shift: A robust approach toward feature space analysis,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 24, no. 5, pp. 603–619, 2002. 68, 107
- [131] B. Leibe, A. Leonardis, and B. Schiele, “Combined object categorization and segmentation with an implicit shape model,” *European Conference on Computer Vision Workshop on Statistical Learning in Computer Vision*, pp. 17–32, 2004. 68

REFERENCES

- [132] [Http://www.ptgrey.com/support/kb/index.asp?a=4&q=103](http://www.ptgrey.com/support/kb/index.asp?a=4&q=103). 70
- [133] N. Otsu, “A threshold selection method from gray-level histograms,” *IEEE Transactions on Systems, Man and Cybernetics*, vol. 9, no. 1, pp. 62–66, 1975. 72
- [134] X. Liu, L. Wan, Y. Qu, T.-T. Wong, S. Lin, C.-S. Leung, and P.-A. Heng, “Intrinsic colorization,” *ACM Transactions on Graphics*, vol. 27, no. 5, pp. 152:1–152:9, 2008. 79, 101, 105, 113, 115
- [135] A. Torralba, R. Fergus, and W. T. Freeman, “80 million tiny images: A large data set for nonparametric object and scene recognition,” *IEEE Trans. on Pattern Analysis and Machine Intelligence*, vol. 30, no. 11, pp. 1985–1970, 2008. 79, 80
- [136] M. K. Johnson, K. Dale, S. Avidan, H. Pfister, W. T. Freeman, and W. Matusik, “Cg2real: Improving the realism of computer generated images using a large collection of photographs,” *IEEE Transactions on Visualization and Computer Graphics*, vol. 17, no. 9, pp. 1273–1285, 2011. 79
- [137] C. Goldberg, T. Chen, F.-L. Zhang, A. Shamir, and S.-M. Hu, “Data-driven object manipulation in images,” *Computer Graphics Forum*, vol. 31, no. 2, 2012. 79, 80
- [138] [Http://en.wikipedia.org/wiki/Flickr](http://en.wikipedia.org/wiki/Flickr). 79
- [139] [Http://www.photoweeklyonline.com/the-number-of-photos-on-facebook-is-exploding-infographic/](http://www.photoweeklyonline.com/the-number-of-photos-on-facebook-is-exploding-infographic/). 79
- [140] H. Tamura and N. Yokoya, “Image database systems: A survey,” *Pattern Recognition*, vol. 17, no. 1, pp. 29–43, 1984. 80
- [141] J. Eakins and M. Graham, *Content-based image retrieval*. Technical Report, University of Northumbria at Newcastle, 1999. 80
- [142] I. K. Sethi, I. L. Coman, and D. Stan, “Mining association rules between low-level image features and high-level concepts,” *Data Mining and Knowledge Discovery: Theory, Tools, and Technology III*, vol. 4384, 2001. 80

REFERENCES

- [143] W. ying Ma and B. Manjunath, “Netra: a toolbox for navigating large image databases,” *International Conference on Image Processing*, pp. 568–571, 1997. 80
- [144] B. Park, K. Lee, and S. Lee, “Color-based image retrieval using perceptually modified hausdorff distance,” *EURASIP Journal on Image and Video Processing*, 2007. 80, 81
- [145] Y. D. Chun, N. C. Kim, and I. H. Jang, “Content-based image retrieval using multiresolution color and texture features,” *IEEE Trans. on Multimedia*, vol. 10, no. 6, pp. 1073–1084, 2008. 80, 81, 82
- [146] J. Z. Wang, J. Li, and G. Wiederhold, “Simplicity: Semantics-sensitive integrated matching for picture libraries,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 23, no. 9, pp. 947–963, 2001. 80
- [147] H. Neemuchwala, A. Hero, and P. Carson, “Image matching using alpha-entropy measures and entropic graphs,” *Journal on Signal Processing*, vol. 85, pp. 277–296, 2005. 81
- [148] Y. Rui, T. S. Huang, and S.-F. Chang, “Image retrieval: Current techniques, promising directions, and open issues,” *Journal of Visual Communication and Image Representation*, vol. 10, pp. 39–62, 1999. 81, 82
- [149] A. W. Smeulders, M. Worring, S. Santini, A. Gupta, and R. Jain, “Content-based image retrieval at the end of the early years,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 22, no. 12, pp. 1349–1380, 2000. 81
- [150] Y. Liu, D. Zhang, G. Lu, and W.-Y. Ma, “A survey of content-based image retrieval with high-level semantics,” *Pattern Recognition*, vol. 40, pp. 262–282, 2007. 81
- [151] R. Datta, D. Joshi, J. Li, , and J. Z. Wang, “Image retrieval: Ideas, influences, and trends of the new age,” *ACM Computing Surveys*, vol. 40, pp. 5:1–5:60, 2008. 81

REFERENCES

- [152] R. Dorairaj and K. R. Namuduri, “Compact combination of mpeg-7 color and texture descriptors for image retrieval,” *Asilomar Conference on Signals, Systems and Computers*, vol. 1, pp. 387–391, 2004. 81, 82
- [153] D. Feng, W. Siu, and H. J. Zhang, *Multimedia Information Retrieval and Management*. Springer-Verlag New York, Inc. New York, 2003. 82
- [154] M. J. Swain and D. H. Ballard, “Color indexing,” *International Journal of Computer Vision*, vol. 7, pp. 11–32, 1991. 82
- [155] M. A. Stricker and M. Orengo, “Similarity of color images,” *SPIE Storage and Retrieval for Image and Video Databases*, pp. 381–392, 1995. 82
- [156] K. Meskaldji, S. Boucherkha, and S. Chikhi, “Color quantization and its impact on color histogram based image retrieval,” *International Conference on Networked Digital Technologies*, pp. 515–517, 2009. 82
- [157] J.-L. Shih and L.-H. Chen, “Color image retrieval based on primitives of color moments,” *International Conference on Recent Advances in Visual Information Systems*, pp. 88–94, 2002. 82
- [158] H. Tamura, S. Mori, and T. Yamawaki, “Textural features corresponding to visual perception,” *IEEE Transactions on Systems, Man, & Cybernetics*, vol. 8, no. 6, pp. 460–473, 1978. 82
- [159] J. R. Smith and S. fu Chang, “Automated binary texture feature sets for image retrieval,” *IEEE International Conference on Acoustics, Speech, and Signal Processing*, 1996. 82
- [160] W. Y. Ma and B. S. Manjunath, “A comparison of wavelet transform features for texture image annotation,” *IEEE International Conference on Image Processing*, vol. 3, 1995. 82
- [161] S. Berretti, A. D. Bimbo, and P. Pala, “Retrieval by shape similarity with perceptual distance and effective indexing,” *IEEE Transaction on Multimedia*, vol. 2, no. 4, pp. 225–239, 2000. 82

REFERENCES

- [162] V. Mezaris, I. Kompatsiaris, and M. G. Strintzis, “An ontology approach to object-based image retrieval,” *IEEE International Conference on Image Processing*, vol. 2, pp. 511–514, 2003. 82
- [163] J. F. Nunes, P. M. Moreira, and J. M. R. S. Tavares, “Shape based image retrieval and classification,” *Iberian Conference on Information Systems and Technologies*, pp. 1–6, 2010. 82
- [164] S. Belongie, J. Malik, and J. Puzhica, “Shape matching and object recognition using shape contexts,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 24, no. 4, pp. 509–532, 2002. 82, 86, 89
- [165] Q. Tian, N. Sebe, and M. S. Lew, “Image retrieval using wavelet-based salient points,” *Journal of Electronic Imaging*, vol. 10, no. 4, pp. 835–849, 2001. 82
- [166] K. Mikolajczyk and C. Schmid, “Scale and affine invariant interest point detectors,” *International Journal of Computer Vision*, vol. 60, no. 1, pp. 63–86, 2004. [Online]. Available: <http://lear.inrialpes.fr/pubs/2004/MS04> 83
- [167] K. Grauman and T. Darrell, “Efficient image matching with distributions of local invariant features,” *IEEE Conference on Computer Vision and Pattern Recognition*, 2005. 83
- [168] Y. Rubner, C. Tomasi, and L. J. Guibas, “The earth movers distance as a metric for image retrieval,” *International Journal of Computer Vision*, vol. 40, no. 2, pp. 99–121, 2000. 83
- [169] J. Zhu, S. C. Hoi, M. R. Lyu, and S. Yan, “Near-duplicate keyframe retrieval by nonrigid image matching,” *ACM international conference on Multimedia*, pp. 41–50, 2008. 83, 92, 97, 98, 99
- [170] A. Oliva and A. Torralba, “Building the gist of a scene: The role of global image features in recognition,” *Visual Perception, Progress in Brain Research*, vol. 155, 2006. 83, 107

REFERENCES

- [171] X. Li, C. Wu, C. Zach, S. Lazebnik, and J.-M. Frahm, “Modeling and recognition of landmark image collections using iconic scene graphs,” *European Conference on Computer Vision*, pp. 427–440, 2008. 83
- [172] J. Sivic and A. Zisserman, “Video google: A text retrieval approach to object matching in videos,” *International Conference on Computer Vision*, pp. 1470–1477, 2003. 83
- [173] D. G. Lowe, “Distinctive image features from scale-invariant keypoints,” *Int. Journal of Computer Vision*, vol. 60, pp. 91–110, 2004. 83, 90
- [174] C. Liu, J. Yuen, and A. Torralba, “Sift flow: dense correspondence across different scenes and its applications,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 33, no. 5, 2011. 83
- [175] D. Hoiem, R. Sukthankar, H. Schneiderman, and L. Huston, “Object-based image retrieval using the statistical structure of images,” *IEEE Conference on Computer Vision and Pattern Recognition*, 2004. 83
- [176] K. Tieu and P. Viola, “Boosting image retrieval,” *International Journal of Computer Vision*, vol. 56, no. 1-2, pp. 17–36, 2004. 83, 97
- [177] V. Jain and M. Verma, “Learning to re-rank: query-dependent image re-ranking using click data,” *International Conference on World Wide Web*, pp. 277–286, 2011. 83
- [178] F. Richter, S. Romberg, E. Hrster, and R. Lienhart, “Multimodal ranking for image search on community databases,” *International conference on Multimedia information retrieval*, pp. 63–72, 2010. 83
- [179] F. Schroff, A. Criminisi, and A. Zisserman, “Harvesting image databases from the web,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 33, no. 4, pp. 754–766, 2011. 83
- [180] F. Richter, S. Romberg, E. Hrster, and R. Lienhart, “Leveraging community metadata for multimodal image ranking,” *Multimedia Tools and Applications*, vol. 56, pp. 35–62, 2012. 83

REFERENCES

- [181] M. Crucianu, M. Ferecatu, and N. Boujemaa, “Relevance feedback for image retrieval: a short survey,” *In State of the Art in Audiovisual Content-Based Retrieval, Information Universal Access and Interaction including Datamodels and Languages*, 2004. 84
- [182] P. Auer and A. P. Leung, “Relevance feedback models for content-based image retrieval,” *Multimedia Analysis, Processing and Communications*, vol. 346, 2011. 84
- [183] T. Liu, J. Sun, N.-N. Zheng, X. Tang, and H.-Y. Shum, “Learning to detect a salient object,” *IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1–8, 2007. 85
- [184] C. Rother, V. Kolmogorov, and A. Blake, “Grabcut: Interactive foreground extraction using iterated graph cuts,” *ACM Trans. Graph. (Proc. SIGGRAPH)*, vol. 23, pp. 309–314, 2004. 85
- [185] O. G. Cula and K. J. Dana, “3d texture recognition using bidirectional feature histograms,” *International Journal of Computer Vision*, vol. 59, no. 1, pp. 33–60, 2004. 89
- [186] J. Zhang, M. Marszalek, S. Lazebnik, and C. Schmid, “Local features and kernels for classification of texture and object categories: A comprehensive study,” *Int. J. of Computer Vision*, vol. 73, no. 2, pp. 213–238, 2007. 89
- [187] [Http://images.google.com/](http://images.google.com/). 92, 97, 99
- [188] [Http://flickr.com](http://flickr.com). 92
- [189] [Http://picasa.google.com/](http://picasa.google.com/). 92
- [190] A. Levin, D. Lischinski, and Y. Weiss, “Colorization using optimization,” *ACM Trans. on Graphics*, vol. 23, no. 3, pp. 689–694, 2004. 101, 103, 105
- [191] Y.-C. Huang, Y.-S. Tung, J.-C. Chen, S.-W. Wang, and J.-L. Wu, “An adaptive edge detection based colorization algorithm and its applications,” *ACM international conference on Multimedia*, pp. 351–354, 2005. 101, 103

REFERENCES

- [192] L. Yatziv and G. Sapiro, “Fast image and video colorization using chrominance blending,” *IEEE Transactions on Image Processing*, vol. 15, no. 5, pp. 1120–1129, 2006. 101, 103
- [193] Y. Qu, T.-T. Wong, and P.-A. Heng, “Manga colorization,” *ACM Trans. on Graphics*, vol. 25, no. 3, pp. 1214–1220, 2006. 101, 103
- [194] Q. Luan, F. Wen, D. Cohen-Or, L. Liang, Y.-Q. Xu, and H.-Y. Shum, “Natural image colorization,” *Proc. Eurographics Symposium on Rendering*, 2007. 101, 103
- [195] N. Dongdong, M. Qinyong, M. Lizhuang, and X. Shuangjiu, “Optimization based grayscale image colorization,” *Pattern Recognition Letters*, vol. 28, pp. 1445–1451, 2007. 101
- [196] O. Lezoray, V.-T. Ta, and A. Elmoataz, “Nonlocal graph regularization for image colorization,” *Int. Conf. on Pattern Recognition*, pp. 1–4, 2008. 101
- [197] E. Reinhard, M. Ashikhmin, B. Gooch, and P. Shirley, “Color transfer between images,” *IEEE Computer Graphics and Applications*, vol. 21, no. 5, pp. 34–41, 2001. 101
- [198] T. Welsh, M. Ashikhmin, and K. Mueller, “Transferring color to greyscale images,” *ACM Transactions on Graphics*, vol. 21, no. 3, pp. 277–280, 2002. 101, 104, 113, 116, 117, 118, 122
- [199] A. Hertzmann, C. E. Jacobs, N. Oliver, B. Curless, and D. H. Salesin, “Image analogies,” *Proc. SIGGRAPH*, pp. 327–340, 2001. 101
- [200] R. Irony, D. Cohen-Or, and D. Lischinski, “Colorization by example,” *Eurographics Symposium on Rendering*, 2005. 101, 105
- [201] G. Charpiat, M. Hofmann, and B. Schlkopf, “Automatic image colorization via multimodal predictions,” *Proc. European Conf. on Computer Vision*, 2008. 101, 105, 113, 116, 117, 118, 122
- [202] G. Charpiat, I. Bezrukov, Y. Altun, M. Hofmann, and B. Schlkopf, *Computational Photography: Methods and Applications*. CRC Press, 2010. 101

REFERENCES

- [203] A. Hertzmann, C. E. Jacobs, N. Oliver, B. Curless, and D. H. Salesin, “Image analogies,” *SIGGRAPH*, pp. 327–340, 2001. 101
- [204] R. Szeliski, “Locally adapted hierarchical basis preconditioning,” *ACM Trans. on Graph.*, vol. 25, no. 3, pp. 1135–1143, 2006. 103
- [205] O. Dalmau-Cedeno, M. Rivera, and P. P. Mayorga, “Computing the alpha-channel with probabilistic segmentation for image colorization,” *International Conference on Computer Vision*, pp. 1–7, 2007. 103
- [206] P. Lagodzinski and B. Smolka, “Colorization of medical images,” *Proceedings of APSIPA Annual Summit and Conference*, pp. 769–772, 2009. 103
- [207] D. Šýkora, J. Buriánek, and J. Žára, “Unsupervised colorization of black-and-white cartoons,” *Proc. 3rd international symposium on Non-photorealistic animation and rendering*, pp. 121–127, 2004. 104
- [208] R. K. Gupta, A. Y.-S. Chia, D. Rajan, and H. Zhiyong, “A learning-based approach for automatic image and video colorization,” *Computer Graphics International*, 2012. 105
- [209] R. K. Gupta, A. Y.-S. Chia, D. Rajan, E. S. Ng, and H. Zhiyong, “Image colorization using similar images,” *ACM Multimedia*, 2012. 105, 113, 116, 117, 118, 122
- [210] Y. Li, J. Sun, C.-K. Tang, and H.-Y. Shum, “Lazy snapping,” *ACM Transactions on Graphics*, vol. 23, no. 3, pp. 303–308, 2004. 106
- [211] A. Oliva and A. Torralba, “The role of context in object recognition,” *Trends in Cognitive Sciences*, vol. 11, no. 12, 2007. 106
- [212] N. Komodakis and G. Tziritas, “Approximate labeling via graph-cuts based on linear programming,” *IEEE Trans. on Pattern Analysis and Machine Intelligence*, vol. 29, no. 8, pp. 1436–1453, 2007. 110
- [213] D. Hasler and S. Strunk, “Measuring colourfulness in natural images,” *SPIE Electronic Imaging: Human Vision and Electronic Imaging VIII*, vol. 5007, pp. 87–95, 2003. 111

REFERENCES

- [214] Y.-W. Tai, J. Jia, and C.-K. Tang, “Local color transfer via probabilistic segmentation by expectation-maximization,” *IEEE Conference on Computer Vision and Pattern Recognition*, pp. 747–754, 2005. 113, 116, 117, 118, 122