COMPUTER VISION METHODS FOR LARGE-SCALE ONLINE CLUSTERING AND QUANTITATIVE DERMATOLOGY

BY SIDDHARTH K. MADAN

A dissertation submitted to the

Graduate School—New Brunswick

Rutgers, The State University of New Jersey

in partial fulfillment of the requirements

for the degree of

Doctor of Philosophy

Graduate Program in Electrical and Computer Engineering

Written under the direction of

Dr. Kristin J. Dana

and approved by

_____

_____

_____

_____

_____

New Brunswick, New Jersey

May, 2013

ABSTRACT OF THE DISSERTATION

Computer Vision Methods for Large-scale Online Clustering and Quantitative Dermatology

by Siddharth K. Madan

Dissertation Director: Dr. Kristin J. Dana

In modern computer vision applications where datasets are large and updates with new data may be ongoing, methods of online clustering are extremely important. Online clustering algorithms incrementally cluster the data points, use a fraction of the dataset memory, and update the clustering decisions when new data comes in. In this thesis we adapt a classic online clustering algorithm called Balanced Iterative Reducing and Clustering using Hierarchies (BIRCH) to incrementally cluster large datasets of features commonly used in computer vision, e.g., 840K color SIFT descriptors, 1.09 million color patches, 60K outlier corrupted grayscale patches, and 700K grayscale SIFT descriptors. We use the algorithm to cluster datasets consisting of non-convex clusters, e.g., Hopkins 155 3-D motion segmentation dataset. We call the adapted version *modified-BIRCH* (m-BIRCH). BIRCH was originally developed by the database management community. Modifications made in m-BIRCH enable data driven parameter selection and effectively handle varying density regions in the feature space. Data driven parameter selection automatically controls the level of coarseness of the data summarization. Effective handling of varying density regions is necessary to well represent the different density regions in the data summarization. Our implementation of the algorithm provides a useful clustering tool and is made publicly available.

In the second part of the thesis, we present a micro-level feature based approach to register *time-lapse* skin images acquired over an extended period of time and *multimodal* skin images acquired in quick succession. Misregistration between the images makes it difficult to perform quantitative analysis and track the progression of skin disease. We demonstrate the utility of both types of registration, by employing the results for two quantitative dermatology tasks: 1) automatic detection of acne-like regions, and 2) separation of surface and subsurface reflection. Additionally, we have created a time-lapse video showing the registered time-lapse images, which clearly brings out the evolution of acne lesions with time.

Acknowledgements

I would like to thank my advisor Dr. Kristin Dana for her invaluable guidance throughout the PhD program. Countless discussions with her have played an indispensable role in shaping the thesis. It has been a great learning experience working under her guidance.

I would like to thank Dr. Oana Cula for for her input and collaboration. I would like to thank her for giving me an opportunity to visit and interact with scientists at Johnson & Johnson's research facility.

I would like to thank my colleagues Alexander Wu, Wenjia Yuan, Jayant Silva, Prateek Prasanna, and Parneet Kaur for the wonderful time and discussions in the lab.

Table of Contents

List of Tables

List of Figures

Chapter 1

Problem Definition

We address two challenging image-based problems: a) online clustering of large datasets for computer vision applications, b) precise point to point registration of skin images acquired in dermatology clinical studies. We adapt a classic online clustering algorithm called Balanced Iterative Reducing and Clustering Using Hierarchies (BIRCH) to cluster large datasets of features commonly used in computer vision. We call the adapted version *modified-BIRCH* (m-BIRCH). We use a micro-level feature based registration approach for *multimodal* and *time-lapse* registration of skin images acquired in dermatology clinical studies.

1.1 Online Clustering

Clustering large data sets is a fundamental task required in many modern computer vision problems like object recognition and scene classification. Traditionally, batch based clustering algorithms that process all the points at once have been used for clustering datasets. However, batch based clustering methods have three major drawbacks: a) processing all points at once is very inefficient, b) the algorithms require large amount of memory, equal to the entire dataset size, and c) the algorithms cannot incrementally update the clustering decisions when new data comes in. In order to efficiently cluster very large datasets and datasets to which additional points may be added in the future, in recent years there has been significant interest in developing online clustering algorithms. Online clustering algorithms incrementally and efficiently cluster the data points, use a fraction of the memory required by the entire dataset, and adapt clustering decisions as new data points come in.

1.1.1 Algorithms for Online Clustering

Online clustering is being widely researched by the computer vision community, and numerous interesting approaches have been proposed. For example, in [31] the authors proposed an incremental mean-shift based algorithm to build discriminative vocabularies. The authors load a subset of points in the dataset, and estimate cluster centers using mean-shift. Each loaded point is assigned to the closest cluster center only if the point is within a fixed radius from the center. If a point cannot be assigned to any cluster center, then it is saved and re-tested with new cluster centers discovered when additional points are loaded. In [19], the authors proposed an algorithm to incrementally cluster data using the Dirichlet process mixture models. The authors concisely represent the dataset as a collection of clumps. Each clump represents a set of points in the feature space. Final clusters are obtained using variational inference for Dirichlet process mixtures [6, 35]. In [38] the authors use online non-parametric Bayesian clustering and and Dirichlet process priors for online video segmentation. In [47] the authors presented an online approach to learn sparse dictionaries. Original data points are represented as a sparse linear combination of the learned code words. Online sparse dictionaries are extensively used in computer vision [46, 45, 63].

The discussed online algorithms have numerous limitations. Algorithms proposed in [19, 47] can make online updates only for one specific global clustering algorithm. The online update procedure in [19] can be used only for Dirichlet mixture models and [47] can be used only when data points need to be represented as a sparse linear combination of the learned code words. The algorithm in [31] makes final clustering decisions by running mean-shift on subsets of the data set, and the preset radius parameter remains constant throughout the clustering process. However, as large amounts of new data comes in, the mode location detected by mean-shift can change significantly, and the preset radius becomes imprecise, because cluster sizes will increase.

1.1.2 m-BIRCH Based Online Clustering

We adapt a classic online clustering algorithm called BIRCH [82, 83] for computer vision applications. We call the new version *modified-BIRCH* (m-BIRCH). BIRCH was originally developed by the database management community and has become very popular in that community [11, 8, 32]. The algorithm has won the SIGMOD test of time award. However, BIRCH has not been widely used in the area of computer vision. BIRCH has a number of strengths which makes it a useful algorithm for computer vision applications. The incremental update step of BIRCH can be used with different batch clustering algorithms, resulting in incremental versions of multiple batch clustering algorithms. We use BIRCH with spectral, Gaussian mixture model (GMM), and K-means based clustering. This is the first time that BIRCH has been used with spectral and Gaussian mixture model based clustering. BIRCH makes final clustering decisions on a concise summary of the whole dataset and not on subsets of points; therefore the final clusters reflect the point distribution in the entire dataset. BIRCH can generate correct clusters in presence of outliers.

The m-BIRCH algorithm makes an automatic data driven parameter selection. BIRCH obtains a concise representation of points close to each other in the feature space, and the *threshold* parameter controls the maximum average inter-point distance in the concise representation. The ideal method for data driven selection of the threshold parameter is computationally infeasible. Instead of manually choosing the threshold value or using arbitrary statistics like mean and median, m-BIRCH systematically approximates the computationally infeasible ideal approach. The m-BIRCH algorithm separately processes regions of different density in the feature space. Separate processing ensures that regions of different densities contribute in generating the final cluster centers.

We use m-BIRCH to cluster large datasets of features commonly used in computer vision within 10% to 20% of the dataset memory. Table 1.1 summarizes the features we clustered using m-BIRCH. We cluster 840,000 384-dimensional color SIFT descriptors [39] evaluated at regular grid points and 1,093,680 108-dimensional color patches from

Features Incrementally Clustered with m-BIRCH

Feature	Number of Points	Feature Dimension
Color SIFT	840,000	384
Color Patches	1,093,680	108
Grayscale Patches	60,000	50
Grayscale SIFT	707,877	128

Table 1.1: We use m-BIRCH to cluster large datasets of features commonly used in computer vision. The clustering is done using just 10 to 20% of the dataset memory.

the Oliva and Torralba scene classification dataset [58]. We use the generated clusters to identify the scenes in a separate test set. We test m-BIRCH's outlier handling capability by clustering outlier corrupted grayscale patches from the MNIST digit dataset [37]; visual and quantitative assessment verify that m-BIRCH generates proper clusters in presence of outliers. We cluster 707,877 128-dimensional grayscale SIFT descriptors evaluated at affine-invariant Hessian regions [50] from the Oxford buildings dataset [59]. We use the generated clusters to classify query images supplied with the dataset. Our experiments show that m-BIRCH can efficiently cluster large datasets of features to generate discriminative vocabularies.

1.2 Skin Registration in Dermatology Clinical Studies

Numerous dermatology clinical studies require the acquisition of high resolution skin images at different times and under different modalities. Skin images acquired under different modalities in each imaging session capture complementary features in skin appearance that are not observable under ordinary visible light. For example, it is common to acquire multimodal images of skin pathology under polarized light [3, 52, 34] in quick succession, i.e., within an interval of a second or less. Polarized light imaging offers the key advantage that surface structures can be separated from subsurface structures. Imaging at different imaging sessions, with time interval of days or weeks, captures the appearance of skin during the different stages of the pathology, and is used by dermatologists [25, 71, 64] to track the changes in skin appearance and evolution of disease over time.

For multimodal images acquired within an imaging session and for images acquired in different imaging sessions, spatial alignment is problematic. For multimodal images acquired in quick succession, breathing and involuntary movement cause minute misregistration. For images acquired in different imaging sessions, larger misregistration occurs due to the difference in the relative position between the sensor and the subject. Misregistration between skin images makes it difficult to perform quantitative or qualitative processing on the skin images, and track the progression of the skin disease during the study. We use micro-level features like pores, wrinkles, and other skin texture markings to precisely register the skin images up to sub-pixel accuracy.

1.2.1 Registration Methods

Registration is a widely studied topic and detailed surveys discuss various registration methods [84, 70]. We discuss three major types of registration algorithms: feature-based, mutual information based, and deformable models based registration methods.

Feature-based methods generally extract features and match them to estimate a transformation between the two images. For example, in [12] the authors detect and match Harris corners for mosaicing images acquired by camera rotation, and in [55] the authors used a modified version of SIFT to register a set of ultrasound volumes. In [81], a wide variety of images are registered using SIFT features with multi-scale corner and face features. The authors start by estimating a transformation in a small region around a matched feature pair using the corner and face features, and expand the region to refine the estimate until the entire overlap between the images is covered. Our approach is also feature based, but we use micro-features to achieve sub-pixel, pore-level alignment. This is the first work that uses pores and other barely discenible markings and demonstrates that these are stable features for registration.

Mutual information based registration methods [60, 44, 9] register images by maximizing the mutual information between them. The registration approach assumes that the intensities in the two images being registered are statistically dependent; therefore uncertainty of intensity values in the input image is minimum when registered to the reference image. However, for these methods intensities of neighboring pixels

are assumed independent and underlying probability of density estimates use limited accuracy histogram binning or Parzen windowing.

Deformable models based methods have the ability to handle non-rigid motion. For example in [65] the authors use cubic B-splines to register breast MRI images and in [49] the authors use cubic B-splines to register PET and CT chest images. Active appearance models [66, 10, 79], are deformable models used in computer vision for registering and tracking faces. However, active appearance models rely on macro-level features like lips, face boundary, nose, and eyes instead of micro-level features.

1.2.2 Micro-Level Feature Based Registration

We use the micro-level feature-based registration approach for registering both multimodal skin images acquired in a particular imaging session of a clinical study and registering skin images acquired in different imaging sessions of a clinical study. In particular we register: a) multimodal images acquired under polarized, fluorescence, and visible light within a second time interval, b) sequence of time-lapse skin images with acne lesions acquired over a three month period clinical study.

We show that performing quantitative processing directly on the input multimodal images acquired in quick succession results in misregistration artifacts. The misregistration artifacts can be eliminated by precisely aligning the polarized images using the micro-level feature-based registration approach. In particular, the multimodal images corresponding to cross and parallel polarized images can be used to separate reflectance into subsurface and surface reflectance by a pixel-by-pixel subtraction. However, the input images must be exactly aligned for this method to work.

For time-lapse images, skin regions with acne lesions can be used to analyze the change in appearance of acne lesions and the evolution of disease over time. However, the misregistration between the time-lapse images makes it difficult to perform any analysis on the images. We show that the time-lapse set can be precisely registered using the micro-level feature-based registration approach. We use the precisely registered time-lapse images to create and analyze a time-lapse video, which to the best of our knowledge is the first of its kind. This video impressively demonstrates the

power of time-lapse imaging in evaluating the evolution of lesions over a time sequence. We demonstrate the quantitative use of the registered set of time-lapse sequence by building a classifier for detecting acne-like regions, i.e., regions which appear like acne lesions and whose appearance evolves with time. Detecting acne-like regions serves as a promising first step towards the development of a computational acne lesion detector. A computational lesion detector can be used to automate the lesion counting process widely used to test the efficacy of an acne treatment procedure [77, 76, 13]. Lesion counting is often done manually and is time consuming, tedious, and prone to errors. Learning based methods are being increasingly explored for computer assisted recognition of skin diseases. For example, in [17] and [67] the authors explored the use of neural networks for detecting cancerous skin lesions, and in [15] the authors trained a classifier to distinguish between different skin pathologies.

Videos showing the misregistered and registered multimodal and time-lapse skin images are available online [1], and all images used to create the time-lapse video are shown in section 5.3. We have developed a toolbox to precisely register skin images using micro-level features, which is also available online. [2]

[1] http://www.ece.rutgers.edu/~kdana/Research.html

[2] http://www.eden.rutgers.edu/~smadan/skin_reg.html

Chapter 2

m-BIRCH Algorithm

We discuss the classic BIRCH and m-BIRCH online clustering algorithms. The algorithms incrementally generate a concise dataset summary and update the summary when new data comes in. Final clusters are obtained by adapting a global clustering algorithm to work on the dataset summary. The m-BIRCH algorithm makes an automatic data driven parameter selection and obtains a summary which well represents regions of different density in the feature space. In this chapter we discuss the data summary generation and update, and outlier elimination steps. We discuss the adaptation of global clustering in chapter 3.

2.1 BIRCH Algorithm

2.1.1 Clustering-feature

Clustering-feature provides concise summary of a set of points close to each other in the feature space and retains information necessary for data clustering.

Let $S = \{\mathbf{x}_1, \ldots, \mathbf{x}_N\}$ be a set of d-dimensional points. The size, D, of set S is defined as the average distance between any two points,

$$D = \sqrt{\frac{\sum_{i=1}^{N}\sum_{j=1}^{N} \|(\mathbf{x}_i - \mathbf{x}_j)\|_2^2}{N(N-1)}}. \tag{2.1}$$

Let S_1 and S_2 be two sets of d-dimensional points. The distance, $d(S_1, S_2)$, between the two sets is defined as the average inter-point distance,

$$d(S_1, S_2) = \sqrt{\frac{\sum_{i=1}^{N_1}\sum_{j=1}^{N_2} \|\mathbf{x}_i^1 - \mathbf{x}_j^2\|_2^2}{N_1 N_2}}, \tag{2.2}$$

where $\mathbf{x}_i^1$ are the points in S_1 and $\mathbf{x}_i^2$ are the points in S_2.

The clustering-feature representation, **CF**, of set S is given by,

$$\mathbf{CF} = (N, \mathbf{L}, s)\,, \tag{2.3}$$

where N is the number of points in the set, $\mathbf{L} = \sum_{i=1}^{N} \mathbf{x}_i$ is the sum of all the points in the set, and the scalar $s = \sum_{i=1}^{N} \|\mathbf{x}_i\|_2^2$ is the sum of squares of all the components of all the points in the set. Individual points are simply sets containing only one point. Therefore an individual point can also be represented by a clustering-feature.

Clustering-feature concisely represents the set S, and retains information to calculate the centroid, $\mathbf{x}_c$, of the set, the set size D, and the distance, $d(S_1, S_2)$, between two sets. These can be expressed as,

$$\mathbf{x}_c = \frac{\mathbf{L}}{N}, \tag{2.4}$$

$$D = \begin{cases} 0 & \text{if } N = 1, \\ \sqrt{\frac{2s}{(N-1)} - \frac{2\mathbf{L}^\top \mathbf{L}}{N(N-1)}} & \text{if } N > 1, \end{cases} \tag{2.5}$$

$$d(S_1, S_2)^2 = \frac{s_1}{N_1} + \frac{s_2}{N_2} - \frac{2}{N_1 N_2} \mathbf{L}_1^\top \mathbf{L}_2, \tag{2.6}$$

where $\mathbf{CF}_1 = (N_1, \mathbf{L}_1, s_1)$ and $\mathbf{CF}_2 = (N_2, \mathbf{L}_2, s_2)$ are the clustering-feature representations of S_1 and S_2. Further, the clustering-feature of the union, $S_1 \bigcup S_2$, is simply the sum, $\mathbf{CF}_1 + \mathbf{CF}_2$, of the clustering-feature of each set.

Working with clustering-features instead of individual points reduces both the amount of memory required and the number of entities to be processed.

2.1.2 Simple Example of BIRCH Clustering

We demonstrate the use of BIRCH using a simple illustrative example. Figure 2.1 (a) shows an example 121×81 image from the Berkeley image segmentation dataset [48]. Figure 2.1 (b) shows the plot of the red, green, and blue intensity values. The dataset consists of 9801 points. We demonstrate how BIRCH works by clustering the intensity values to seven clusters.

BIRCH summarizes the entire dataset as a collection of clustering-features, where each clustering-feature represents a group of points close to each other in the feature

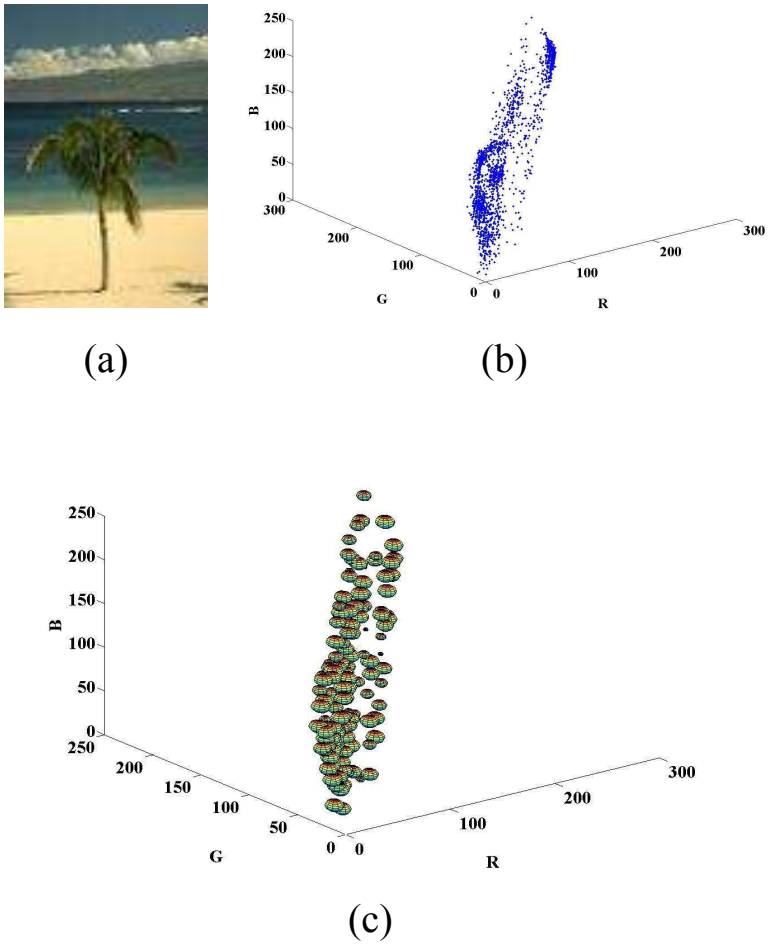


Figure 2.1: (a): 121×81 image from the Berkeley image segmentation dataset [48]. (b): 9801 red, green, and blue intensity values of the points in the image. (c): 299 clustering-features used by BIRCH to summarize the dataset. Each clustering-feature is represented using a sphere placed at the centroid of the points represented by the clustering-feature. Working with clustering-features requires much lesser memory and the processing of much lesser number of entities.

space. Figure 2.1 (c) shows the clustering-feature summarization. The center of each sphere is placed at the centroid of the points represented by the clustering-feature. The entire processing was done within 10% of the dataset memory, and BIRCH summarized the dataset using 299 clustering-features. The reduction of memory and number of entities to be processed makes BIRCH a powerful tool in clustering. Clustering the points under eight byte double precision requires processing 9801 points and 230 KB of memory. However, clustering the clustering-features requires the processing of only 299 entities and only 11 KB of memory. Although the dataset considered in Figure 2.1 is low

Cluster Centers	
Batch K-means	BIRCH + K-means
(41.08, 52.53, 25.08)	(42.05, 53.08, 24.32)
(46.38, 68.82, 66.76)	(47.05, 68.98, 65.73)
(77.58, 99.70, 98.04)	(77.26, 99.51, 98.02)
(79.62, 84.02, 40.84)	(83.53, 86.22, 40.17)
(125.69, 134.90, 103.57)	(123.59, 133.33, 102.57)
(173.98, 171.33, 135.98)	(171.92, 169.63, 134.71)
(247.22, 221.39, 159.92)	(246.96, 221.24, 159.84)

Table 2.1: Cluster centers obtained by running K-means on all the points in the dataset (left) and the concise 299 clustering-feature summarization generated using 10% of the dataset memory (right) are almost the same.

dimensional and dense, our experiments show that suitable condensation is obtained on high dimensional datasets as well. BIRCH incrementally updates the clustering-feature collection when new data points are added. Final clusters are obtained by adapting a global clustering algorithm to work on the clustering-features.

Table 2.1 shows the centers obtained by directly applying K-means to data points and applying K-means to the clustering-features. Figure 2.2 (a) and (b) show the corresponding segmented images. Note that the cluster centers and the segmented images obtained by running K-means on the concise 299 clustering-feature summarization incrementally generated using just 10% of the dataset memory is very similar to those obtained by running K-means on every individual point in the dataset.

2.1.3 Dataset Summarization

BIRCH obtains the clustering-feature summarization of the dataset by incrementally building a height-balanced **CF**-tree [82]. Each node of the **CF**-tree stores a maximum of B clustering-features, and the size of each clustering-feature in the leaf nodes is less

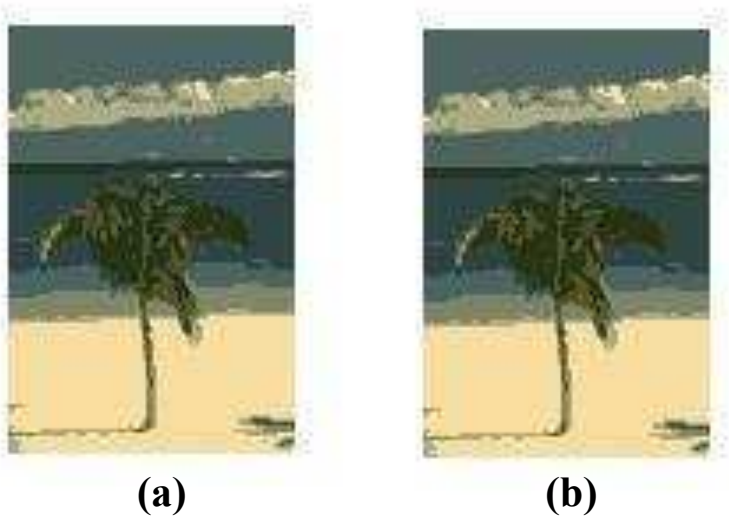


Figure 2.2: Segmented images obtained by applying K-means directly to the points in the dataset (a) and applying K-means to the concise clustering-feature representation (b). The segmented image obtained by applying K-means on the concise clustering-feature representation is very similar to the segmented image obtained by applying K-means to all the points.

than a threshold T. An initial value of T is chosen, which is refined as the tree is built. The **CF**-tree places clustering-features close to each other in the same leaf nodes. The clustering-features in the leaf nodes provide a concise summary of the dataset.

Building the **CF**-tree consists of alternating between two steps: insertion and rebuild. During the insertion step clustering-features are added to the tree, which increases the memory occupied by the tree. When memory occupied by the **CF**-tree becomes equal to the maximum available memory, a smaller tree is rebuilt and the current threshold value T is increased to $T + \Delta T$. The insertion and rebuild steps are repeated until all points are added to the tree.

Insertion

Suppose we want to insert a new clustering-feature, $\mathbf{CF}_{in}$, in the **CF**-tree. In any node, $node_i$, of the tree the sum of all the clustering-features, $\mathbf{CF}_{node_i} = \sum_j \mathbf{CF}_j$, represents the set $\bigcup_j S_j$, where S_j is the set of points represented by the clustering-feature $\mathbf{CF}_j$. $\mathbf{CF}_{node_i}$ is stored in the parent node. We call $node_i$ the child of $\mathbf{CF}_{node_i}$. The parent node is linked to the child node of each clustering-feature stored in it.

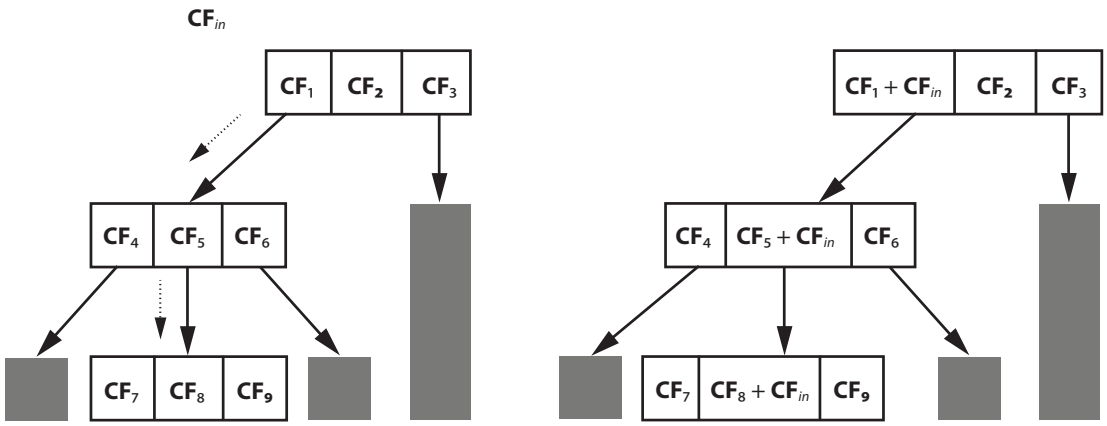


Figure 2.3: (a, b): **CF**-tree before and after $\mathbf{CF}_{in}$ is inserted. $\mathbf{CF}_{in}$ gets absorbed by $\mathbf{CF}_8$. The grey rectangles denote sub-trees, which are not encountered while inserting $\mathbf{CF}_{in}$. The dotted arrows in (a) indicate the path taken during insertion of $\mathbf{CF}_{in}$ to reach the leaf node.

In order to insert $\mathbf{CF}_{in}$ we find the closest leaf node by traversing the tree starting at the root node; at each level we proceed to the child node of the closest clustering-feature. We step through the leaf node to check if $\mathbf{CF}_{in}$ can be *absorbed* in any clustering-feature without violating the threshold condition. $\mathbf{CF}_{in}$ can be absorbed in a leaf clustering-feature only if the size of the resulting clustering-feature is less than threshold T. If $\mathbf{CF}_{in}$ cannot be absorbed, we check whether the leaf has less than B clustering-features. If the leaf node has less than B clustering-features, then $\mathbf{CF}_{in}$ can be *accommodated* and stored in the leaf node. In the event of a successful absorption or accommodation, $\mathbf{CF}_{in}$ is added to each clustering-feature which was determined to be the closest when the tree was traversed from the root to the leaf node.

Figure 2.3 illustrates the tree insertion process, when $\mathbf{CF}_{in}$ is successfully absorbed in a leaf clustering-feature. Figure 2.3 (a) shows the **CF**-tree before $\mathbf{CF}_{in}$ is inserted. The dotted arrows show the path taken when the tree was traversed from root to the leaf node. Figure 2.3 (b) shows **CF**-tree after $\mathbf{CF}_{in}$ is inserted. Successful absorption of $\mathbf{CF}_{in}$ does not result in increase of memory used by the **CF**-tree. Successful accommodation of $\mathbf{CF}_{in}$ does result in an increase of memory used by the **CF**-tree.

Node *split* occurs when $\mathbf{CF}_{in}$ can neither be absorbed nor accommodated in the

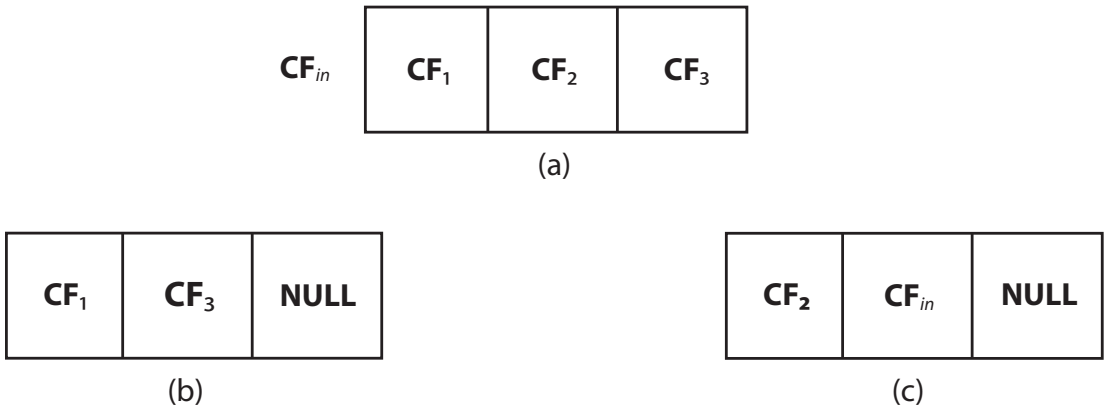


Figure 2.4: (a): Original leaf node. (b, c): *nodeA* and *nodeB* generated after the split, with clustering-features distributed between them.

leaf node. Node split results in the generation of two new nodes, *nodeA* and *nodeB*. Clustering-features of the original leaf node and $\mathbf{CF}_{in}$ are distributed between the two new nodes. The furthest pair of clustering-features are found; one clustering-feature of the pair is stored in *nodeA* and the other in *nodeB*. Remaining clustering-features are distributed depending on the distance between the clustering-feature and the current clustering-feature representation of *nodeA* and *nodeB*. Figure 2.4 illustrates the distribution of clustering-features when a node split takes place. Figure 2.4 (a) shows the original tree node, and Figure 2.4 (b) and (c) show *nodeA* and *nodeB* generated after the split. The *NULL* in Figure 2.4 (b) and (c) indicates no clustering-feature is present. Suppose $\mathbf{CF}_1$ and $\mathbf{CF}_2$ are the furthest pair; thus $\mathbf{CF}_1$ is stored in *nodeA* and $\mathbf{CF}_2$ is stored in *nodeB*. $\mathbf{CF}_3$ is closer to $\mathbf{CF}_1$ than to $\mathbf{CF}_2$; thus $\mathbf{CF}_3$ is stored in *nodeA*. $\mathbf{CF}_{in}$ is closer to $\mathbf{CF}_2$ than to $\mathbf{CF}_1 + \mathbf{CF}_3$; thus $\mathbf{CF}_1$ is stored in *nodeB*. Once the clustering-features are distributed *nodeA* replaces the original leaf node in the **CF**-tree, and the information along the path traversed to reach the leaf node is updated. Let $\mathbf{CF}_{nodeB}$ be the sum of all the clustering-features in *nodeB*. We attempt to accommodate $\mathbf{CF}_{nodeB}$ in the parent of the original leaf node. If $\mathbf{CF}_{nodeB}$ is successfully accommodated, then parent node is linked to *nodeB*, and information in the **CF**-tree is updated. If $\mathbf{CF}_{nodeB}$ cannot be accommodated in the parent node, then the parent node itself is split and the procedure is repeated. If the split propagates to the root and the root is split, then the tree height increases by one.

Rebuild

Suppose in the current tree, the leaf nodes are numbered from left to right starting with one, and the i^{th} leaf node in the original tree is denoted as $node_i$. Clustering-features of $node_1$ are inserted in the new tree by creating a path from root to leaf node identical to the path in the current tree. The first clustering-feature of $node_1$ is stored in exactly the same location as in the original tree. Remaining clustering-features in $node_1$ are stored in the leaf node of the new tree containing the first clustering-feature of $node_1$; at this point the new tree has only one leaf node. We step through the leaf node to check if the clustering-feature can be absorbed in any of the clustering-features currently stored, without the size of the resulting clustering-feature becoming greater than the incremented threshold $T + \Delta T$. If the absorption is unsuccessful, the clustering-feature is accommodated in the leaf node of the new tree.

Nodes in the current tree are deleted, when they are no longer required for the insertion of the remaining clustering-features. During the rebuild procedure at any point there are at most only h extra nodes compared to the total number of nodes in the current tree, where h is the height of the current tree.

Figure 2.5 illustrates the tree rebuild procedure. Figure 2.5 (a) shows the current **CF**-tree, at the beginning of the rebuild procedure. Figure 2.5 (b) shows the insertion for the first clustering-feature of $node_1$ in the new tree and Figure 2.5 (c) shows the insertion of remaining clustering-features of $node_1$. Figure 2.5 (d)–(f) show the insertion of the clustering-features of $node_2$. We find the closest leaf node to $\mathbf{CF}_4$ by traversing the tree shown in Figure 2.5 (c). Figure 2.5 (d) shows that $\mathbf{CF}_4$ could be absorbed by a clustering-feature in the closest leaf node. We find the closest leaf node to $\mathbf{CF}_5$ by traversing the tree shown in Figure 2.5 (d). $\mathbf{CF}_5$ could not be absorbed in any clustering-feature stored in the closest leaf node. We accommodated $\mathbf{CF}_5$ in a new leaf node as shown in Figure 2.5 (e); note that the path of the new leaf node is exactly same as the path of $node_2$ in the original tree. Figure 2.5 (f) shows that the closest leaf node to $\mathbf{CF}_6$ in the new tree is the second leaf node. $\mathbf{CF}_6$ was successfully absorbed by a clustering-feature in the closest leaf node. Remaining clustering-features are inserted

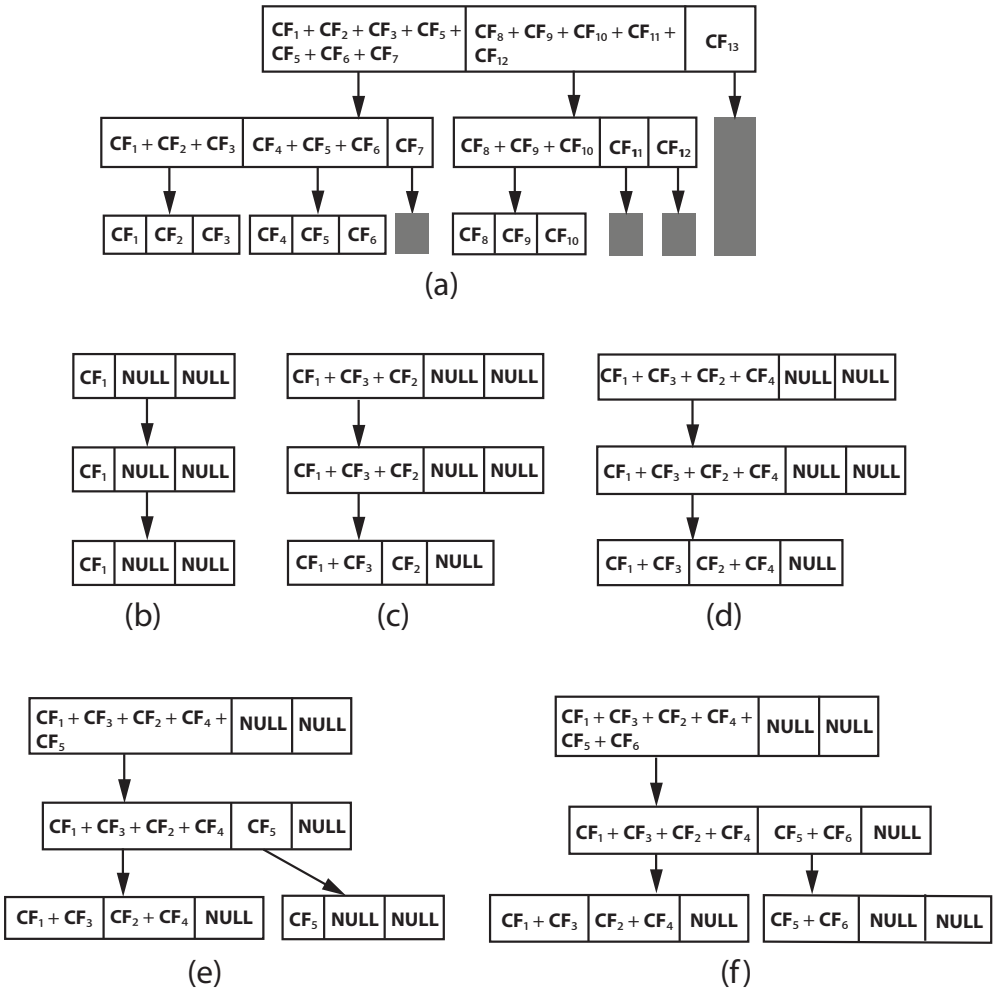


Figure 2.5: (a): Current **CF**-tree at the beginning of the rebuild procedure. (b): Insertion of the first clustering-feature of $node_1$. (c): Insertion of remaining clustering-features of $node_1$. (d, e, f): Insertion of clustering-features of $node_2$.

in a similar way.

Detailed complexity analysis of BIRCH clustering can be found in [82]. The cost of inserting the entire dataset in the tree is $O\big(H\big(1+\log\frac{M_o}{P}\big)\big)$, where H is the total number of points in the dataset, M_o is the total memory size of the entire dataset, and P is the total memory of each node of the CF-tree. The cost of reach rebuild operation is $O\big(\frac{M_o}{M_{\mathbf{cf}}}\big(1+\log\big(\frac{M_o}{P}\big)\big)\big)$, where $M_{\mathbf{cf}}$ is the size of each clustering-feature. The stated complexities are strong upper bounds.

2.2 m-BIRCH Algorithm

The m-BIRCH algorithm [40] automatically determines the incremented threshold value, T_i, for the i^{th} rebuild step, and the initial threshold, T_0, with which the **CF**-tree is initially built. The proposed method approximates the computationally infeasible ideal approach for determining the threshold value. The m-BIRCH algorithm effectively handles regions of varying density, and ensures that regions of different density contribute in generating the final cluster centers.

2.2.1 Automatic Threshold Selection

Suppose during the i^{th} rebuild step we want to increase the threshold so that the closest C clustering-features are merged. Ideally the incremented threshold value can be determined in the following way,

1. Calculate the distances between all clustering-features.
2. Merge the closest two clustering-features, and calculate the distance of the resulting merged clustering-feature with all the other clustering-features.
3. If C clustering-features have been merged then set the incremented threshold value to be the size of the clustering-feature obtained by the last merge, otherwise repeat step 2.

However, the ideal method is computationally infeasible, because it requires the calculation and processing of a large number of distances. If the **CF**-tree consists of M clustering-features at the beginning of the rebuild stage, step 1 would require calculating approximately $\frac{M^2}{2}$ distances and step 2 would require calculating at least $\frac{CM}{2}$ distances. We approximate the ideal method with a computationally feasible approach.

During step 2, we are interested in merging the closest clustering-features. BIRCH places clustering-features close to each other in the same leaf node. Instead of calculating the distance of each clustering-feature with every other clustering-feature in the **CF**-tree, we only calculate the distances between clustering-features in same leaf node. The modification reduces the number of distances calculated in step 1 from $\frac{M^2}{2}$ to at

most $\frac{M(B-1)}{2}$, where $B \ll M$. During step 2 the distance of each merged clustering-feature is calculated only with other clustering-features from the same leaf node. We process only the closest αC clustering-features, where $\alpha \geq 1$ is a constant. Considering only αC closest clustering-features assumes that any two clustering-features which are not considered cannot be merged with other clustering-features such that they are within the C closest clustering-features. The modification reduces the number of distances calculated in step 2 from at least $\frac{CM}{2}$ to at most $C(B-2)$, where $B \ll M$.

The initial threshold T_0 is similarly determined by merging the closest clustering-features in a **CF**-tree built with zero threshold value. The number of clustering-features merged while determining T_0 is conservatively chosen, because the threshold value is refined during each rebuild step of the data summarization process.

The proposed approach requires additional memory for pointers to the closest αC clustering-features and the distances between them. However, as the dimensionality of the data increases, the memory overhead required to store pointers and distances between two clustering-features becomes negligible compared to the memory required to store the clustering-features themselves in the **CF**-tree. In our experiments, the memory overhead is less than 1%.

2.2.2 Multi-Tree Summarization

When the entire dataset is summarized using a single **CF**-tree the final threshold value is large enough to concisely represent the low density regions of the feature space using clustering-features. However, due to the large final threshold value each clustering-feature placed in dense regions represents very large number of points. The final summary is such that few clustering-features represent all the points in the entire dense regions and most of the clustering-features represent points in the low density regions. Using the data summarization with global clustering algorithms like K-means result in few cluster centers in the dense regions and most of the centers are placed in the low density regions.

In order to ensure that different density regions are well represented in the final

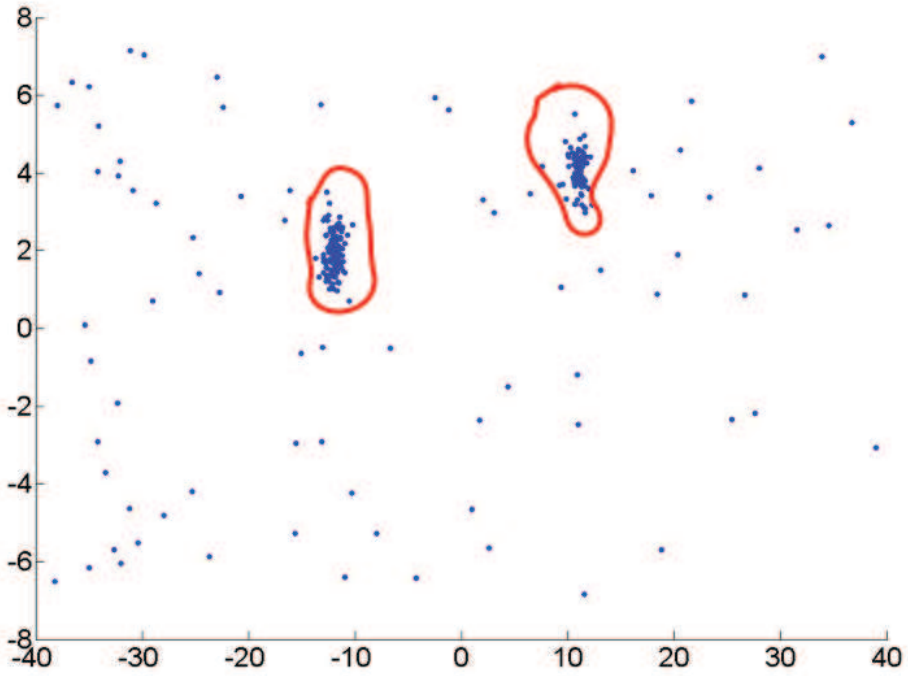


Figure 2.6: The dataset has two clusters consisting of points inside the two curves. The remaining points do not belong to any cluster and are called as outliers.

summary we build multiple **CF**-trees. Each **CF**-tree summarizes different density regions in the feature space. The first **CF**-tree is built such that the dense regions are summarized by the clustering-features and points in the remaining regions are written out as outliers, the next **CF**-tree is built using the outliers of the first tree such that regions with relatively less density are summarized, and so on.

2.3 Outlier Elimination

Outliers are points in the dataset which do not belong to any cluster. For example, consider the dataset of points shown in Figure 2.6. The dataset has two clusters consisting of points inside the two curves. The remaining points do not belong to any cluster and are called outliers. In order to identify correct clusters in presence of outliers a clustering algorithm should be able to detect and remove the outlier points. Each outlier does not have many points in it's neighborhood; therefore each outlier belongs to a clustering-feature representing only few points. During each rebuild step the leaf

clustering-features representing less than N_{out} points are written out as potential outliers. Only the clustering-features representing greater than or equal to N_{out} number of points are inserted in the new tree. During the next rebuild step the list of potential outliers is scanned to see if then can be inserted in any leaf clustering-feature. Arrival of new data and the increase in threshold generates the possibility that density in the region of the points represented by a potential outlier clustering-feature has increased. Once all the points have been inserted, the set of clustering-features remaining in list are determined to be outliers.

We discussed the original BIRCH and the m-BIRCH algorithms. The BIRCH algorithm concisely represents the dataset as a collection of clustering-features using a fraction of the dataset memory, and obtains the final clusters by adapting a global clustering algorithm. The maximum allowed size of clustering-features limits the maximum allowed average inter-point distance between points represented by the same clustering-feature. The m-BIRCH algorithm automatically determines the maximum allowed size by approximating the ideal method using a computationally feasible approach, and uses different maximum sizes for regions of different density. Working with clustering-features significantly reduces the number of entities and memory required to cluster datasets.

Chapter 3

Global Clustering on Clustering-features

We obtain final clusters by adapting GMM, K-means, and spectral clustering algorithms to work on clustering-features. This is the first time that GMM and spectral clustering have been adapted to cluster clustering-features.

3.1 GMM on Clustering-features

GMM models each cluster as a Gaussian distribution. Let $\mathcal{N}(\boldsymbol{\mu}_k, \sigma_k^2\mathbf{I})$ denote the Gaussian distribution of the k^{th} cluster and a_k the corresponding prior probability. GMM [24] estimates the mixture parameters, $\boldsymbol{\mu}_k$, σ_k^2, and a_k, using the expectation maximization (EM) algorithm [51]. Each iteration of EM maximizes expected log likelihood given the observations and the current estimate of the parameters,

$$Q(\boldsymbol{\theta}'|\boldsymbol{\theta}) = E\left[\log\left(f\left(\mathbf{X}_1 \ldots \mathbf{X}_H|\boldsymbol{\theta}'\right)\right)|\mathbf{x}_1 \ldots \mathbf{x}_H, \boldsymbol{\theta}\right], \tag{3.1}$$

where $\mathbf{X}_i$ is the random variable used to model the i^{th} observation, $\mathbf{x}_i$ is the i^{th} observation, H is the number of points, $\boldsymbol{\theta}'$ is the vector of parameters, $\boldsymbol{\theta}$ is the vector of current estimates of the parameters, and $f\left(\mathbf{X}_1 \ldots \mathbf{X}_H|\boldsymbol{\theta}\right)$ is the joint probability density function.

Without loss of generality, suppose the points are arranged in the order of the clustering-features used to represent them. We assume the points represented by the same clustering-feature belong to the same cluster. The objective function maximized

during each iteration gets modified as,

$$\begin{aligned} Q(\boldsymbol{\theta}'|\boldsymbol{\theta}) = E[\log\left(f\left(\mathbf{X}_1 \ldots \mathbf{X}_H|\boldsymbol{\theta}'\right)\right)| \\ \mathbf{x}_1 \ldots \mathbf{x}_H, \boldsymbol{\theta}, \\ K_1 = K_2 = \ldots = K_{N_1}, \\ K_{N_1+1} = K_{N_1+2} = \ldots = K_{N_1+N_2}, \ldots], \end{aligned} \tag{3.2}$$

where $N_1, N_2, \ldots$ are the number of points represented by the clustering-features and $K_1, K_2, \ldots$ are the cluster identities of the points.

Maximizing the modified objective function results in the following update equations,

$$a'_k = \frac{\sum_{i=1}^{M}(r_{i,k} \times N_i)}{\sum_{i=1}^{M} N_i}, \tag{3.3}$$

$$\boldsymbol{\mu}'_k = \frac{\sum_{i=1}^{M}(r_{i,k} \times \mathbf{L}_i)}{\sum_{i=1}^{M}(r_{i,k} \times N_i)}, \tag{3.4}$$

$$\sigma_k^{2\prime} = \frac{\sum_{i=1}^{M} r_{i,k}\left(s_i - 2\boldsymbol{\mu}_k{}'^T\mathbf{L}_i + N_i(\boldsymbol{\mu}_k{}'^T\boldsymbol{\mu}_k{}')\right)}{d\sum_{i=1}^{M}(r_{i,k} \times N_i)}, \tag{3.5}$$

where M is the number of clustering-features, d is the data dimension, and $r_{i,k}$ is the posterior probability of the k^{th} component given the i^{th} clustering-feature.

3.2 K-means on Clustering-features

K-means is an iterative distance based clustering algorithm. Suppose x_{ck} denotes the current estimate of the k^{th} cluster. The K-means algorithm determines the updated location, x'_{ck}, of the k^{th} cluster by minimizing the equation,

$$F = \sum_{k=1}^{K}\sum_{i=1}^{H} b_{i,k}\|\mathbf{x}_i - \mathbf{x}'_{ck}\|_2^2, \tag{3.6}$$

where K is the number of clusters, and $b_{i,k}$ is the indicator variable taking the value one or zero depending on whether the point $\mathbf{x}_i$ is closest to x_{ck}, the current estimate of the k^{th} cluster.

Without loss of generality, suppose the points are arranged in the order of the clustering-features used to represent them. K-means can be adapted to work on clustering-features by constraining $b_{i,k}$ to be one only if the clustering-feature representing the set

containing the i^{th} point is closest to x_{ck}. The objective function in equation 3.6 gets modified as,

$$F = \sum_{k=1}^{K} e_{1,k} \sum_{i=1}^{N_1} \|\mathbf{x}_i - \mathbf{x}'_{ck}\|_2^2 + \sum_{k=1}^{K} e_{2,k} \sum_{i=N_1+1}^{N_1+N_2} \|\mathbf{x}_i - \mathbf{x}'_{ck}\|_2^2 + \ldots, \tag{3.7}$$

where $N_1, N_2, \ldots$ are the number of points represented by the clustering-features, and $e_{1,k} = b_{1,k} = b_{2,k} = \ldots = b_{N_1,k}$, $e_{2,k} = b_{N_1+1,k} = b_{N_1+2,k} = \ldots = b_{N_1+N_2,k}, \ldots$ are the values of the indicator variables of points represented by the same clustering-feature. The modified objective function in equation 3.7 is minimized by,

$$\begin{aligned} \mathbf{x}'_{ck} &= \frac{e_{1,k} \sum_{i=1}^{N_1} \mathbf{x}_i + e_{2,k} \sum_{i=N_1+1}^{N_1+N_2} \mathbf{x}_i + \ldots}{N_1 e_{1,k} + N_2 e_{2,k} + \ldots} \\ &= \frac{e_{1,k}\mathbf{L}_1 + e_{2,k}\mathbf{L}_2 + \ldots}{N_1 e_{1,k} + N_2 e_{2,k} + \ldots}, \end{aligned} \tag{3.8}$$

where $\mathbf{L}_1, \mathbf{L}_2 \ldots$ are the linear sum component of the clustering-features. During each iteration $e_{i,k}$ is one only if the i^{th} clustering-feature is closest to the current estimate of the k^{th} cluster center, $\mathbf{x}_{ck}$. Equation 3.8 reduces to,

$$\mathbf{x}'_{ck} = \frac{\sum_{i \in I_k} \mathbf{L}_i}{\sum_{i \in I_k} N_i}, \tag{3.9}$$

where the index i belongs to set I_k if the i^{th} clustering-feature is closest to the k^{th} center.

3.3 Spectral Clustering on Clustering-features

We use the spectral clustering algorithm proposed in [54]. The algorithm clusters a set of points $P = \{\mathbf{p}_1 \ldots \mathbf{p}_H\}$ to K clusters in the following way,

1. Let $\mathbf{A} \in \Re^{H \times H}$ be the affinity matrix defined by $A_{i,j} = e^{-\frac{\|\mathbf{p}_i - \mathbf{p}_j\|_2^2}{2\sigma^2}}$ if $i \neq j$ and $A_{i,i} = 0$.

2. Let $\mathbf{D}$ be a diagonal matrix defined by $D_{i,i} = \sum_{j=1}^{H} A_{i,j}$.

3. The matrix $\mathbf{L}$ is defined by $\mathbf{L} = \mathbf{D}^{\frac{1}{2}}\mathbf{A}\mathbf{D}^{\frac{1}{2}}$.

4. Let $\mathbf{v}_1 \ldots \mathbf{v}_k$ be the k largest eignevectors of $\mathbf{L}$, and $\mathbf{V} = [\mathbf{v}_1, \ldots, \mathbf{v}_k]$ be the matrix consisting of the eigenvectors as columns.

5. The matrix **W** is defined by normalizing the rows of **V** to unity.

6. The rows of **W** are clustered to K clusters using K-means.

7. The i^{th} original point is assigned to the same cluster as the corresponding row of matrix **W**.

We cluster the clustering-feature summarization instead of individual points. When spectral clustering is used on clustering-features, the non-zero element , $A_{i,j}$, at row i and column j of the affinity matrix is given by,

$$A_{i,j} = e^{-\frac{d(S_i,S_j)^2}{2\sigma^2}}, \tag{3.10}$$

where σ^2 is the scaling parameter, S_i and S_j are the set of points represented by the i^{th} and j^{th} clustering-features and $d(S_i, S_j)$ is the distance between them as defined in equation 2.2.

Using spectral clustering on the clustering-feature summarization significantly reduces the size of the affinity matrix. If the ratio of the number of data points to the number of clustering-features is r, the size of the affinity matrix reduces by a factor of r^2. For example, if the dataset consists of 1000 points and m-BIRCH summarizes the dataset using 100 clustering-features, the size of the affinity matrix reduces from 1000×1000 to 100×100.

Chapter 4

Clustering Results

We use m-BIRCH to cluster large datasets of features commonly used in computer vision: color SIFT descriptors [39] on regular grid intervals, color patches, grayscale SIFT descriptors on affine-invariant Hessian regions [50], outlier corrupted and PCA projected grayscale patches, and outlier corrupted non-convex datasets. We compare the performance of m-BIRCH with sequential GMM online clustering algorithm.

The m-BIRCH algorithm required only 10 to 20% of the dataset memory to perform the clustering. Table 4.1 shows the memory required by m-BIRCH for each dataset.

We evaluate the vocabularies by classifying separate test sets using k-nearest neighbor (k-NN) classifier. The classification experiments use normalized histogram representation and the distances between them are measured using the L_1-norm [56]. In each experiment, we evaluate the success rate for one to ten nearest neighbors and reported the best results. We chose K-NN over more powerful classification approaches so that the success rates are due the clustering quality and not due to the discriminative power of the classifier.

We automatically determine the threshold using the approach discussed in section 2.2.1. We set the number of clustering-features merged during each rebuild step to 30% of the maximum number of clustering-features which can be held within the memory constraints and use twice the number of clustering-features for distance evaluations. In case sufficient number of clustering-features are not retained after writing the outliers, we reduce the number of clustering-features by 20%. Note that similar settings are used to determine the thresholds for a very wide variety of features being clustered.

m-BIRCH Memory Requirements

Dataset	Percentage of Dataset Memory Required
840K Color SIFT	10%
1M Color patches	10%
60K Grayscale Patches with Outliers	15%
700K Grayscale SIFT	20%

Table 4.1: The m-BIRCH algorithm incrementally clustered the datasets using just 10% to 20% of the dataset memory.

Figure 4.1: Example images from each scene category in the OT dataset. Top row: coast, highway, open country, and inside city. Bottom row: mountain, street, forest, and tall building.

4.1 840,000 Color SIFT Descriptors

We cluster color SIFT descriptors extracted from the Oliva and Torralba (OT) scene classification dataset [58]. The dataset consists of 2688 256×256 images representing eight scene categories: coast, highway, open country, inside city, mountain, street, forest, and tall building. Figure 4.1 shows an example image from each scene category. The OT dataset is very widely used in computer vision to test the performance of scene classification algorithms [75, 78, 7].

We use half the images from each category for training and half for testing. We represent the images in the HSV format in the range 0 to 1, extract SIFT descriptors from

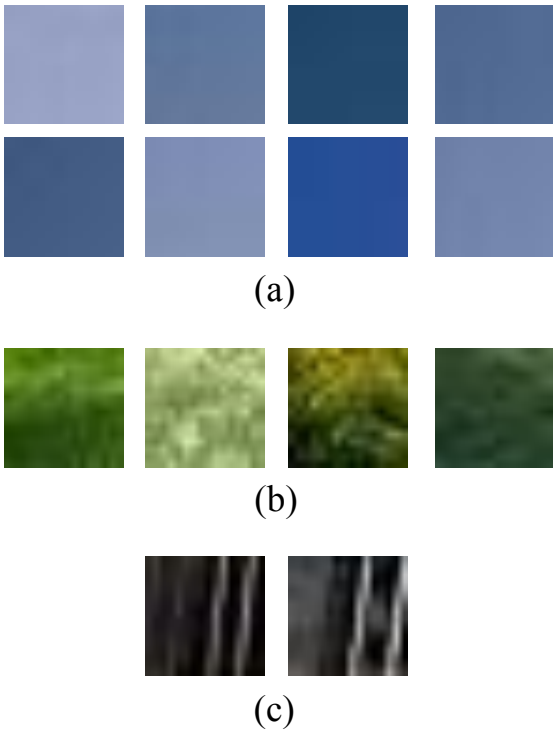


Figure 4.2: Example patches whose color SIFT descriptors have the same closest clustering-feature. Note that patches with the same closest clustering-feature have similar appearance.

each component at regular intervals on a grid, and concatenate SIFT descriptors from the different components to generate 384-dimensional color SIFT descriptor vectors. In our experiments, we use a grid spacing of 10 pixels, and obtain 840,000 descriptors from the training images. The descriptor components are represented using 8 byte double precision.

We use m-BIRCH to represent the entire training set using 74,449 clustering-features, which require just 9% of the training set memory. We obtain the condensed representation by building three **CF**-trees in succession. Each tree is built within 10% of the training set memory, has a branch factor of 20, and N_{out} is set to 16, 4, and 2 respectively. Figure 4.2 (a), (b), and (c) show example patches whose color SIFT descriptors have the same closest clustering-feature. Note that patches with the same closest clustering-feature are similar in appearance.

We cluster the clustering-feature representation of the training set to 1000 centers using spectral, K-means, and GMM based clustering. We use the generated vocabulary

Color SIFT Based Vocabularies

Clustering Method	Classification Accuracy
Batch K-means	74.18
m-BIRCH + Spectral	78.01
m-BIRCH + K-means	75.91
m-BIRCH + GMM	77.27

Table 4.2: Classification result of batch K-means vocabulary establishes a baseline for measuring classification accuracy. The dataset is too large to be clustered using batch spectral and batch GMM algorithms. The baseline shows that m-BIRCH based vocabularies generated incrementally using just 10% of the dataset memory have high discriminative power.

to obtain normalized histogram representation of the images in the dataset, and k-NN classification. Table 4.2 summarizes the overall classification results obtained by averaging the classification success rate of the different categories, and Table 4.3 shows the confusion matrix for the m-BIRCH + Spectral vocabulary. The first row of table 4.2 shows the classification result for the batch K-means algorithm, which serves as a baseline for measuring the classification accuracy of m-BIRCH based vocabularies. The dataset is too large to be clustered using batch spectral and batch GMM algorithms. The baseline shows that m-BIRCH vocabularies incrementally generated using 10% of the dataset memory have high discriminative power.

4.2 1,093,680 Color Patches

We cluster 6×6 color patches extracted from four scene categories of the OT dataset: coast, highway, forest, and street. Images were converted to HSV format in the range 0 to 1. The patches were represented as 108-dimensional vectors obtained by vectorizing the intensities of each component and concatenating the different components. We use half the images for training and half for testing. The training set generates 1,093,680 patches.

We use m-BIRCH to represent the training set using 38,121 clustering-features by building four **CF**-trees within 10% of the training set memory. Each tree has a branch

Confusion matrix for m-BIRCH + Spectral clustering on color SIFT

coast	**73.89**	9.44	13.33	0	1.67	0.56	1.11	0
highway	5.39	**83.85**	3.08	0.77	3.08	2.31	1.54	0
open country	6.34	5.85	**59.02**	0	6.34	0	21.95	0.49
inside city	2.60	1.95	1.30	**83.12**	0	7.80	2.60	0.65
mountain	4.28	1.07	3.21	0.54	**74.33**	0.54	16.04	0
street	0	2.06	0.69	10.27	0	**84.25**	1.37	1.37
forest	0	0	0.61	0	0.61	0	**98.78**	0
tall building	1.12	3.93	0	10.67	6.74	3.93	6.74	**66.85**

Table 4.3: Confusion matrix for spectral clustering applied to m-BIRCH generated data summarization of the 840,000 point dataset. Note that the vocabulary obtained by using m-BIRCH + spectral clustering has high discriminative power.

factor of 20, and N_{out} is set to 20, 10, 4, and 2 respectively. We cluster the clustering-features to 1000 centers using spectral, K-means, and GMM based clustering. Table 4.4 shows the classification performance on the test set using normalized histogram representation of the images and k-NN classifier. The classification result for batch K-means vocabulary establishes a baseline for measuring the classification accuracy. The dataset is too large to be clustered using batch spectral and batch GMM algorithms. The baseline accuracy shows that m-BIRCH based vocabularies incrementally generated using 10% of the dataset memory have high discriminative power.

4.3 60,000 Outlier Corrupted PCA Projected Grayscale Patches

We cluster 50-dimensional PCA projected grayscale patches obtained from the MNIST dataset [37]. The dataset enables us to visually verify, in addition to quantitative classification based assessment, the capability of m-BIRCH to cluster outlier corrupted datasets.

Color Patches Based Vocabularies

Clustering Method	Classification Accuracy
Batch K-means	85.81
m-BIRCH + Spectral	86.61
m-BIRCH + K-means	85.97
m-BIRCH + GMM	87.42

Table 4.4: Classification result of batch K-means vocabulary establishes a baseline for measuring classification accuracy. The dataset is too large to be clustered using batch spectral and batch GMM algorithms. The baseline shows that m-BIRCH based vocabularies generated incrementally using just 10% of the dataset memory have high discriminative power.

The MNIST dataset consists of 60,000 28×28 dimensional grayscale images of handwritten digits between zero and nine. Figure 4.3 (a) shows example images from the MNIST dataset. Figure 4.3 (b) shows three different instances of digits four and zero; note the significant intra-digit variation of style. We normalize the images to unit length, subtract the mean, and project to 50-dimensional space using principal component analysis (PCA) [24]. We add 25,714 outliers to PCA projected images; thus 30% of the points in the resulting dataset are outliers. The i^{th} component of the outlier points are selected to be uniformly distributed in the range $m_i \pm 20\sigma_i$, where m_i and σ_i are the mean and the standard deviation of the inlier points in the i^{th} dimension. In our experiments, m_i is zero for all dimensions, because we subtract the mean before projection. Figure 4.3 (c) shows example images of the outliers. The components of the 50-dimensional vectors were represented using 8 byte double precision.

We use m-BIRCH to represent the entire dataset using 2272 clustering-features. The condensation is obtained by building one **CF**-tree within 15% of the dataset memory, and having a branch factor 10 and outlier threshold 3. Figure 4.4 (a), (b), and (c) shows example images which have the same closest clustering-feature. Images with the same closest clustering-feature are similar in appearance. The clustering-features represented 59,941 points, which is almost equal to the number of inlier points, 60,000, in the dataset.

We cluster the clustering-features to 100 cluster centers using spectral, K-means, and

(a)

(b)

(c)

Figure 4.3: (a): 30 handwritten digits from the MNIST dataset. (b): Three instances of digits four and zero. Note the significant intra-digit variation of style. (c): Sample outlier points added to the dataset.

GMM based clustering. We chose 100 cluster centers to capture the intra-digit variation of style. Figure 4.5 (a), (b), and (c) show the cluster centers obtained by condensing the dataset using m-BIRCH followed by applying spectral, K-means, and GMM based clustering respectively , and Figure 4.5 (d) shows the cluster centers obtained by directly applying K-means to the dataset. Note that m-BIRCH based vocabularies effectively handled the outliers and generated clean cluster centers. However, direct application of K-means clustering resulted in incorrect cluster centers due to outliers.

We classify a separate test set consisting of 10,000 images using nearest neighbor

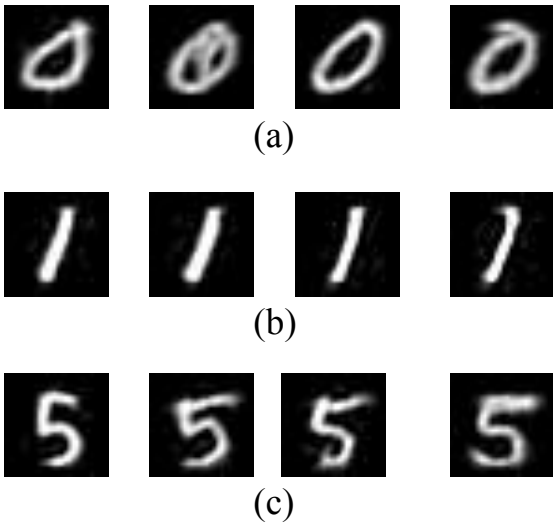


Figure 4.4: Example digit images which have the same closest clustering-feature. Note that digits with the same closest clustering-feature have similar appearance.

classification with L_2-norm. We chose nearest neighbor over more powerful classification approaches so that the success rates are due the clustering quality and not due to the discriminative power of the classifier. Table 4.5 summarizes the classification results for: 1) m-BIRCH based vocabularies generated with 15% memory using the outlier corrupted MNIST dataset, 2) m-BIRCH based vocabularies generated with 10% memory using MNIST dataset without outliers, and 3) directly applying the batch clustering algorithms to the MNIST dataset without outliers. Note that the classification accuracy of m-BIRCH based vocabularies incrementally generated using only fraction of the dataset memory is close to the classification accuracy of vocabularies generated using batch clustering algorithms, which require much higher memory.

4.4 707,877 Grayscale SIFT

We cluster 128-dimensional grayscale SIFT descriptors evaluated on affine-invariant Hessian regions extracted from the Oxford buildings dataset [59]. The Oxford buildings dataset is widely used in computer vision to assess the performance of content-based object retrieval algorithms [30, 73, 29].

The dataset consists of 11 landmarks, and the images of each landmark are divided into three types: *Good* if the landmark is clearly visible, *Ok* if more than 25% of

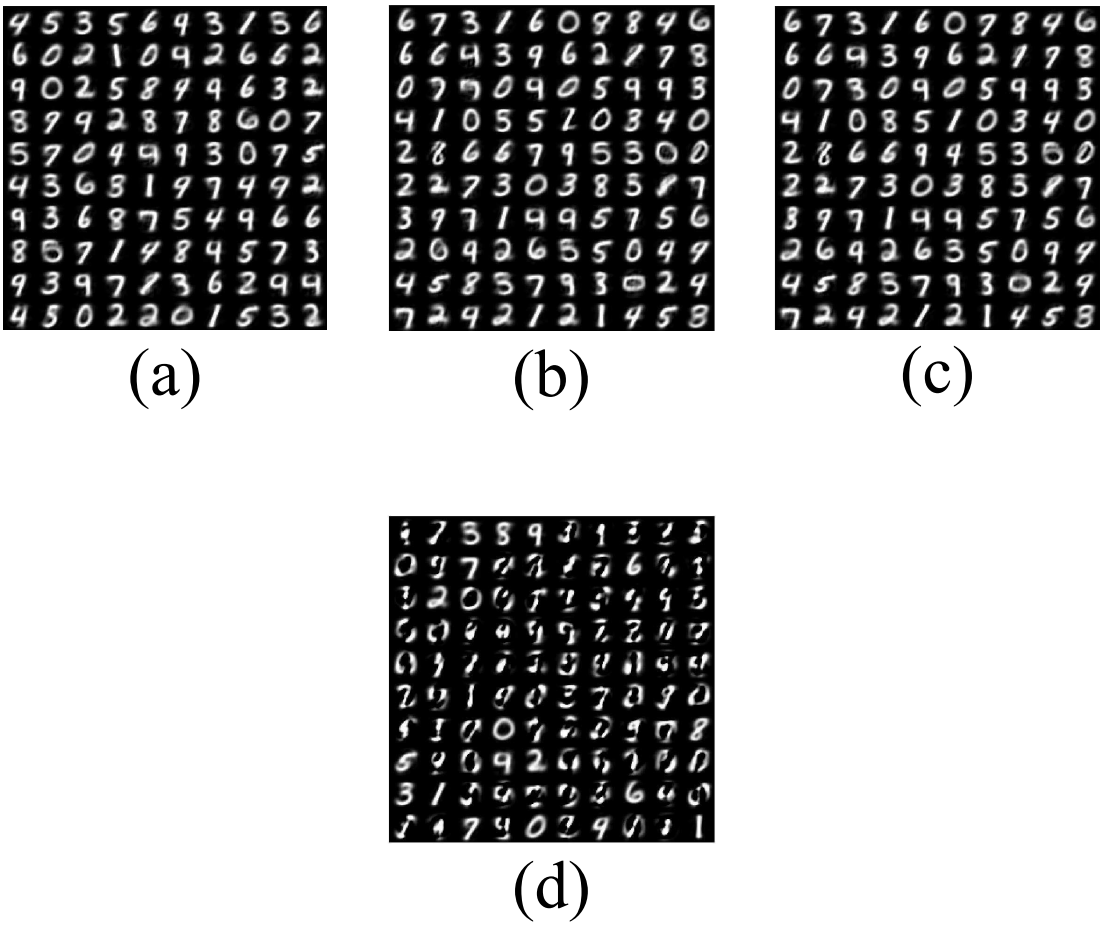


Figure 4.5: (a,b,c): Cluster centers obtained by condensing the dataset using m-BIRCH followed by spectral, K-means, and GMM based clustering respectively. (d): Cluster centers obtained by directly applying K-means to the dataset. Note that m-BIRCH based vocabularies generated clean cluster centers. However, directly applying K-means to the dataset generated incorrect cluster centers due to outliers.

the landmark is visible, and *Junk* if less than 25% of the landmark is visible. In our experiments, we consider the *Good* images and categories with more than 10 *Good* images, which generates a training set of 198 images from four landmarks. Figure 4.6 shows an example image of each landmark. The training set consists of 707,877 128-dimensional grayscale SIFT descriptors. Each element of the descriptor vector was represented using one byte unsigned char datatype.

We use m-BIRCH to concisely represent the entire dataset using 28001 clustering-features. We obtain the condensed representation by building three **CF**-trees in succession. Each tree is built within 20% of the dataset memory, has branch factor of 20, and

Classification Results on the MNIST Dataset for m-BIRCH (top) and Batch (bottom) Clustering Algorithms

Clustering method	MNIST + outliers	MNIST without outliers
m-BIRCH + Spectral	84.39	86.91
m-BIRCH + K-means	82.67	86.33
m-BIRCH + GMM	81.57	86.11

Clustering method	MNIST without outliers
Batch Spectral	91.68
Batch K-means	89.3
Batch GMM	87.96

Table 4.5: Classification accuracy of m-BIRCH based vocabularies generated using only fraction of the dataset memory is close to the classification accuracy of vocabularies generated using batch clustering algorithms.

Figure 4.6: Example images from the Oxford buildings dataset.

N_{out} is set to 20, 10, and 4. We cluster the clustering-features to 1000 cluster centers using spectral, K-means, and GMM based clustering.

We evaluate the clusters by retrieving eight query images, two per category, given with the dataset. Figure 4.7 shows the query images. We represent the images as normalized histograms and measure performance using mean average precision (mAP) [14]. The average precision of each query image is defined as the mean value of precision for all recall values. The average precision of each category is defined as the mean of average precision of all query images belonging to the category. The mAP is defined as the mean of the average precision of every category. Table 4.6 shows the mAP obtained for batch K-means and m-BIRCH based vocabularies. Table 4.7 shows the average precision of each category for batch K-means and m-BIRCH based vocabularies. The first two queries correspond to the first category, the next two correspond to the

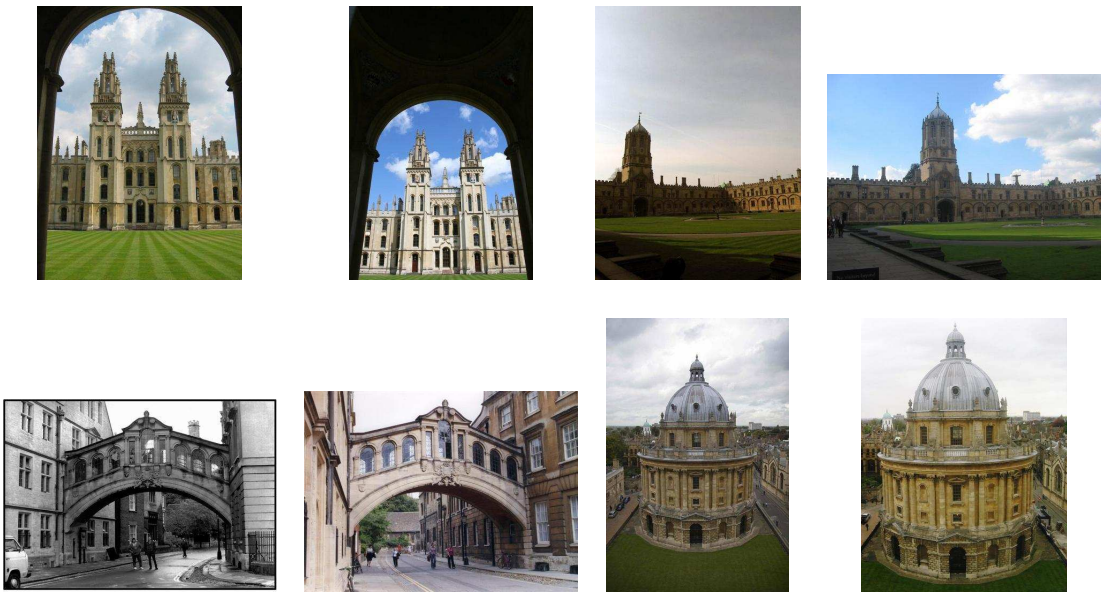


Figure 4.7: Query images to test the retrieval performance.

second category, and so on. The classification result for batch K-means vocabulary establishes a baseline for measuring the classification accuracy. The dataset is too large to be clustered using batch spectral and batch GMM algorithms. The baseline accuracy shows that m-BIRCH based vocabularies incrementally generated using 20% of the dataset memory have high discriminative power.

4.5 Non-Convex Clustering

We cluster non-convex datasets using m-BIRCH with spectral clustering. Algorithms like K-means and GMM are known to perform poorly on datasets containing non-convex clusters. We use m-BIRCH with spectral clustering to discover non-convex clusters in the problem of 3-D motion segmentation. We also use m-BIRCH with spectral clustering to discover non-convex clusters in patterns of known structure, where the non-convexity of the cluster can be easily visualized.

4.5.1 3-D Motion Segmentation

We cluster features extracted from sequences of the Hopkins 155 3-D motion segmentation dataset [72]. Figure 4.8 shows example images from sequences in the dataset.

mAP for Categories from Oxford Building Dataset

Clustering Method	mAP
Batch K-means	77.80
m-BIRCH + Spectral	80.43
m-BIRCH + K-means	80.36
m-BIRCH + GMM	82.00

Table 4.6: Classification result of batch K-means vocabulary establishes a baseline for measuring classification accuracy. The dataset is too large to be clustered using batch spectral and batch GMM algorithms. The baseline shows that m-BIRCH based vocabularies generated incrementally using just 20% of the dataset memory have high discriminative power.

The sequences consist of objects undergoing different types of motion. For each sequence, the dataset provides pixel location of points detected and tracked across the entire sequence. Features are obtained by concatenating the pixel location of points in all images of the sequence. If the sequence has H points tracked across F frames, the dataset consists of H features in $2F$ dimensional space. The sequences in the Hopkins 155 dataset have 192 to 556 points and 25 to 61 frames; thus the sequences consist of 192 to 556 features in 50 to 122 dimensional space. Motion segmentation algorithms cluster the features according to the type of motion they are undergoing. The clustering problem is non-convex. In motion segmentation literature, the non-convex clustering is often done using spectral clustering [36, 80].

Features belonging to the same cluster lie in a four dimensional linear subspace [80, 36, 72] under the affine projection model. Let $\{\mathbf{p}_{f,i}\}_{f=1,\ldots F}^{i=1,\ldots,G}$ be the pixel locations of the points lying on the same rigidly moving object and $\{\mathbf{P}_i\}$ the corresponding world points. The points lying on the same object undergo identical motion and belong to the same cluster. The projection from world to pixel location is given by,

$$\begin{bmatrix} \mathbf{p}_{1,1} & \cdots & \mathbf{p}_{1,G} \\ \vdots & \vdots & \vdots \\ \mathbf{p}_{F,1} & \cdots & \mathbf{p}_{F,G} \end{bmatrix} = \begin{bmatrix} \mathbf{C}_1 \\ \vdots \\ \mathbf{C}_F \end{bmatrix} \begin{bmatrix} \mathbf{P}_1 & \ldots & \mathbf{P}_G \end{bmatrix}, \tag{4.1}$$

where $\mathbf{C}_f$ is the 2×4 affine projection matrix for frame f. The columns of the matrix on the left hand side of equation 4.1 represent the features obtained from the points

Query Average Precision: Two queries were used per category

Query	Batch K-means	m-BIRCH + Spectral	m-BIRCH + K-means	m-BIRCH + GMM
Query 1	80.95	83.81	73.91	77.04
Query 2	80.37	77.96	81.50	81.04
Query 3	76.94	94.05	71.69	80.94
Query 4	39.63	54.62	57.52	63.83
Query 5	81.91	89.36	90.15	88.75
Query 6	78.63	80.03	81.91	80.17
Query 7	91.40	92.02	92.47	91.52
Query 8	92.57	91.58	93.73	92.68

Table 4.7: Classification accuracy for batch K-means and m-BIRCH based vocabularies on query images supplied with the dataset. The success rates show that m-BIRCH based vocabularies generated using just 20% of the dataset memory have high discriminative power.

lying on the object. The matrix rank is at most four, because the first matrix on the right hand side of equation 4.1 has only four independent columns.

The batch algorithm [80] projects the features to a lower dimensional $4n + 1$ space [36], where n is the number of clusters. The projected features are normalized to unit length. The batch algorithm fits a four dimensional subspace at each normalized point using the point and its four nearest neighbors. The value, $A_{i,j}$, of the affinity matrix at row i and column j is defined as the distance between the subspace at the i^{th} and j^{th} points,

$$A_{i,j} = e^{-\sum_{k=1}^{4} \sin^2(\gamma_k)}, \tag{4.2}$$

where γ_k are the principal angles between the two subspaces.

We use m-BIRCH to concisely represent the features within 15% of the dataset memory. We perform spectral clustering on clustering-features representing at least four features. The distance between two clustering-features is defined as the distance between the corresponding four dimensional linear subspaces. Clustering-features representing less than four features are associated with subspaces closest to their centroid. In our experiments less than 5% of the features are represented by the sparse clustering-features.

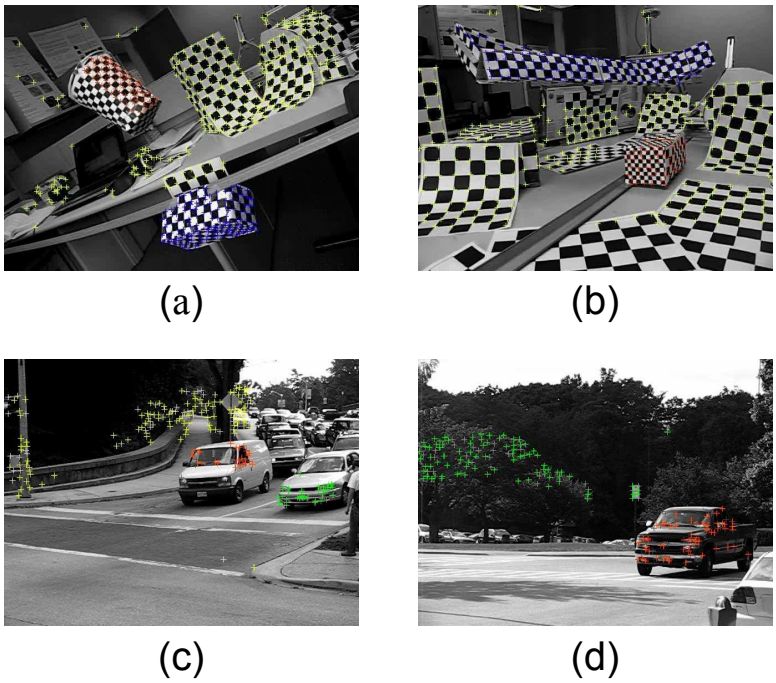


Figure 4.8: Example images from sequences in the Hopkins 155 3-D motion segmentation dataset.

Clustering error is measured as the ratio of the number of incorrectly clustered features to the total number of features in the dataset. The ground truth is provided with the Hopkins 155 dataset. The ground truth is obtained by manually segmenting the features according to the type of motion they are undergoing. Table 4.8 shows the clustering error for different sequences in the Hopkins 155 dataset. The sequence names in Table 4.8 are the same as those in the dataset. The error rates show that m-BIRCH generated summaries were successfully used to discover non-convex clusters.

4.5.2 Outlier Corrupted Data Patterns

Figure 4.9 (a) shows datasets consisting of non-convex clusters. The datasets are similar to those used in [54] for demonstrating the ability of spectral clustering to identify non-convex clusters. In our experiments we make the datasets challenging by corrupting them with outliers. The datasets consist of 2125, 1300, and 1100 points respectively.

We condense the datasets using one **CF**-tree built within 15% of the dataset memory. Figure 4.9 (b) shows the clustering-feature centroids as big dots on the inlier points.

Clustering Error

Sequence	m-BIRCH + Spectral	Batch Spectral
1R2TCR	0.72	0.54
2RT3RCT_B	1.49	0
2T3RCRTP	2.24	2.85
cars9	1.82	0
cars8	1.04	2.08
cars7	0	6.97

Table 4.8: Clustering error for the batch spectral and m-BIRCH + spectral clustering algorithms run on the Hopkins 155 3-D motion segmentation datasets. The error for m-BIRCH + spectral algorithm, which discovers the non-convex clusters using just 15% of the dataset memory, is comparable to that of batch spectral clustering algorithm.

Note that BIRCH correctly placed all the clustering-features around the inlier points. We cluster the clustering-feature representation of the three datasets to three, two, and two clusters respectively. Clustering-features clustered to the same clusters are shown in Figure 4.9 (b) using the same color. BIRCH correctly discovered the non-convex clusters in the datasets. It is well known and was shown in [54], that applying K-means to the datasets generates incorrect clusters.

4.6 Comparison with Sequential GMM Clustering

We compare the performance of m-BIRCH algorithm with sequential GMM clustering [2]. The comparison experiments show that sequential GMM is unable to handle outliers, m-BIRCH generated vocabularies have higher discriminative power, and m-BIRCH algorithm better handles datasets where subsets of points do not reflect the cluster distribution in the entire dataset.

Figure 4.10 shows the performance of sequential GMM clustering on the MNIST dataset with outliers and without outliers. The sequential GMM algorithm is unable to generate clean clusters in presence of outliers. However, as shown in Figure 4.5 m-BIRCH generates clean clusters on the outlier corrupted MNIST dataset.

Table 4.9 shows a comparison of the classification success rate on the OT dataset using color SIFT descriptors and color patches as features. Note that the discriminative

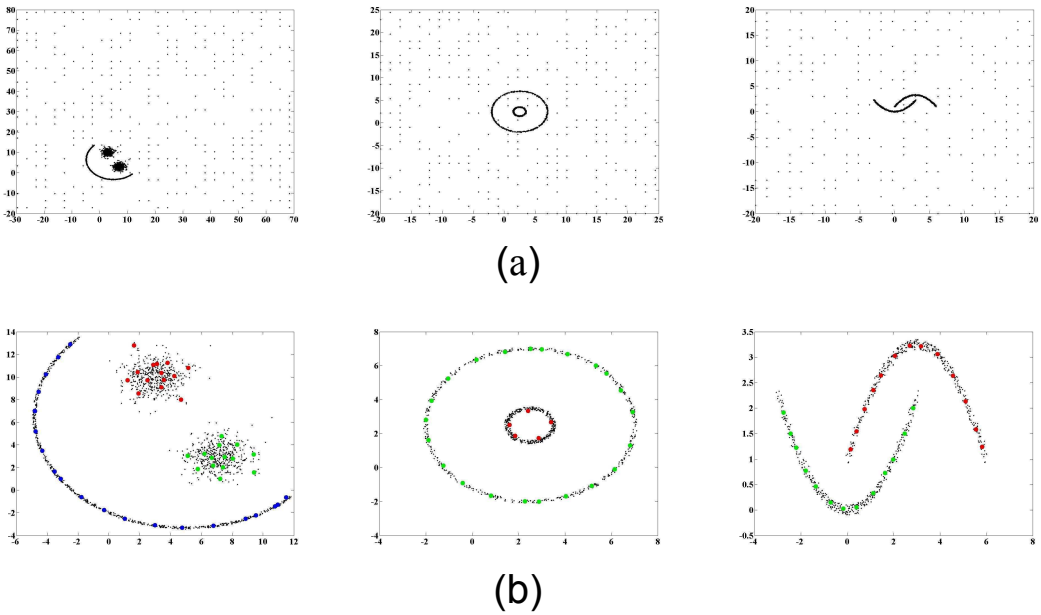


Figure 4.9: *Please view in color.* (a): Outlier corrupted datasets consisting of non-convex clusters. (b): Clustering-feature centroids plotted as big dots on the inlier points. Clustering-features clustered to the same cluster using spectral clustering are shown using identical colors. The clustering results show that BIRCH generated summaries can be used to discover non-convex clusters.

power of m-BIRCH generated vocabulary is higher than that of sequential GMM.

Sequential GMM is unable to discover correct clusters when the subsets of points do not reflect the cluster distribution in the entire dataset. Consider the dataset shown in Figure 4.11 (a), which consists of three clusters. Suppose the dataset is ordered such that points belonging to the middle cluster are in the beginning, followed by points belonging to the right cluster, and the points belonging to the left cluster are at the end. Figure 4.11 (b) shows the cluster centers discovered by sequential GMM. Note that sequential GMM is unable to discover correct cluster centers. The incorrect results are because sequential GMM makes final clustering decisions based on subsets of the data points which do not capture the pattern of the entire dataset. Figure 4.11 (c) shows the cluster centers discovered using m-BIRCH. The m-BIRCH algorithm is able to discover correct cluster centers, because final clustering decisions are made on a concise summary capturing the point distribution in the entire dataset. Another example with the same data-ordering pattern is the MNIST dataset rearranged such that all the zeros are in the beginning, followed by all the ones, followed by all the twos and so on. Figure 4.12

(a) (b)

Figure 4.10: Cluster centers discovered by sequential GMM in the MNIST dataset corrupted with outliers (a) and without outliers (b). Note that in presence of outliers sequential GMM is unable to generate clean clusters.

Performance Comparison of Sequential GMM and m-BIRCH Clustering

Features	Sequential GMM	BIRCH + GMM
Color SIFT	70.83	77.27
Color Patches	84.68	87.42

Table 4.9: Classification accuracy of m-BIRCH + GMM is better than that of sequential GMM clustering algorithm.

(a) shows the incorrect clusters discovered by sequential GMM, and Figure 4.12 (b) shows the clusters discovered by m-BIRCH for the same data-ordering.

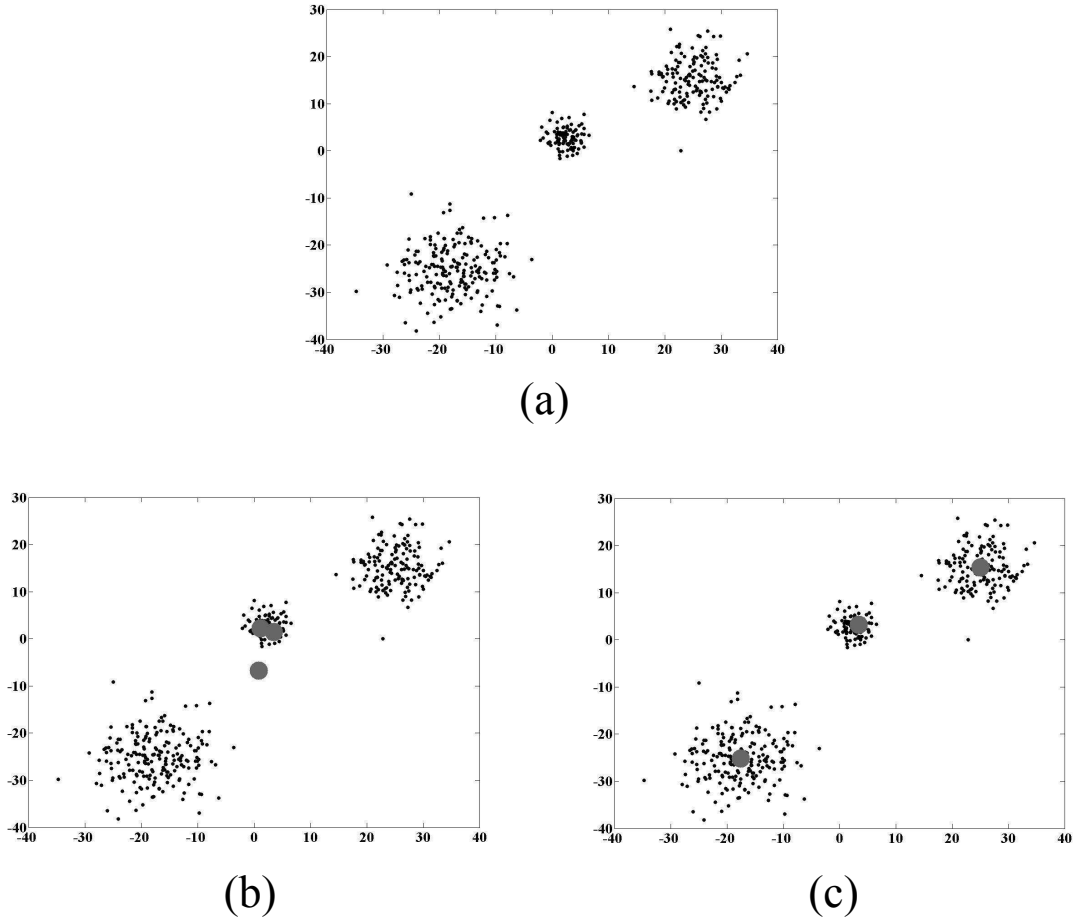


Figure 4.11: (a) Dataset consisting of three clusters. The dataset is ordered such that points belonging to the middle cluster are in the beginning, followed by points belonging to the right cluster, and the points belonging to the left cluster are at the end. (b) Cluster centers obtained using sequential GMM. (c) Cluster centers obtained using m-BIRCH. The cluster centers are shown as grey dots on the data points. Note that sequential GMM incorrectly places all three centers around the middle cluster, whose points are in the beginning of the dataset.

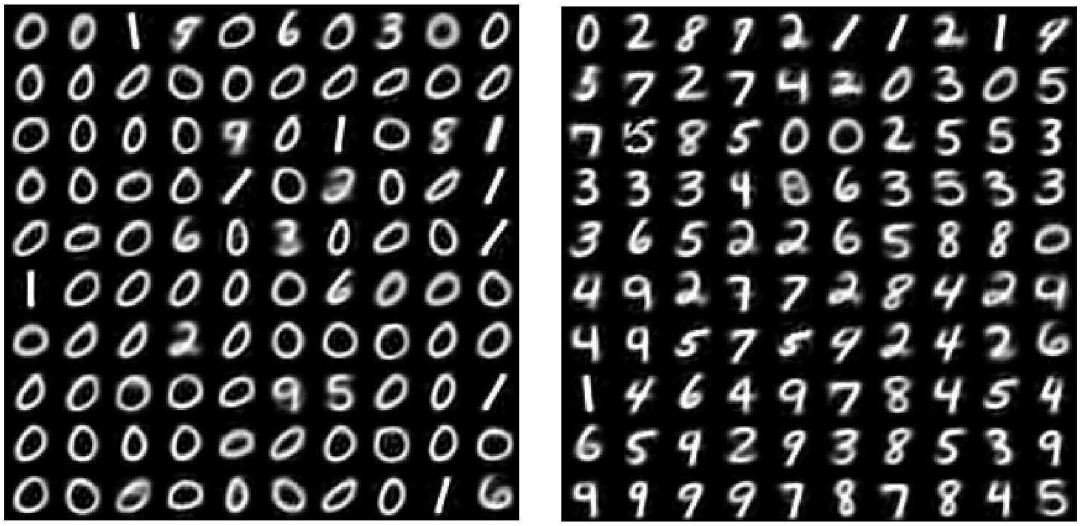


Figure 4.12: (a): Clusters discovered by sequential GMM on the MNIST dataset, when the points are arranged such that all the zeros are in the beginning, followed by all the ones, followed by all the twos, and so on. (b): Clusters discovered by m-BIRCH for the same data-ordering. Sequential GMM is unable to discover the correct clusters.

Chapter 5

Multimodal and Time-lapse Skin Imaging

We discuss the acquisition of multimodal and time-lapse skin images [41, 43] in dermatology clinical studies. We acquire polarized and fluorescence multimodal images in a single imaging session, and time-lapse skin images with acne lesions over a three month period.

5.1 Surface-subsurface Reflectance Model

Figure 5.1 schematically illustrates the multi-layered structure of skin and its interaction with visible light. Skin is a multi-layered medium, with complex fine scale geometry, an oily layer at the air-skin interface, and with layers of different types of cells in the stratum corneum, epidermis, and dermis. Consequently, the way skin reflects visible light is defined by the reflection and inter-reflection of incident light at the interfaces between layers with different indices of refraction. A simplified model of skin reflectance is the surface-subsurface model [53], which models the reflectance as a sum of surface and subsurface components. The surface component is the part of the incident light reflected off the skin surface. The surface component arises due to the oily layer present on normal facial skin. The subsurface component is the part of the incident light traveling through the stratum corneum and the epidermis. The subsurface component suffers multiple subsurface scattering events before it exits the skin.

5.2 Multimodal Skin Images

We acquire skin images under five different modalities, three in reflectance mode (visible, parallel-polarized, cross-polarized) and two in fluorescence mode (UVA and blue light excitation). The different modalities capture different information about skin.

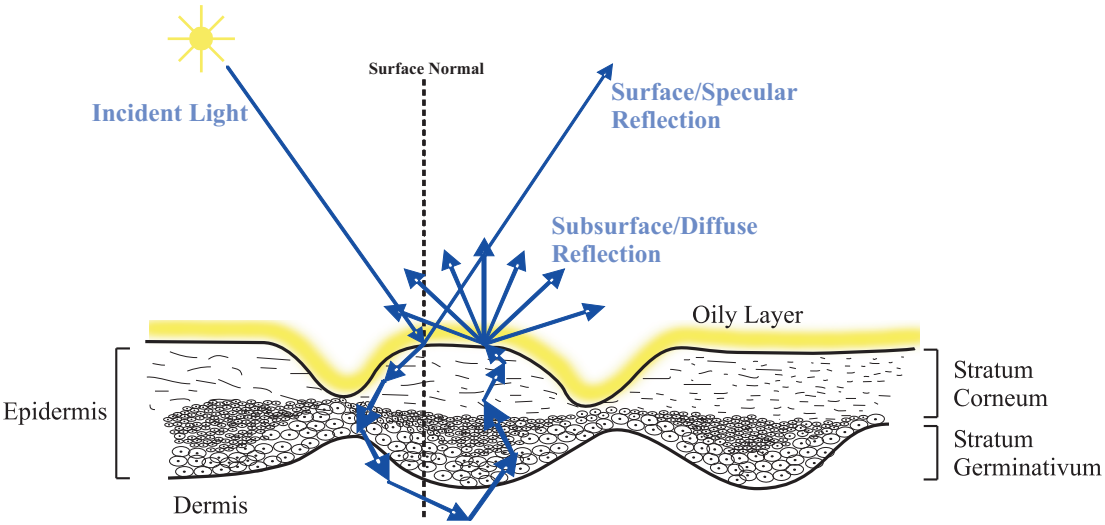


Figure 5.1: Interaction of light with the skin surface.

Reflectance images consist of a standard visible light image and two images acquired under polarized lighting. The standard visible light image refers to the skin image acquired under ordinary light. The intensity I_v measured by the sensor is the sum of the surface component, I_s, and the subsurface component, I_d,

$$I_v = I_s + I_d. \tag{5.1}$$

We use the term *visual image* for an image acquired under ordinary visible light. Polarized images are acquired by placing a linear polarizer in front of the light source and the sensor [1, 69, 26]. When the polarizer in front of the sensor is parallel to the polarizer in front of the light source, the entire surface component is measured by the sensor, but only half the subsurface component reaches the sensor. The intensity I_p measured by the sensor is,

$$I_p = I_s + \frac{1}{2} I_d. \tag{5.2}$$

We use the term *parallel-polarized image* for an image acquired with parallel polarizers in front of the light source and the sensor. The parallel-polarized image enhances the surface component, and brings out surface features like raised borders of lesions, pore structure, and wrinkles [33]. When the polarizer in front of the sensor is perpendicular to the polarizer in front of the light source, the entire surface component gets blocked, and the sensor measures only half the subsurface component. The intensity I_x measured

by the sensor is,

$$I_x = \frac{1}{2} I_d. \tag{5.3}$$

We use the term *cross-polarized image* for an image acquired with perpendicular polarizers in front of the light source and the sensor. The cross-polarized image brings out subsurface skin features like color variation due to melanin erythema [34].

Fluorescence images are obtained with either ultraviolet (UVA) excitation at 360-400 nm or blue light excitation at 400-460 nm. Under UVA excitation [4, 20] the collagen cross-links fluoresce and photodamaged regions appear as dark spots produced by absorbance of epidermal pigmentation. Collagen cross-links increase with exposure and age. Under blue light excitation elastin cross-links fluoresce. Blue light excitation enables us to detect both melanin pigmentation and superficial vasculature, and sebum producing pores appear as yellow-white spots.

Figure 5.2 shows an example image from each modality. The eye region in Figure 5.2 and all face images have been blacked out to preserve the identity of the subject. Figure 5.2 (a), (b), and (c) show reflectance images acquired under visible, parallel-polarized, and cross-polarized modalities. Figure 5.2 (d) and (e) show fluorescence images acquired under UVA and blue excitation.

5.3 Time-lapse Skin Images

We image a subject with acne lesions at 39 different time points over a three month period. Figure 5.3 shows example images from the set of 39 time-lapse images. The face images contain acne lesions predominantly in the forehead region. We precisely register the forehead region of the face images using the micro-level feature based registration approach in conjunction with a standard computer vision registration algorithm. We demonstrate the standard computer vision algorithm alone does not suffice for pore level registration. Figure 5.4 and 5.5 show the 39 forehead regions registered using the micro-level feature based approach. During the three month period the acne lesions appear, evolve, and disappear.

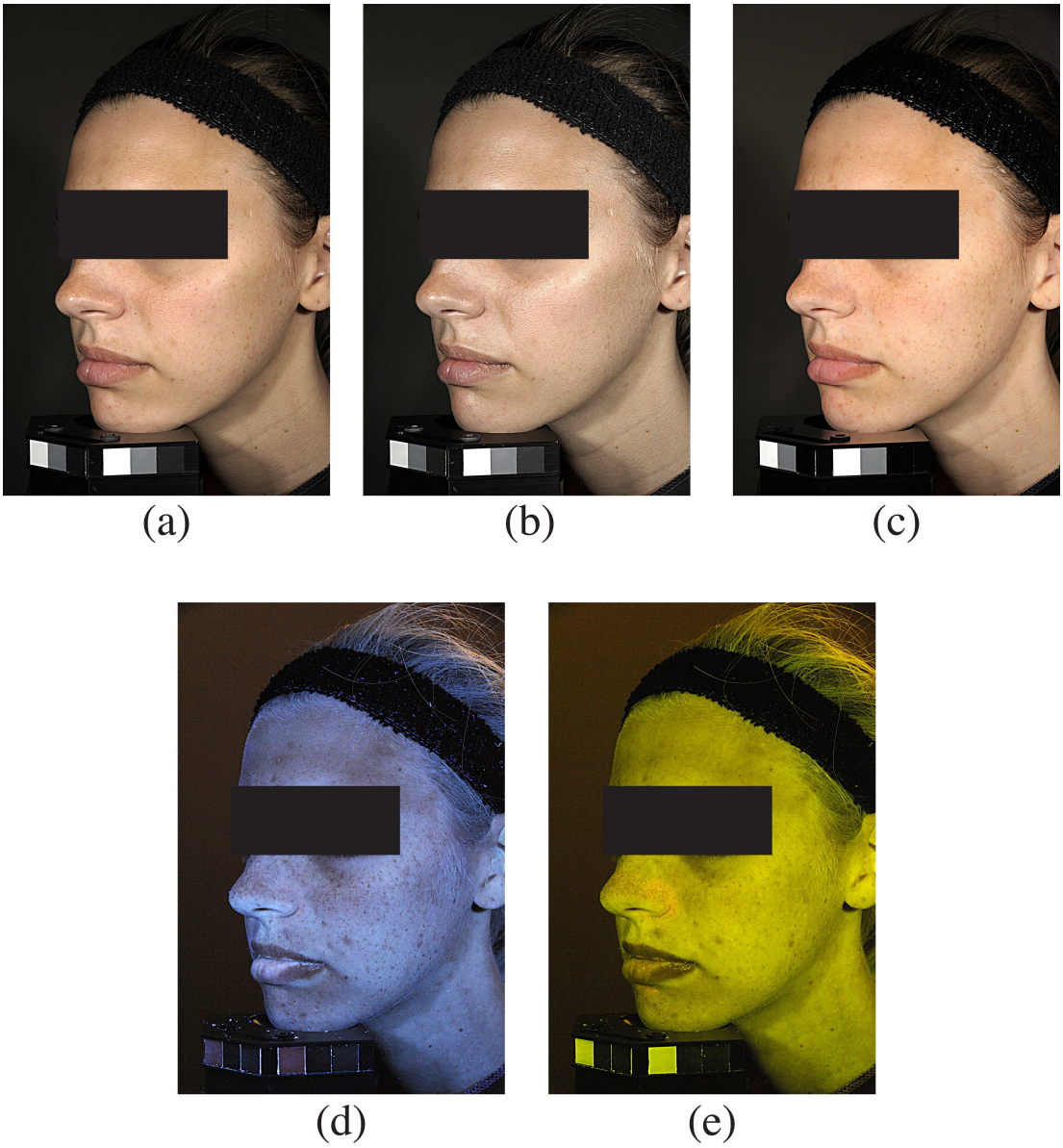


Figure 5.2: Multimodal skin images. Reflectance images: visual (a), parallel-polarized (b), and cross-polarized (c). Fluorescence images: UVA (d) and blue light (e) excitation.

Figure 5.3: Sample images of a person with acne lesions.

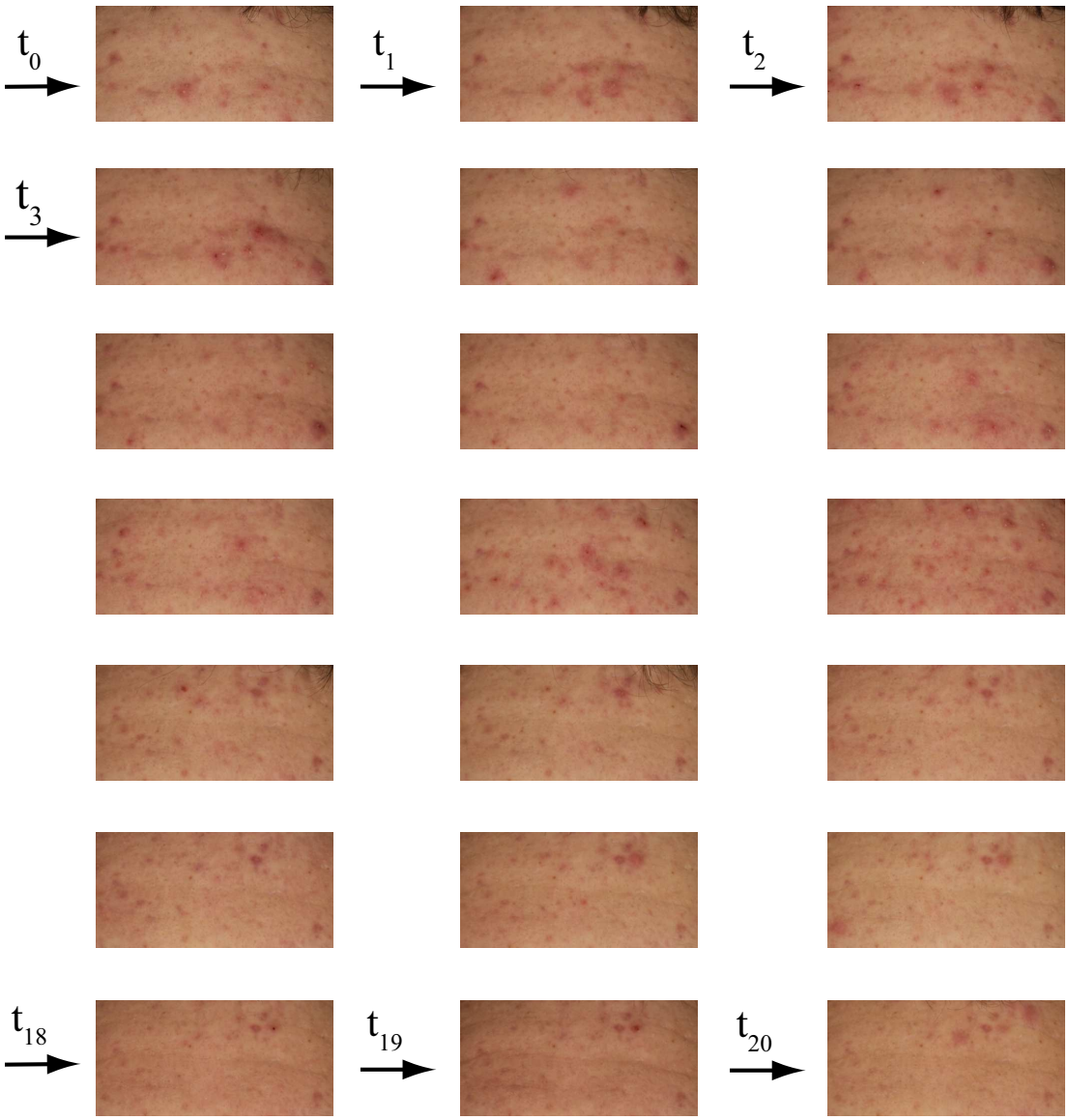


Figure 5.4: First 21 of the 39 time-lapse skin images with acne lesions.

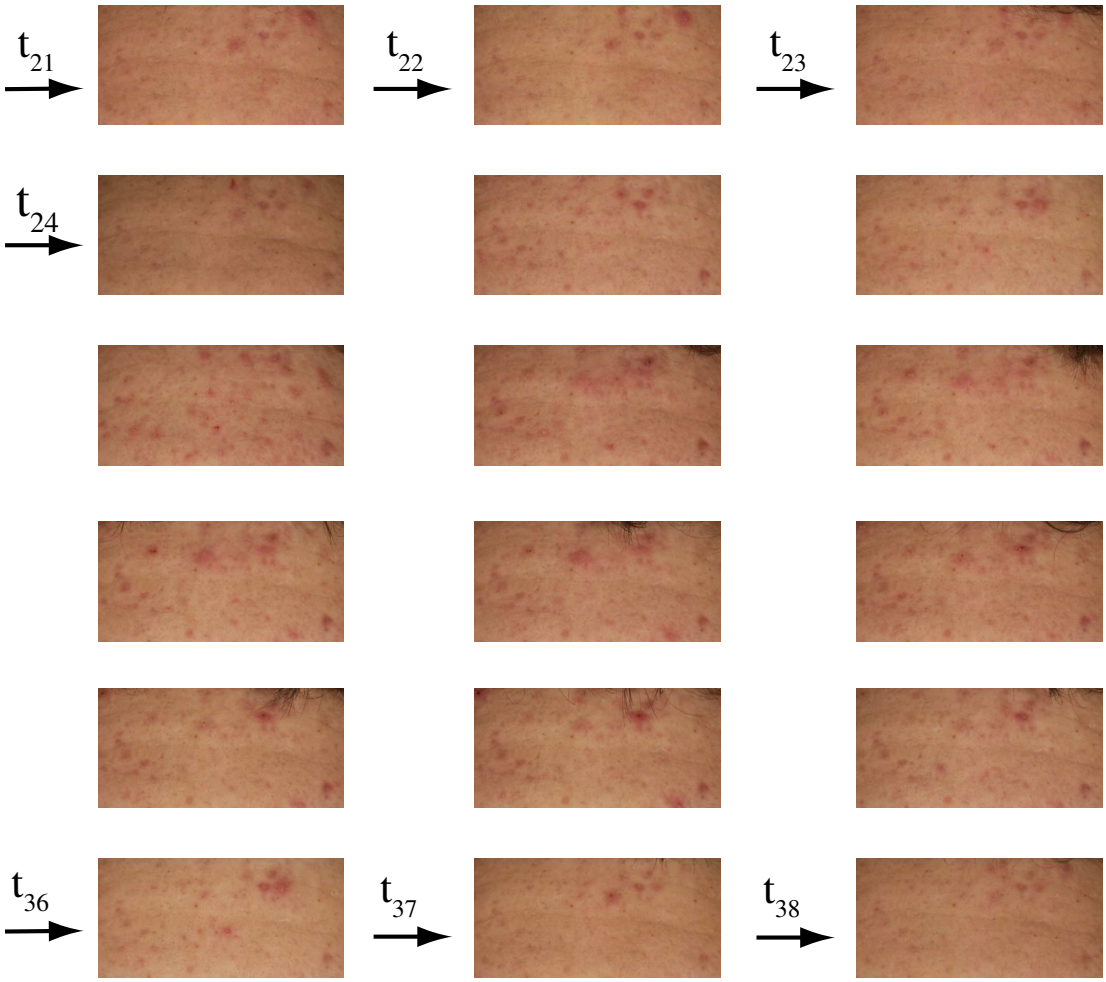


Figure 5.5: Last 18 of the 39 time-lapse skin images with acne lesions.

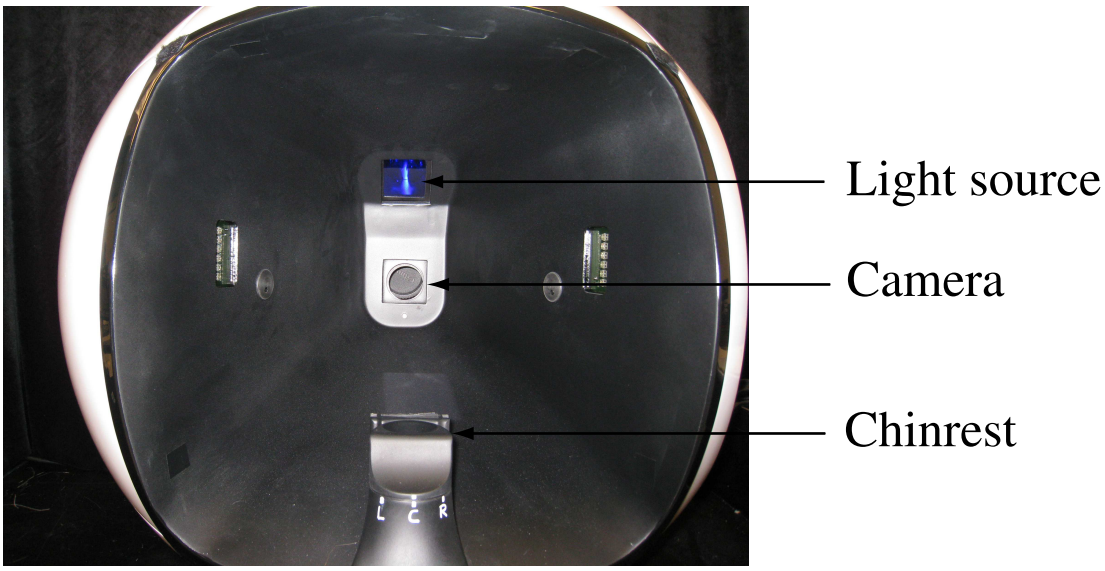


Figure 5.6: Imaging equipment used to acquire the skin images.

5.4 Imaging Equipment

We acquire the multimodal and time-lapse skin images using a custom built equipment shown in Figure 5.6. The imaging equipment consists of a light source, a sensor, and polarizer filters placed in front of the light source and the sensor. In each imaging session the subject rests his/her face on the chinrest, and images of the subject are acquired within a one second time interval. During the acquisition, the operation of the sensor, the light source, and the orientation of the polarizers are synchronized using a computer.

Chapter 6

Multimodal Registration

We use micro-level features to register multimodal images acquired in a single imaging session during a clinical study; the images have minute misregistration due to breathing and involuntary movement. We demonstrate that pore level alignment of multimodal images is required in order to perform point to point quantitative operations in dermatology applications.

6.1 Method

We discuss the registration of the cross-polarized and visual modalities. Registration of remaining modalities is identical. While registering fluorescence images we perform an additional pre-processing step of histogram matching before applying the registration approach discussed in this section.

Figure 6.1 (a) and (b) show a pair of 2602×3906 cross-polarized and visual images of a human face. We register the face images by dividing them into component patches and registering the corresponding patch images. Figure 6.1 (c) and (d) show 400×400 patches extracted around a common pixel location from each imaging modality. The patches extracted are small compared to the size of the face images, and can be approximated as quasi-planar surfaces. Note that the patches are extremely smooth and do not appear to contain high number of feature points. In traditional computer vision algorithms such regions are treated as featureless regions. However, the patch images are abundant in micro-level features like pores, wrinkles, and skin texture, and we register the patch images by extracting and matching the micro-level features. The visual patch is the reference image, and the cross-polarized patch is registered onto the visual patch. Registration of the cross-polarized and visual patch images consists of the following

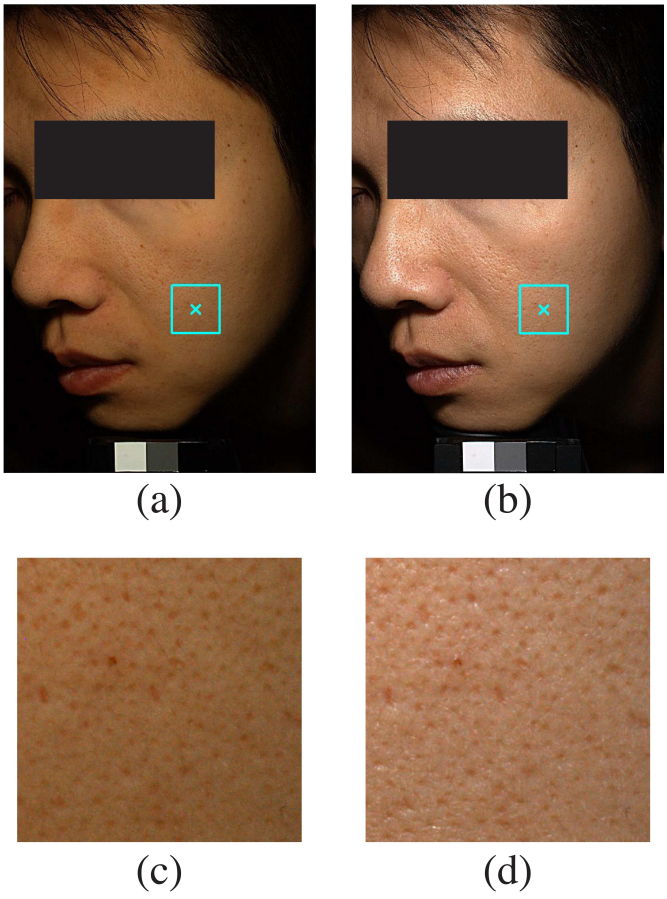


Figure 6.1: (a, b): Images of a face under cross-polarized and visual modalities. (d, e): Patches extracted from the cross-polarized and visual face images. In the face images, '×' denotes the common pixel location around which the patches have been extracted and the rectangle denotes the boundary of the patch.

steps:

6.1.1 Feature Extraction and Matching in Featureless Patches

We detect and match feature points in the patch images using the scale invariant feature transform (SIFT) [39] interest point detector, and applying a local intensity transform to enhance subtle features that are present in high resolution face images. We increase the contrast of the patch images by stretching the intensities of the images in grayscale such that one percent of the intensities are saturated. Figure 6.2 shows the original and the contrast enhanced patch images in grayscale. Features are clearly brought out in the contrast enhanced patch images. We run SIFT on the contrast enhanced

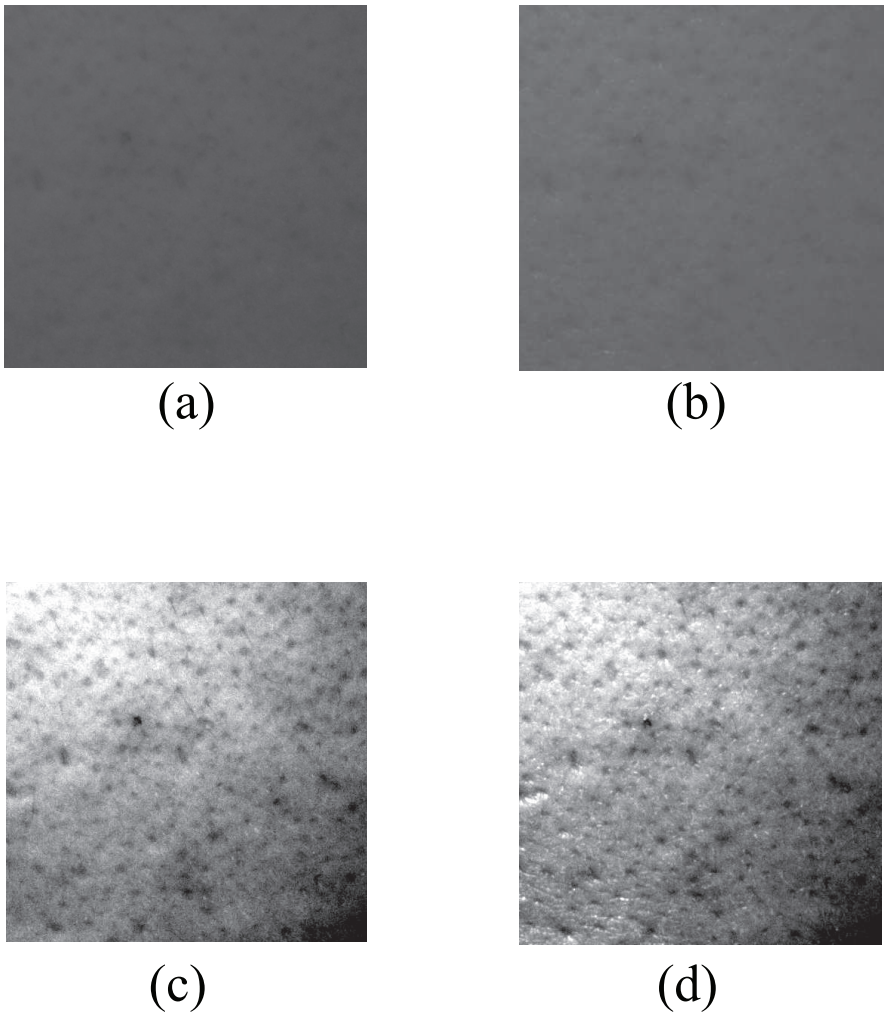
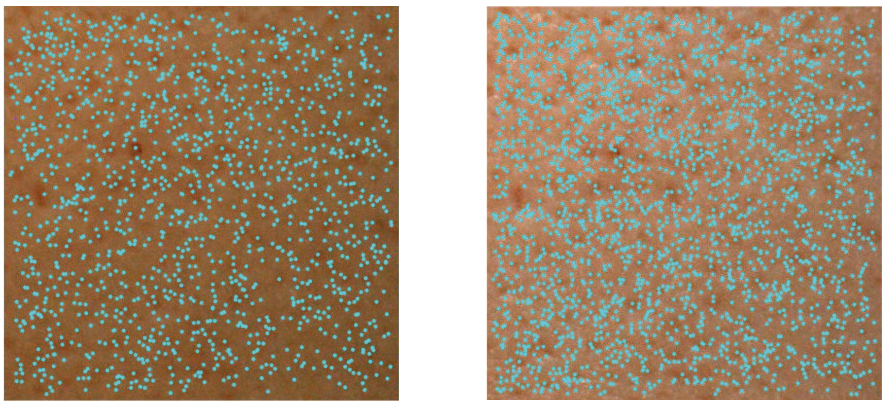


Figure 6.2: (a, b): Original cross-polarized and visual patch images in grayscale. (c, d): Grayscale contrast enhanced cross-polarized and visual patch images. Note that the smooth skin region has abundant features for alignment, which are clearly visible in the contrast enhanced patch images.

grayscale images. Figure 6.3 shows the feature points obtained when SIFT was run on the contrast enhanced patch images. A total of 1465 feature points were detected in the cross-polarized patch and 2215 feature points were detected in the visual patch. The point sets shown in Figure 6.3 are far too dense. However, a meaningful reduction of features can be accomplished.

SIFT describes each detected feature point using a 128 length descriptor vector. We match every feature point detected in the polarized patch image to the feature point in the visual patch image having the closest descriptor vector. Some of the matches



(a) (b)

Figure 6.3: Feature points detected when SIFT was run on the contrast enhanced cross-polarized (a) and visual (b) patch images.

are incorrect; therefore the matches obtained contains outliers. We initially detect and remove the easily detected outliers, which we refer as gross outliers. The matches retained after removing the gross outliers still contains some outliers. The remaining outliers are removed while estimating a homography between the quasi-planar patch images.

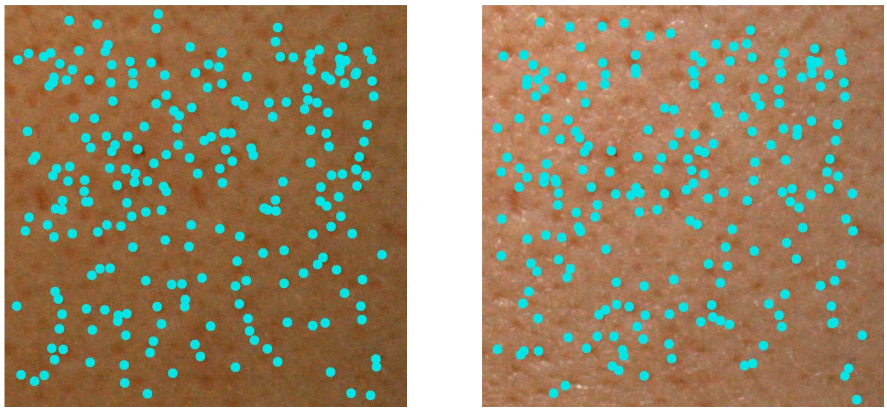
6.1.2 Removal of Gross Outliers

The extent of misregistration between the images is minute, but significant. Matches for which the difference in the pixel locations of the corresponding points is greater than a certain threshold are tagged as gross outliers and removed. We set a threshold Δx and Δy in the x and y directions. If a pair of corresponding points have pixel locations (x_1, y_1) and (x_2, y_2), then the pair is tagged as gross outliers if,

$$|x_2 - x_1| \geq \Delta x \tag{6.1}$$

$$|y_2 - y_1| \geq \Delta y. \tag{6.2}$$

We set Δx and Δy empirically. Figure 6.4 (a) and (b) show the feature points retained in the cross-polarized and visual patch images after removing the gross outliers. A total of 236 feature points were retained.



(a) (b)

Figure 6.4: Feature points retained in cross-polarized (a) and visual (b) patch images after removing the gross outliers.

6.1.3 Homography Estimation in Presence of Outliers

We use the matched feature points to estimate a transformation between the patch images. The transformation can be modeled as quadratic, affine, or homography, i.e., a 3×3 invertible matrix. Note that the cross-polarized and visual patch images correspond to a small region of the subject's face, which can be approximated as a quasi-planar surface, and the misregistered patch images can be seen as images from two viewpoints of the small quasi-planar surface. According to theory of multiview geometry, when a planar surface is imaged from two different viewpoints, it can be proved [22] that the transformation between the two images can be modeled using a homography; therefore we choose to model the transformation between the patch images using a homography.

In particular, there exists a matrix $\mathbf{H}$ such that,

$$s \begin{bmatrix} \mathbf{x}_1 \\ 1 \end{bmatrix} = \mathbf{H} \begin{bmatrix} \mathbf{x}_2 \\ 1 \end{bmatrix} \tag{6.3}$$

where $\mathbf{x}_1 = [x_1\, y_1]^T$ is the pixel location of a scene point in the cross-polarized image, $\mathbf{x}_2 = [x_2\, y_2]^T$ is the pixel location in the visual image, and s is a scale factor. Equation 6.3 can be rewritten as

$$\begin{bmatrix} \mathbf{x}_2^T & 1 & \mathbf{0}^T & 0 & -x_1\mathbf{x}_2^T & -x_1 \\ \mathbf{0}^T & 0 & \mathbf{x}_2^T & 1 & -y_1\mathbf{x}_2^T & -y_1 \end{bmatrix} \begin{bmatrix} \mathbf{h}_1 \\ \mathbf{h}_2 \\ \mathbf{h}_3 \end{bmatrix} = \mathbf{0}, \tag{6.4}$$

where $\mathbf{h}_1^T$, $\mathbf{h}_2^T$, and $\mathbf{h}_3^T$ are the first, second, and third rows of the homography matrix $\mathbf{H}$. Each pair of point correspondence gives two constraints on the homography matrix. A total of n point correspondences gives $2n$ constraints which can be written as

$$\mathbf{A}\mathbf{h} = \mathbf{0}, \tag{6.5}$$

where $\mathbf{h} = [\mathbf{h}_1^T\, \mathbf{h}_2^T\, \mathbf{h}_3^T]^T$, and $\mathbf{A}$ is a 2n×9 matrix whose (2i-1) and 2i, where i=1 . . . n, rows represent the constraints from the ith point correspondence.

Note that even after removing the gross outliers, the retained set of point correspondences contains outliers. Estimating homography by directly solving equation 6.5 in presence of outliers results in an incorrect homography estimation. In order to estimate the homography, we first remove the remaining outliers using the RANSAC [18] algorithm. The RANSAC algorithm consists of iterating the following steps for a specified number of iterations:

1. Randomly select four corresponding points from the set of point correspondences; a minimum of four point correspondences are required to solve equation 6.5.

2. Use the randomly selected set of four point correspondences to estimate the homography by solving equation 6.5.

3. Find the number of point correspondences which agree with the estimated homography. A pair of point correspondence $\mathbf{x}_1^i$ and $\mathbf{x}_2^i$ agrees with the estimated homography if the sum of the reprojection errors $\|\mathbf{x}_1^i - \hat{\mathbf{x}}_1^i\|^2 + \|\mathbf{x}_2^i - \hat{\mathbf{x}}_2^i\|^2$, where $\hat{\mathbf{x}}_1^i$ is the projection of $\mathbf{x}_2^i$ on the first image and $\hat{\mathbf{x}}_2^i$ is the projection of $\mathbf{x}_1^i$ on the second image using the estimated homography, is less than a pre-selected threshold.

After the iterations are complete, the homography for which maximum number of point correspondences agree is regarded as the correct homography estimate, the point

correspondences in agreement with it are regarded as inliers and retained, and the remaining point correspondences are regarded as outliers and discarded. Figure 6.5 (a) and (b) show the feature points retained in the cross-polarized and visual patch images, when the cross-polarized patch is registered onto the visual patch image. A total of 96 feature points were retained. In Figure 6.5 (c) the feature points are plotted together. White points indicate the pixel locations in the cross-polarized patch image and the black points indicate the pixel locations in the visual patch image. The difference in the locations of the white and black points indicates the cross-polarized and visual patch images are misregistered non-rigidly. Note that the extent of misregistration is small. However, the misregistration is significant for quantitative applications where precise point to point registration is required.

Once all the outliers are removed we use the inlier point correspondences to define the cost function,

$$F(\mathbf{h}) = \sum_i \|\mathbf{x}_1^i - \hat{\mathbf{x}}_1(\mathbf{h}, \mathbf{x}_2^i)\|_2^2 + \|\mathbf{x}_2^i - \hat{\mathbf{x}}_2(\mathbf{h}, \mathbf{x}_1^i)\|_2^2, \tag{6.6}$$

where $\hat{\mathbf{x}}_1(\mathbf{h}, \mathbf{x}_2^i)$ is the projection of $\mathbf{x}_2^i$ on the first image, $\hat{\mathbf{x}}_2(\mathbf{h}, \mathbf{x}_1^i)$ is the projection of $\mathbf{x}_1^i$ on the second image, and $\mathbf{h}$ is the vector obtained by stacking the rows of the homography matrix. The summation is performed over all the inlier point correspondences. Final homography estimate is the one which minimizes the cost function defined in equation 6.6. The optimal solution of equation 6.6 minimizes the total reprojection error when points from the second image are projected onto the first image, and points from the first image are projected onto the second image. We estimate the optimal solution using the Levenberg-Marquardt optimization algorithm [57, 61]. Levenberg-Marquardt is an iterative method for solving non-linear least squares problem. The initial starting point is the solution of equation 6.5 obtained using all the inlier point correspondences. Each iteration updates the current estimate, $\mathbf{h}_k$, of the homography to $\mathbf{h}_k + \Delta\mathbf{h}_k$. The step length $\Delta\mathbf{h}_k$ is obtained by solving the subproblem,

$$\underset{\Delta\mathbf{h}_k}{\text{minimize}} \quad F(\mathbf{h}_k) + \nabla F(\mathbf{h}_k)^T \Delta\mathbf{h}_k + \tfrac{1}{2}\Delta\mathbf{h}_k^T \nabla^2 F(\mathbf{h}_k) \Delta\mathbf{h}_k \tag{6.7}$$

$$\text{subject to } \|\Delta\mathbf{h}_k\|_2 \leq \Delta_k, \tag{6.8}$$

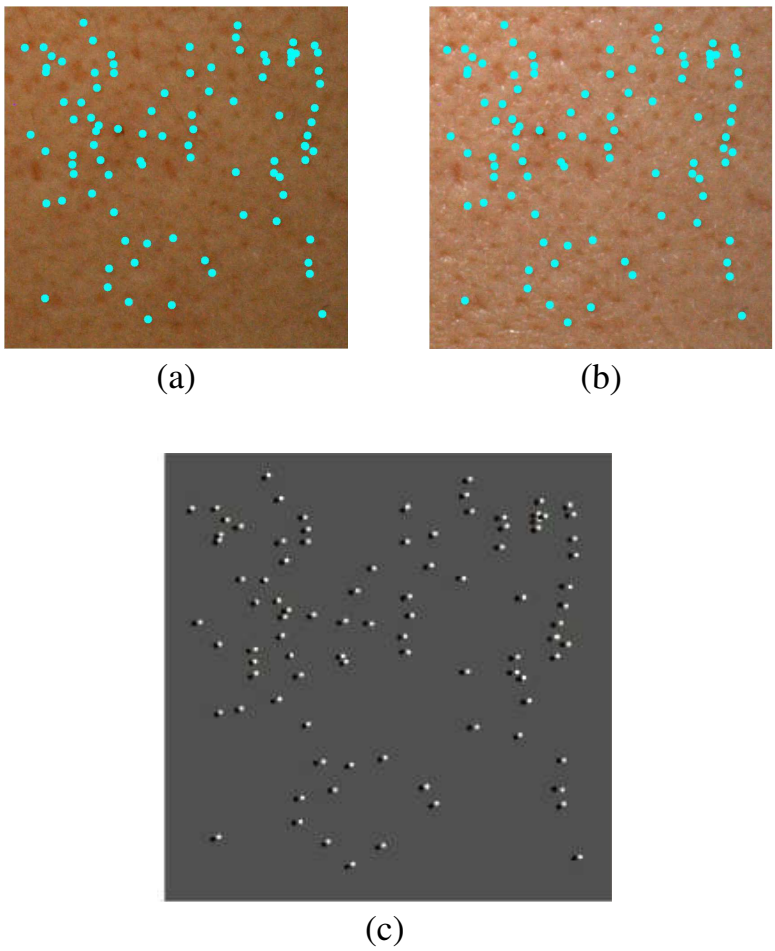


Figure 6.5: (a, b): Feature points retained during the registration of the cross-polarized patch onto the visual patch image. (c): Feature points retained during the registration of the cross-polarized and visual patches are plotted together in black and white points respectively. The difference in the locations indicate the patch images are misregistered.

where Δ_k is the maximum allowed step size for the current iteration. The Levenberg-Marquardt algorithm approximates the Hessian as $\nabla^2 F(\mathbf{h}_k) \approx \mathbf{J}(\Delta\mathbf{h}_k)^T \mathbf{J}(\Delta\mathbf{h}_k)$, where $\mathbf{J}(\Delta\mathbf{h}_k)$ is the Jacobian of the vector obtained by stacking each Euclidean distance in equation 6.6. It can be shown that the approximation is reasonable for non-linear least-squares objective functions. The subproblem can be solved in two steps,

1. Solve the equation $\mathbf{J}(\Delta\mathbf{h}_k)^T \mathbf{J}(\Delta\mathbf{h}_k)\mathbf{h} = -\nabla F(\mathbf{h}_k)$. If $\|\mathbf{h}\|_2 < \Delta$, then set $\Delta\mathbf{h}_k = \mathbf{h}$.

2. If the solution obtained in step one does not satisfy $\|\mathbf{h}\|_2 < \Delta$, then solve

$(\mathbf{J}(\Delta\mathbf{h}_k)^T\mathbf{J}(\Delta\mathbf{h}_k)+\lambda\mathbf{I})\mathbf{h} = -\nabla F(\mathbf{h}_k)$ such that $\|\mathbf{h}\|_2 = \Delta$, where $\lambda > 0$. The solution can be obtained using the Newton root finding algorithm [57]. Set $\Delta\mathbf{h}_k = \mathbf{h}$.

The final estimated homography is used to warp the input patch image onto the reference patch image using bilinear interpolation. Precise point to point registration of corresponding patch images results in a precise point to point registration of the entire face images.

6.2 Registration Results

Figures 6.6 (a)-(e) show the images of a 400×400 skin patch acquired in quick succession under parallel-polarized, visual, cross-polarized, UVA excitation, and blue excitation modalities. The skin appearance is different under different modalities. We register the entire multimodal set by registering: a) parallel-polarized and visual images, b) visual and cross-polarized images, c) UVA excited fluorescence and cross-polarized images, and d) blue excited fluorescence and UVA excited fluorescence images. Each pair of multimodal images has a unique set of corresponding micro-level features. Figure 6.7 (a)-(d) show the pixel locations of corresponding feature points in the misregistered parallel-polarized and visual, cross-polarized and visual, UVA excitation and cross-polarized, and blue excitation and UVA excitation images plotted together as white and black points. Note that the pixel locations do not overlap indicating misregistration. The root mean square (RMS) values of the difference between the pixel locations in the misregistered images are 21.93, 19.00, 9.58, and 3.45 respectively. The images being registered are very high resolution; therefore a small change in the location of a scene point results in a large difference between the pixel corresponding to the new scene location and the pixel corresponding to the original scene location. Figure 6.8 (a)-(d) show the pixel locations of the corresponding feature points in the registered images. Note that the pixel locations overlap indicating precise registration. The RMS values of the difference between the pixel locations in the registered images are 0.51, 0.81, 0.99, and 0.82.

Figure 6.9 (a) and (b) show 1900×2300 full face images acquired in quick succession

Figure 6.6: Images of a skin patch acquired in quick succession under parallel-polarized, visual, cross-polarized, UVA excitation, and blue excitation modalities.

under parallel-polarized and visual modalities. Figure 6.9 (c) and (d) show the pixel locations of corresponding feature points in the misregistered and registered polarized and visual images as white and black points; note that for the misregistered images the points do not precisely overlap. The RMS value of the difference between the pixel locations in the misregistered images is 15.43 and in the registered images is 0.75.

Figures 6.10 (a)-(e) show the images of a large 1000×850 skin region acquired in quick succession under parallel-polarized, visual, cross-polarized, UVA excitation, and blue excitation modalities. Figure 6.11 (a)-(d) show the pixel locations of corresponding feature points in the misregistered parallel-polarized and visual, cross-polarized and visual, UVA excitation and cross-polarized, and blue excitation and UVA excitation images plotted together as white and black points. Note that the pixel locations do not overlap indicating misregistration. The root mean square (RMS) values of the difference between the pixel locations in the misregistered images are 12.20, 9.54, 24.55, and 14.39 respectively. Figure 6.12(a)-(d) show the pixel locations of the corresponding feature points in the registered images. Note that the pixel locations overlap indicating precise registration. The RMS values of the difference between the pixel locations in the registered images are 0.80, 0.91, 1.46, and 0.90.

The results show that we have achieved precise point to point alignment between multimodal images across which skin appearance varies significantly. The high accuracy alignment was made possible by leveraging micro-level features like pores, wrinkles, and skin texture.

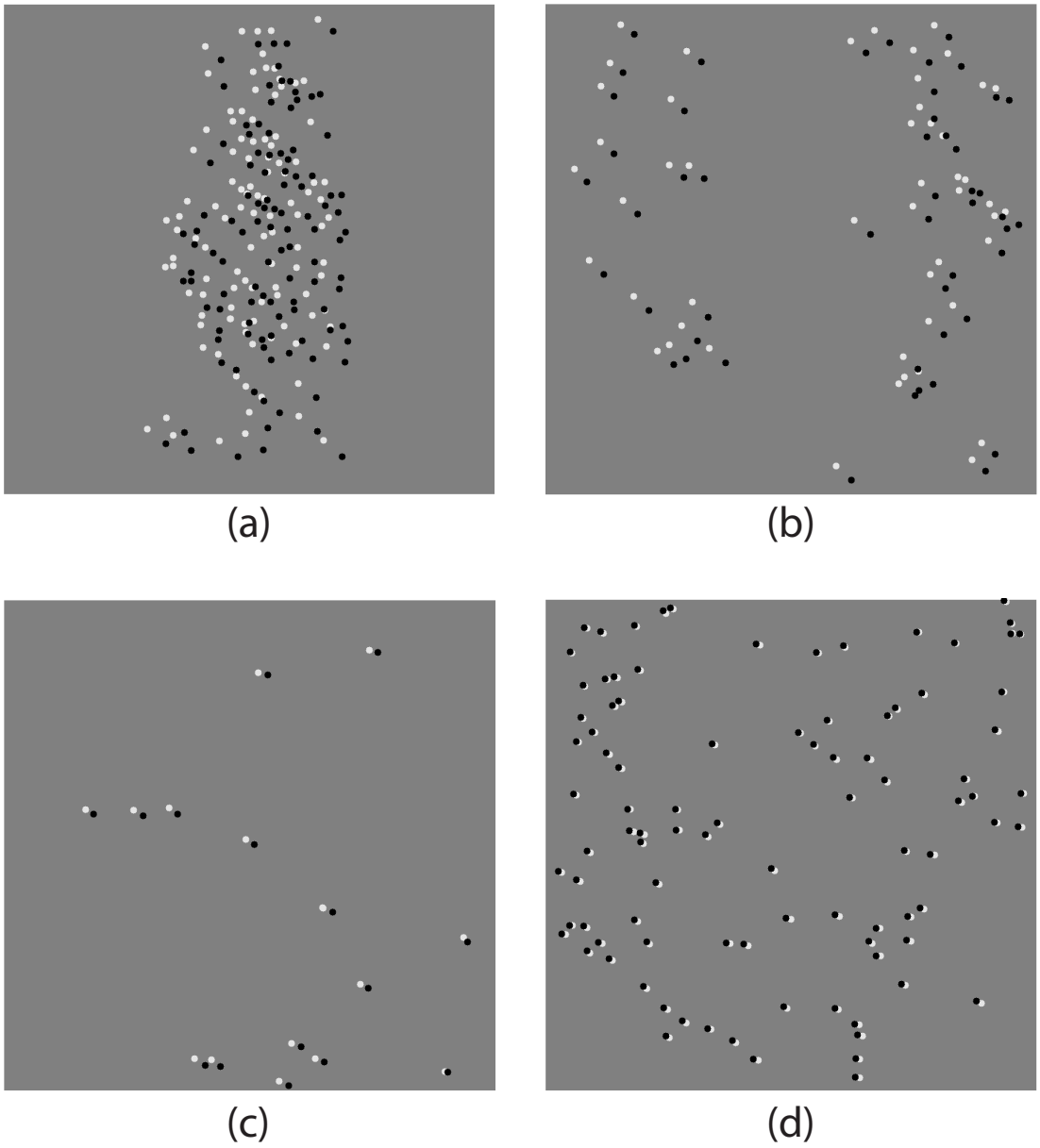


Figure 6.7: Pixel locations of corresponding misregistered feature points in: (a) parallel-polarized and visual, (b) cross-polarized and visual, (c) UVA excitation and cross-polarized, and (d) blue excitation and UVA excitation images; plotted together as white and black points respectively.

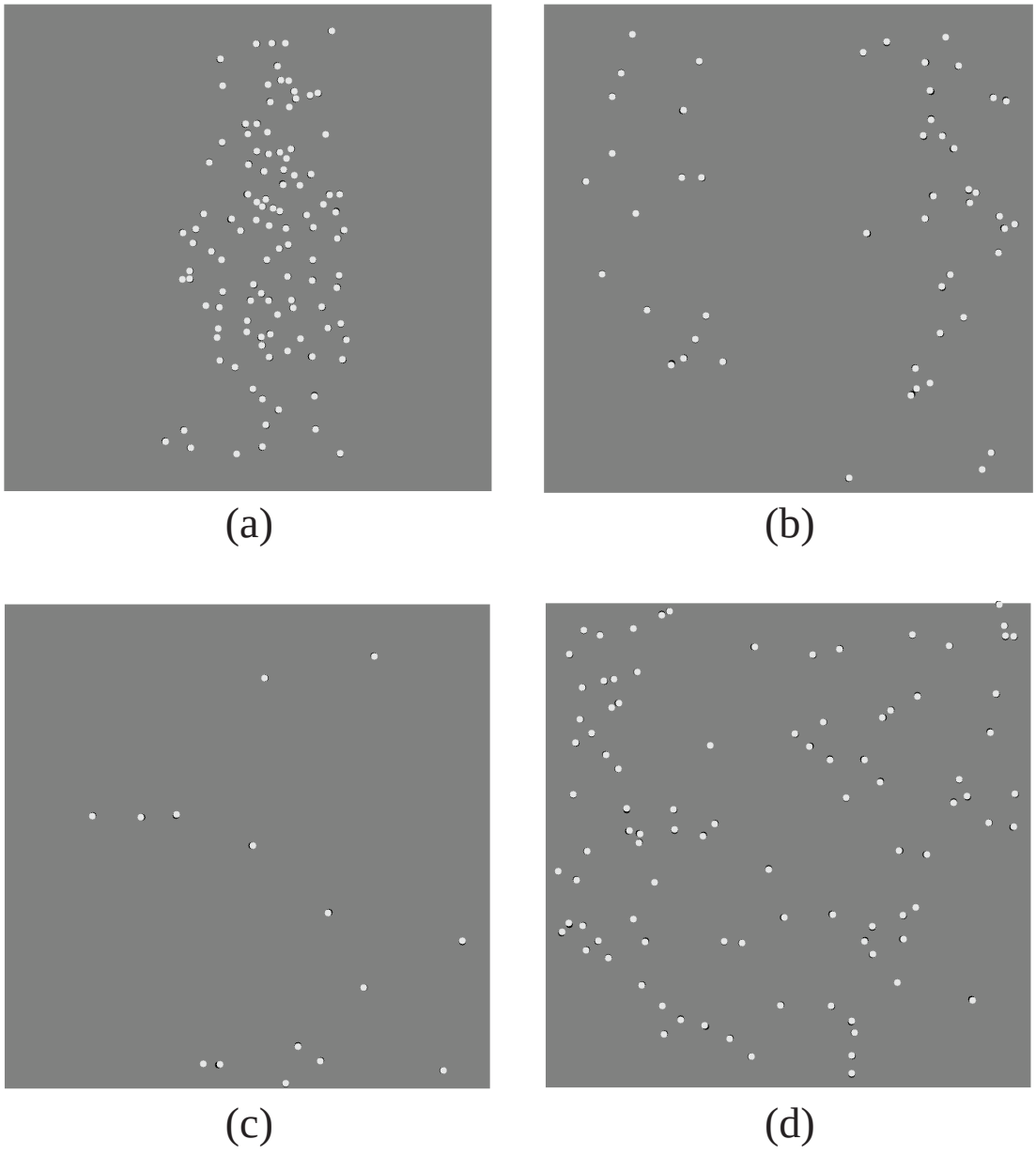


Figure 6.8: Pixel locations of corresponding registered feature points in: (a) parallel-polarized and visual, (b) cross-polarized and visual, (c) UVA excitation and cross-polarized, and (d) blue excitation and UVA excitation images; plotted together as white and black points respectively. For perfect registration only white points appear (superimposed on the black points).

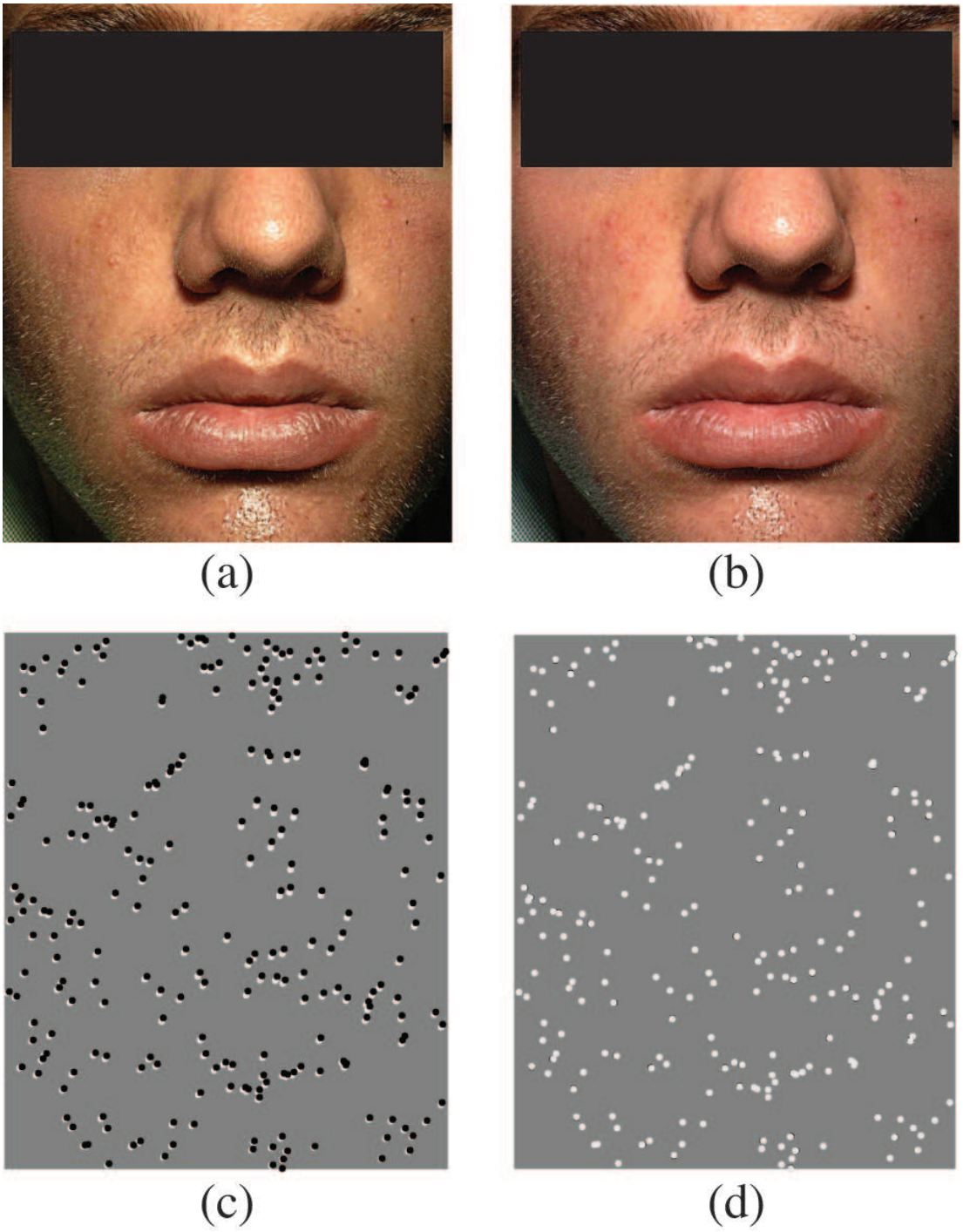


Figure 6.9: (a, b): Parallel-polarized and visual full face images acquired in quick succession. (c): Pixel locations of corresponding feature points in the misregistered parallel-polarized and visual images plotted together. (d): Pixel locations of the corresponding feature points in the registered parallel-polarized and visual patch images plotted together. The pixel locations in the polarized image are indicated in white and the pixel locations in the visual image are indicated in black.

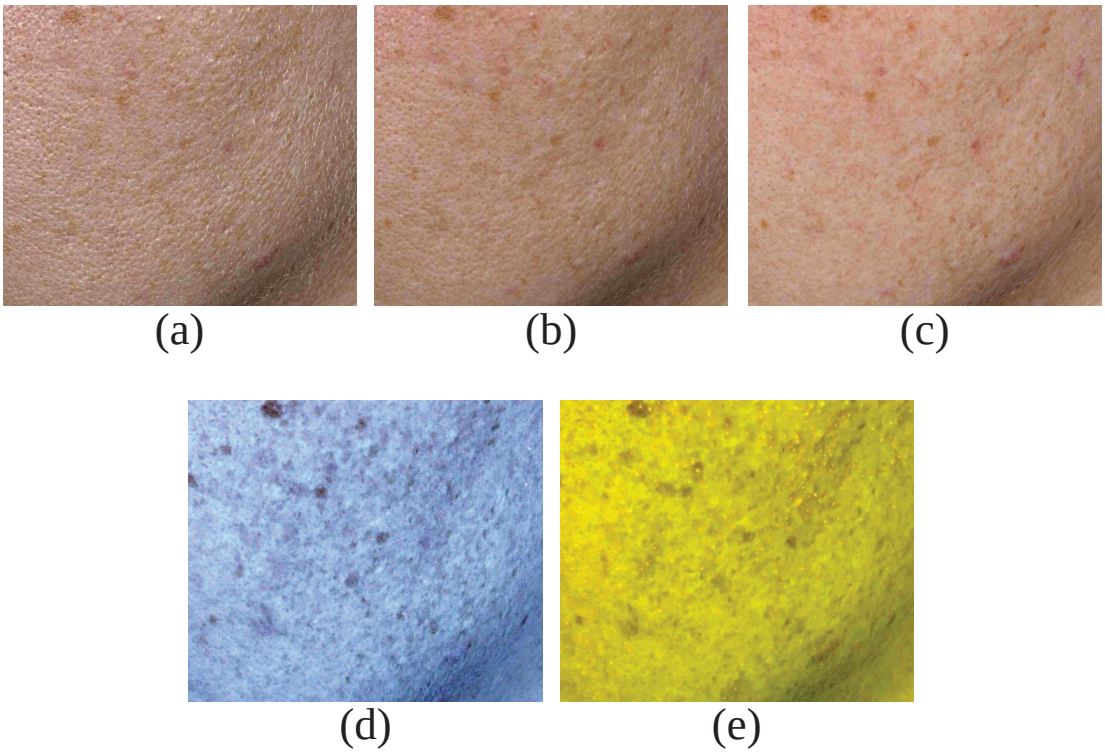


Figure 6.10: Images of a large skin region acquired in quick succession under parallel-polarized, visual, cross-polarized, UVA excitation, and blue excitation modalities.

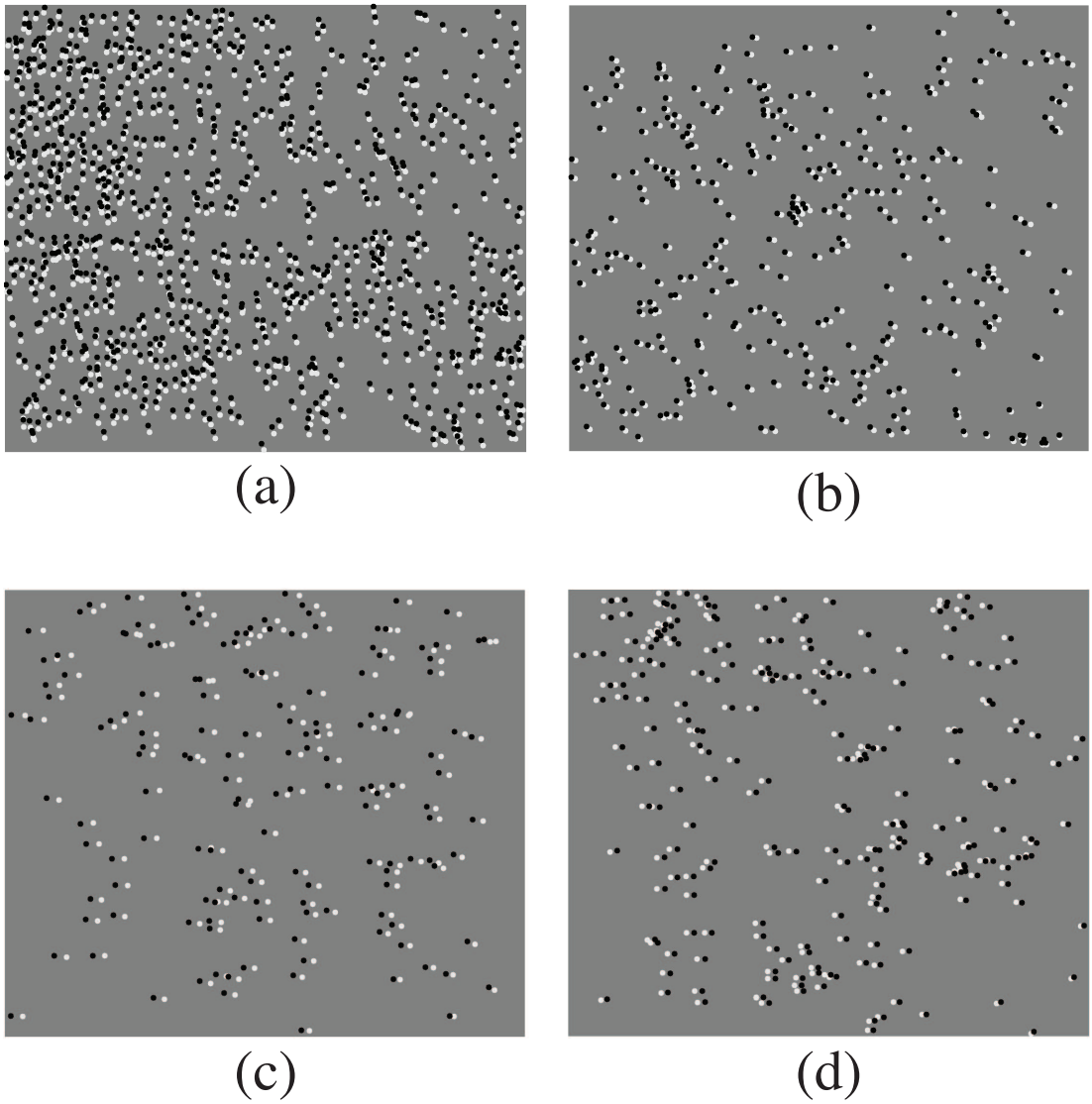


Figure 6.11: Pixel locations of corresponding misregistered feature points in: (a) parallel-polarized and visual, (b) cross-polarized and visual, (c) UVA excitation and cross-polarized, and (d) blue excitation and UVA excitation images; plotted together as white and black points respectively.

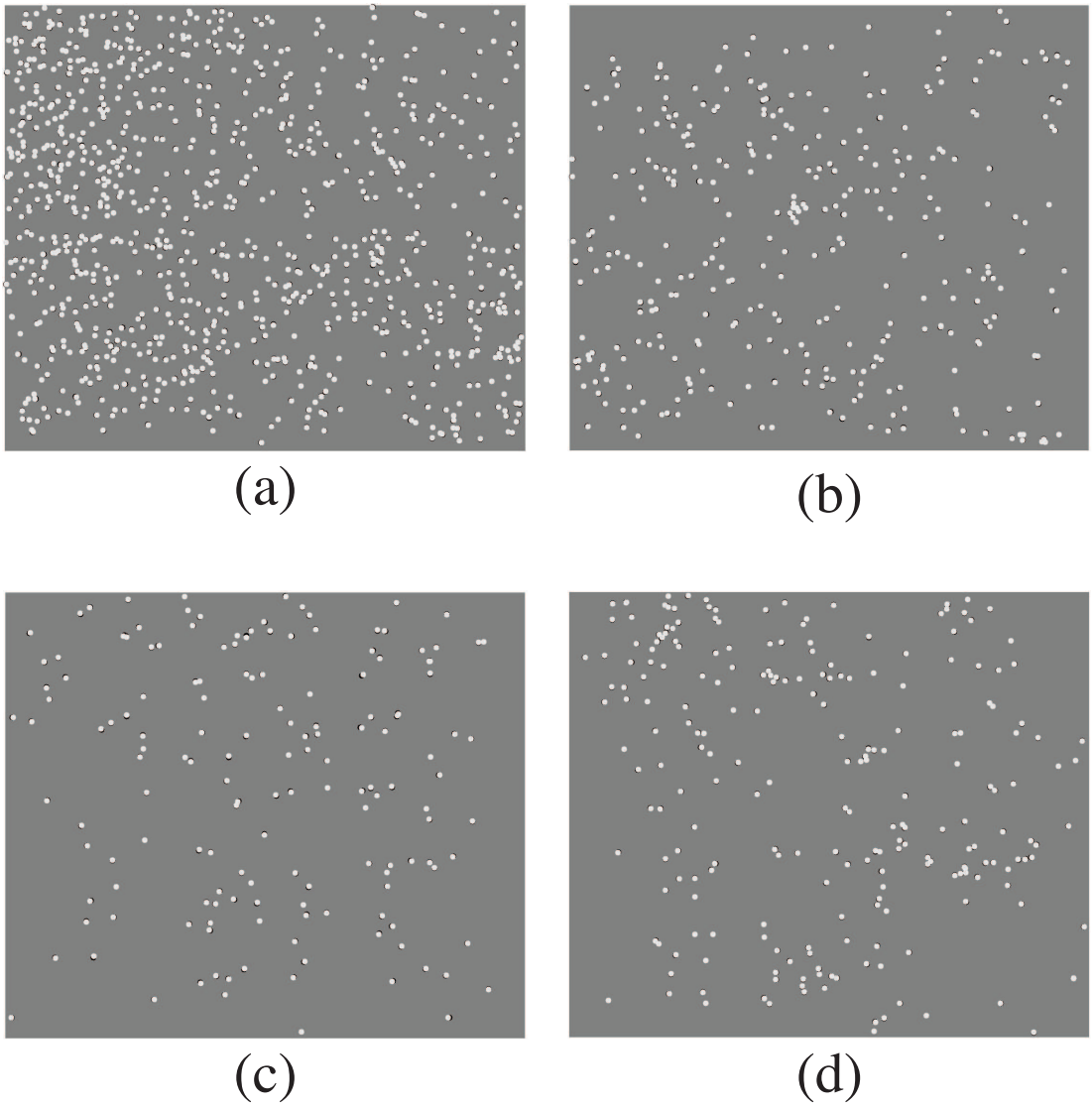


Figure 6.12: Pixel locations of corresponding registered feature points in: (a) parallel-polarized and visual, (b) cross-polarized and visual, (c) UVA excitation and cross-polarized, and (d) blue excitation and UVA excitation images; plotted together as white and black points respectively. For perfect registration only white points appear (superimposed on the black points).

6.2.1 Dermatology Task: Recovering Surface Reflectance

In dermatology it is common to obtain the surface component by subtracting the cross-polarized image from the parallel-polarized image [62, 28, 27]. However, the surface component obtained by subtracting the misregistered input polarized images, without pore level registration, has misregistration artifacts. Figure 6.13 (a) and (b) show a pair of polarized images acquired in quick succession. Figure 6.13 (c) shows the surface component recovered using the input polarized images, and Figure 6.13 (e) shows a close up of the surface component of the nose region within the rectangular boundary marked on the polarized images. Misregistration artifacts are clearly seen in Figure 6.13 (e) in the form of dark spots throughout the image. Misregistration artifacts are also seen in Figure 6.13 (c) around the bottom tip of the nose region. The surface component can be accurately recovered by registering the input polarized images up to the resolution of pore level features using the micro-level feature-based registration algorithm before subtraction. Figure 6.13 (d) shows the surface component recovered by subtracting the registered polarized images, and Figure 6.13 (f) shows the close up of the surface component of the nose region obtained using the registered images. Note that Figure 6.13 (f) does not contain any misregistration artifacts.

Figure 6.14 shows additional examples of surface component recovery using the input and registered polarized images. In each row the first two images show the input cross-polarized and the input parallel-polarized patch images, the third image shows the recovered surface component obtained using the input polarized images, and the fourth image shows the surface component recovered using the registered polarized images. In the first row the incorrect reconstruction due to misregistration is clearly seen around the shadow region in the top part of the image, and in the second row the incorrect reconstruction due to misregistration is clearly seen along the left boundary region of the skin image. The misregistration artifacts are not present in the surface component recovered using the registered patch images.

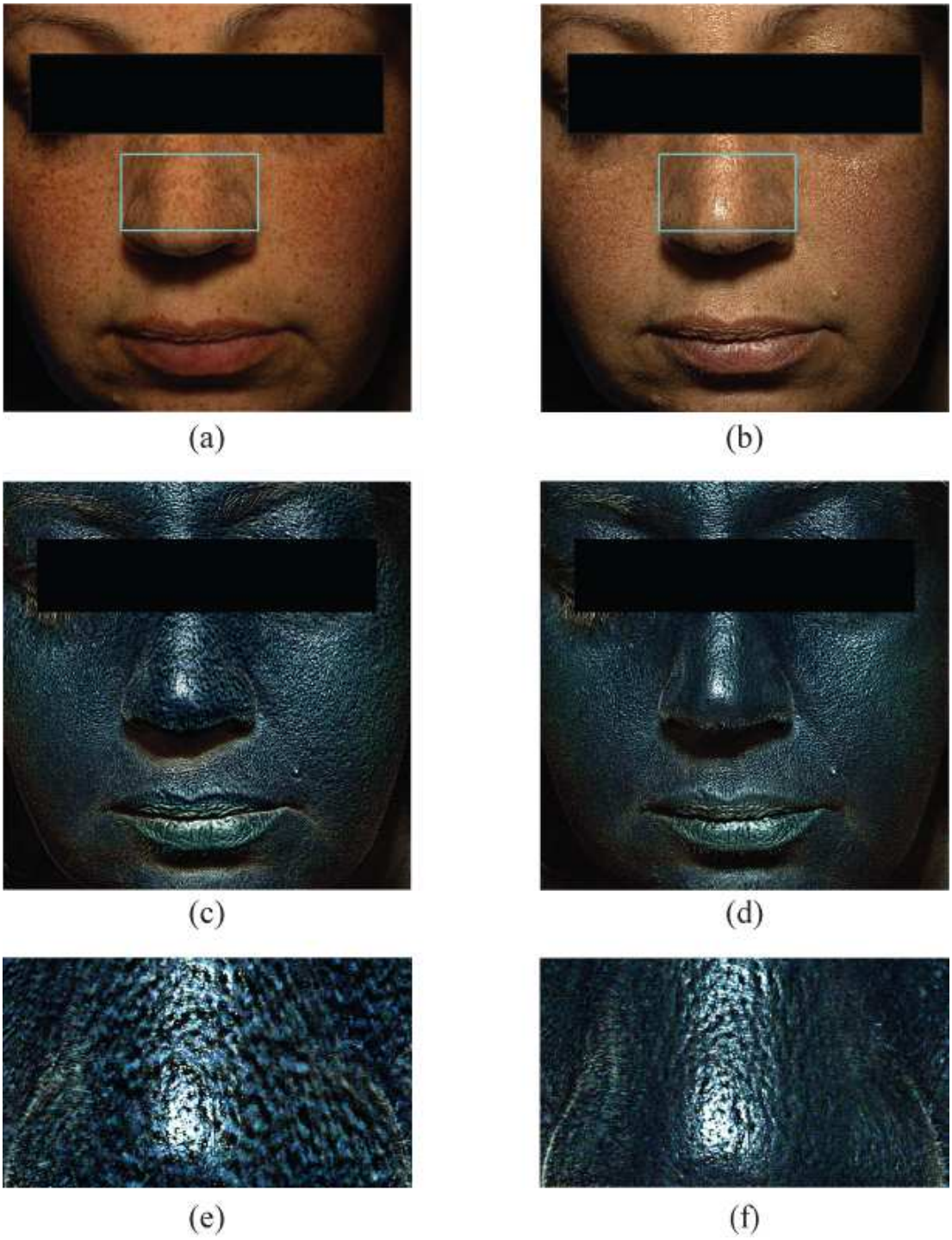


Figure 6.13: *Please view the images in color.* (a, b): Input cross and parallel-polarized full face images. (c, d): Surface component recovered using the input and registered polarized images. (e, f): Surface component of nose region within the rectangular boundary marked on the polarized images recovered using input and registered images. In (e) the misregistration artifacts are clearly seen in the form of dark spots throughout the image, and in (c) the misregistration artifacts are seen around the bottom tip of the nose region. Misregistration artifacts are not present in the surface component recovered using the registered images.

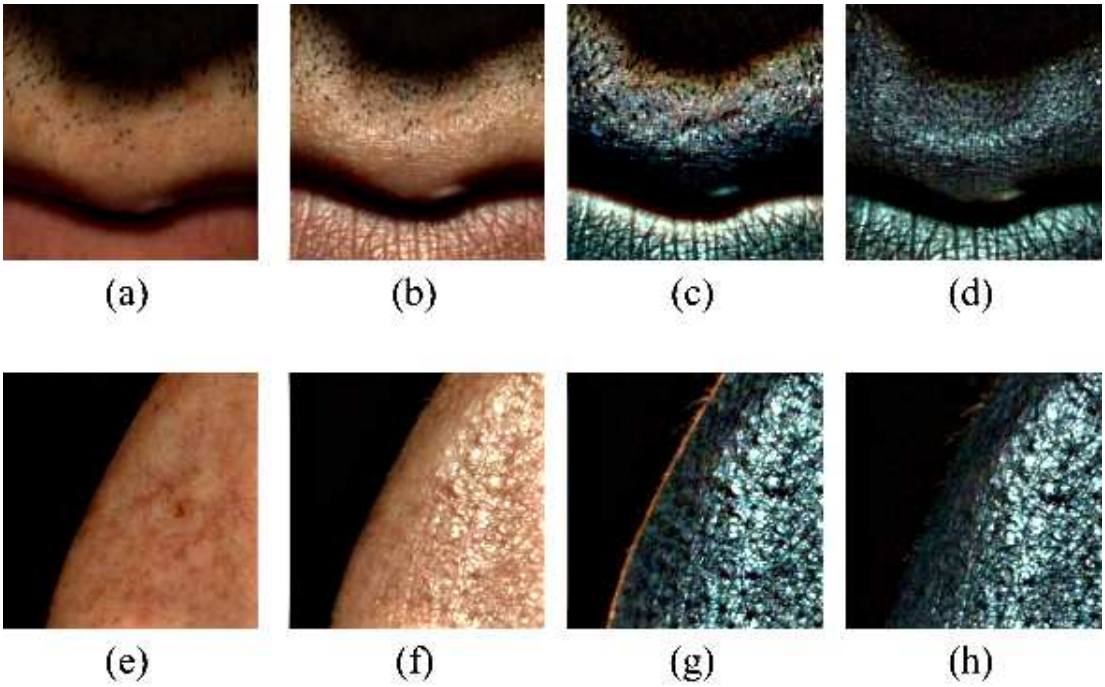


Figure 6.14: *Please view the images in color.* In each row the first two images are the input cross-polarized and parallel-polarized images, the third image is the surface component obtained using the input polarized images, and the fourth image is the surface component obtained using the registered images. Note that the surface component recovered using the registered images do not have any misregistration artifacts.

Chapter 7

Time-lapse Registration

We image a subject with acne lesions at 39 different time points over a three month period. The time-lapse images are misregistered, which makes it difficult to track the evolution of acne lesions over time, or perform quantitative processing on them. Precise point to point registration of the temporal images clearly bring out the evolution of acne lesions over time. We use the registered set to model and classify acne-like regions, i.e., regions which appear like acne lesions and whose appearance evolves with time.

7.1 Method

Figure 7.1 (a) and (b) show the 980×664 top part of two face images in the database.

In the first step, we globally register the top part of the images using the Lucas-Kanade algorithm [5]. The Lucas-Kanade algorithm is applied to the grayscale version of the images. Let $I_1(\mathbf{x})$ and $I_2(\mathbf{x})$ denote the intensity values of the two images at pixel $\mathbf{x}$. Let $\mathbf{W}(\mathbf{x};\mathbf{a})$ denote the warp which maps a pixel $\mathbf{x}$ in I_2 to I_1. The vector $\mathbf{a}$ is the parameters of the warp. The Lucas-Kanade algorithm determines the warp parameters by minimizing the objective function,

$$\sum_{\mathbf{x}} \|I_1(\mathbf{W}(\mathbf{x};\mathbf{a})) - I_2(\mathbf{x})\|_2^2. \tag{7.1}$$

The Lucas-Kanade algorithm starts with a current estimate of warp parameters $\mathbf{a}$ and in each iteration updates the parameter values by,

$$\Delta\mathbf{a} = \mathbf{U}^{-1} \sum_{\mathbf{x}} [\bigtriangledown I_1 \frac{\partial \mathbf{W}}{\partial \mathbf{p}}]^T [I_2(\mathbf{x}) - I_1(\mathbf{W}(\mathbf{x};\mathbf{a}))], \tag{7.2}$$

where $\mathbf{U}$ is the Hessian matrix.

Figure 7.2 (a) and (b) show the globally registered images. Figure 7.2 (c) shows the pixel locations of the corresponding micro-level feature points in the original input

(a) (b)

Figure 7.1: Top part of two face images in the database.

images of the forehead region plotted together. Figure 7.2 (d) shows the pixel locations of the corresponding micro-level feature points in the globally registered images plotted together. The white points indicate the pixel locations in the skin region extracted from Figure 7.2 (a) and the black points indicate the pixel locations in the skin region extracted from Figure 7.2 (b). Note that in Figure 7.2 (d) the pixel locations of corresponding feature points are much closer to each other indicating global registration; however, the pixel locations do not precisely overlap indicating residual misregistration. In the second step, we use the micro-level feature-based registration approach discussed in section 6.1 to remove the residual misregistration between the globally registered images. We present the final point to point registration result obtained after removing the residual misregistration in the results section.

7.1.1 Registration Results

Figure 7.3 (a) shows two of the 39 skin images with acne lesions extracted from face images acquired during the three month clinical study. Figure 7.3 (b) shows the pixel locations of the corresponding micro-level feature points in the input images, and Figure 7.3 (c) shows the pixel locations of the corresponding feature points after globally registering the images. Note that the globally registered images have residual misregistration. Figure 7.3 (d) shows the pixel locations of the corresponding feature points after applying the micro-level feature-based registration algorithm. Note that in Figure 7.3 (d) the pixel locations of corresponding feature points exactly overlap indicating

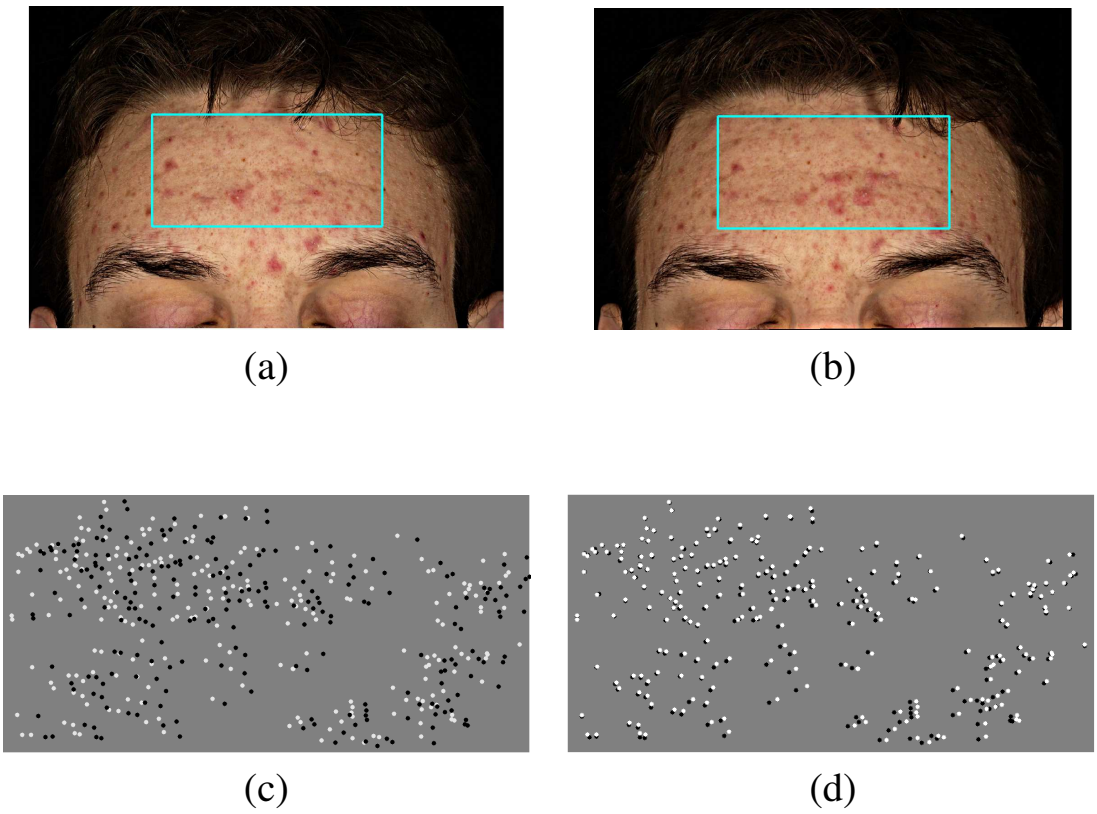


Figure 7.2: (a, b): Globally registered top part of face images. We extract and precisely register the skin regions within the rectangular boundary. (c, d): Pixel locations of corresponding micro-level feature points in the input and globally registered skin regions. The white and black points indicate the pixel locations in the skin region extracted from (a) and (b) respectively.

the images are precisely registered. The RMS value of the difference between the pixel locations is 3.198 for the globally registered images, and 0.861 for the final micro-level feature-based registration approach. Videos showing the entire set of 39 misregistered and the registered time-lapse images are available online. [1]

7.1.2 Dermatology Task: Detecting Acne-Like Regions

We model acne-like regions using a six dimensional feature vector representing temporal and spatial variations [42], and classify using logistic regression [23]. The classification experiment is a small scale experiment designed to show the utility of temporal features. Computing the temporal-spatial features requires a precise point to point registration

[1] `http://www.ece.rutgers.edu/~kdana/Research.html`

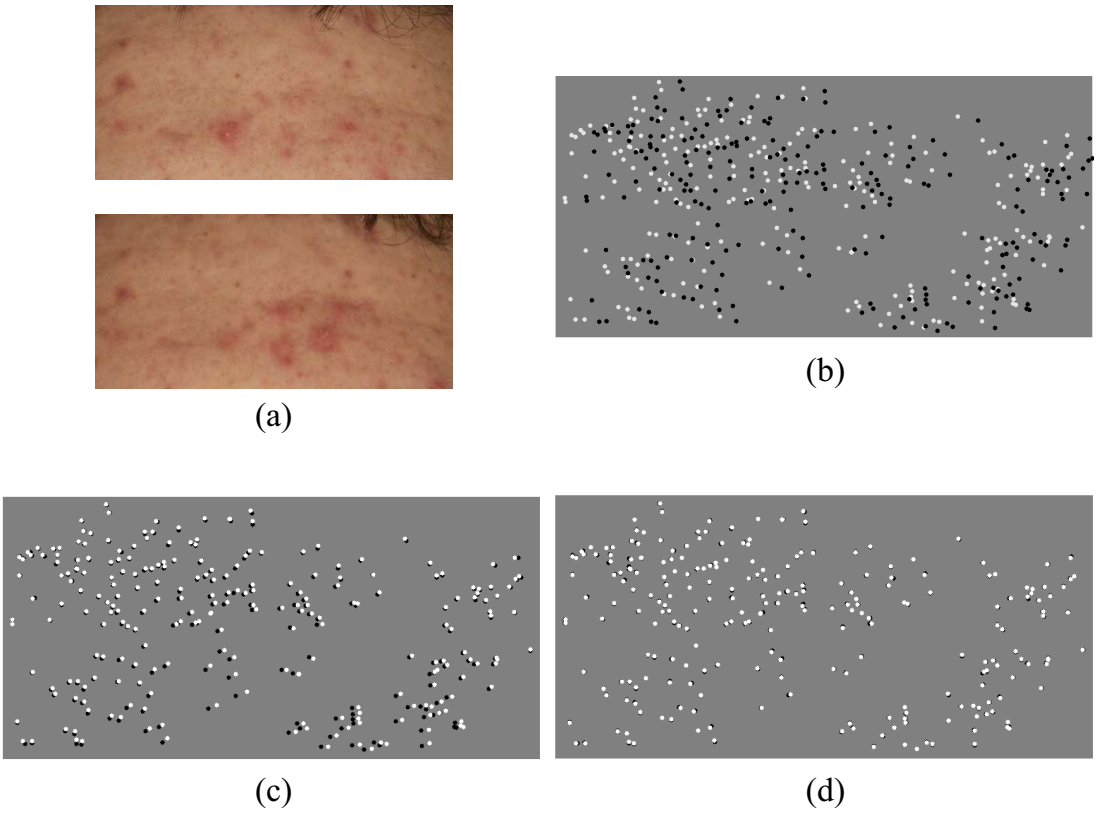


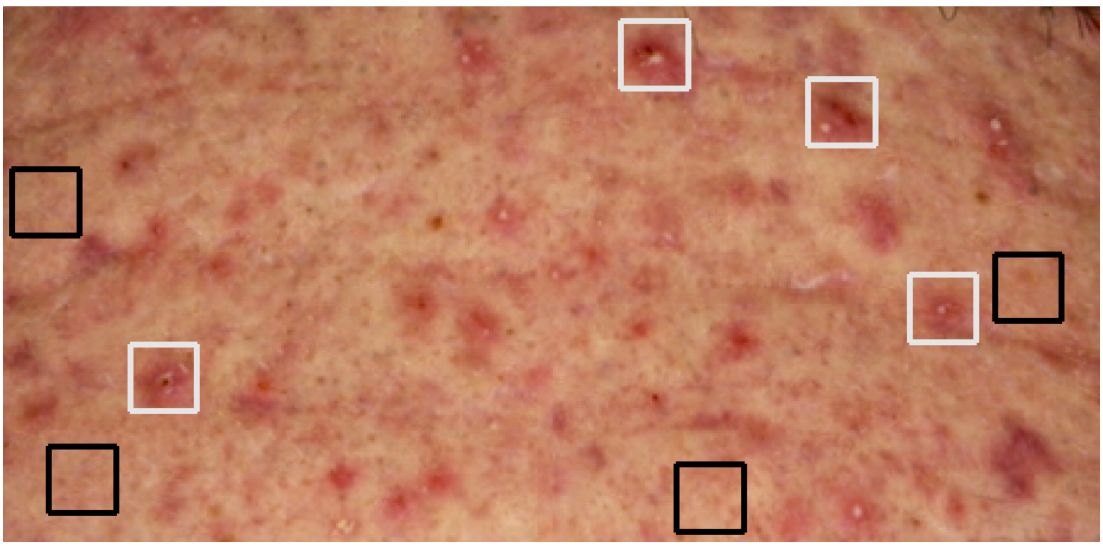
Figure 7.3: (a): Two of the 39 skin images with acne lesions acquired during the three month clinical study. (b): Pixel locations of corresponding feature points in the input images. (c): Pixel locations of corresponding feature points in the globally registered images. (d): Pixel locations of corresponding feature points in the final precisely registered forehead regions.

of micro-level features.

Figure 7.4 shows an example forehead region in the dataset and sample 30×30 acne-like and non-acne regions in the forehead region. The acne-like regions are marked in white and the non-acne regions are marked in black. The first order temporal difference image, $I^d(x, y)$, for a region extracted from a time-lapse image acquired at time t is defined as,

$$I^d(x, y) = I'(x, y) - I(x, y), \tag{7.3}$$

where $I'(x, y)$ is the region image extracted from the time-lapse image acquired at time $t + 1$ and $I(x, y)$ is the region image extracted from the time-lapse image acquired at time t. Figure 7.5 (a) and (b) show the first order temporal difference images for the acne-like regions and non-acne regions in Figure 7.4 (b) and (c); note that the temporal



(a)

(b)

(c)

Figure 7.4: (a): Forehead region with example acne-like regions marked in white and example non-acne regions marked in black. (b): Example acne-like regions marked in the forehead region. (c): Example non-acne regions marked in the forehead region.

difference images for acne-like regions have significantly higher intensity values. The temporal difference captures the variation with time of acne-like regions; therefore we select the mean intensity values, m_R, m_G, and m_B in the red, green, and blue channels of the first order temporal difference image as the first three components of feature vector. The mean intensity value, m_R, in the red channel is defined as,

$$m_R = \frac{1}{N} \sum_{x,y} I_R^d(x, y), \tag{7.4}$$

where N is the number of pixels in the region and $I_R^d(x, y)$ is the intensity of the red channel at pixel location (x, y) in the first order temporal image. m_G and m_B are

(a)

(b)

Figure 7.5: (a): Difference images for acne-like regions shown in Figure 7.4 (b). (b): Difference images for non-acne regions shown in Figure 7.4 (c). Note that the intensity values are generally higher in the difference images for acne-like regions.

defined in a similar way. We use spatial standard deviations σ_R, σ_G, and σ_B in the red, green, and blue channels to capture the intensity profile in acne-like regions. The spatial standard deviation σ_R in the red channel is defined as,

$$\sigma_R = \sqrt{\frac{1}{N}\sum_{x,y}(I_R(x,y) - \bar{I}_R)^2}, \tag{7.5}$$

N is the number of pixels in the region, $I_R(x,y)$ is the intensity of the red channel at pixel location (x,y), and $\bar{I}_R$ is the mean intensity of the red channel in the region. Spatial standard deviations σ_G and σ_B in the green and the blue channels are defined in a similar way. Table 7.1 shows the spatial standard deviations in the red, green, and blue channels for the patch images shown in Figure 7.4 (b) and 7.4 (c). The first four lines show the standard deviations for acne-like regions and the last four lines show the standard deviations for non-acne regions. Note that the spatial standard deviations are generally higher in acne-like patch images, which indicates that spatial standard deviation can be used to classify acne-like regions. We select the spatial standard deviations in the red, green, and blue channels as the last three components of the

Spatial Standard Deviations

Red	Green	Blue
7.0592	11.6148	9.0345
8.3342	12.1767	9.9389
5.7048	9.8521	7.2917
4.8750	8.9595	6.0826
3.3040	4.1848	4.0793
4.6326	4.5517	4.2714
3.5377	4.0475	4.1374
3.8486	6.8987	5.8250

Table 7.1: First four rows show the standard deviation of intensity values for acne like patches and last four columns show the standard deviation of intensity values for non-acne patches. Note that the standard deviation in acne-like patches is higher than in non-acne patches.

feature vector. The final feature vector $\mathbf{x}$ is defined as $\mathbf{x} = [m_R, m_G, m_B, \sigma_R, \sigma_G, \sigma_B]^T$.

Figure 7.6 shows the feature space representation of regions extracted from the chin region. In Figure 7.6, each plot shows two components of the six dimensional feature space. The points corresponding to acne-like regions are shown in white and the points corresponding to non-acne regions are shown in black. Note that in the feature space, the acne-like regions are reasonably well separated from non-acne regions even in the two dimensional projection, and the separating boundary can be modeled using a hyperplane. We learn the separating hyperplane between acne and non-acne regions in the six dimensional feature space using the logistic regression classifier [23]. We train the classifier on 100 30×30 acne/non-acne patches from the chin region, and test on 250 30×30 acne/non-acne patches from the forehead regions. Table 7.2 summarizes the classification results for the training stage, and table 7.3 summarizes the classification results for the testing stage. In the testing stage, we obtained an overall 89.2% success rate in classifying acne-like/non-acne regions. The high success rate in the testing stage implies that local temporal and spatial variations we can be used to successfully classify acne-like regions.

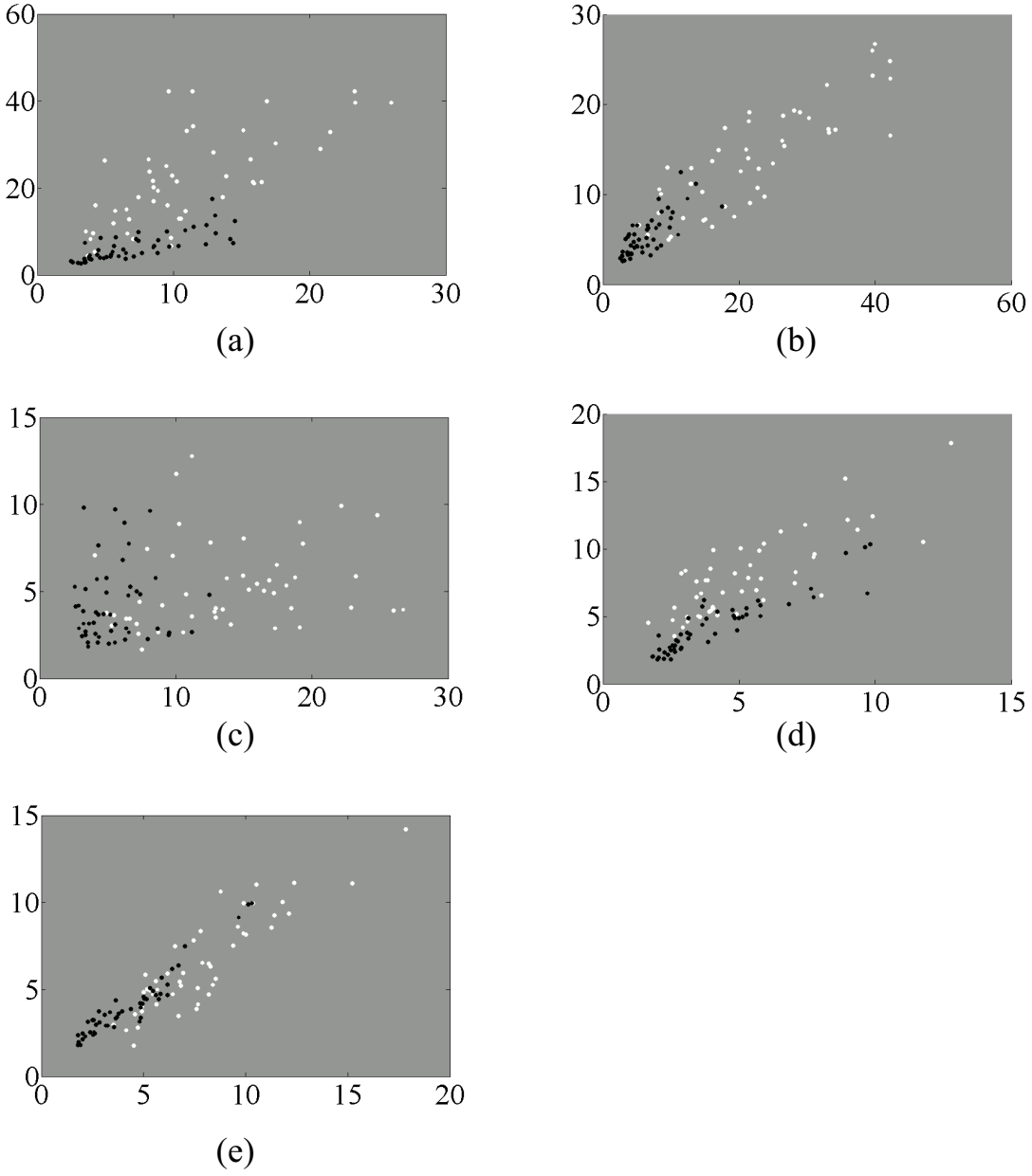


Figure 7.6: Training set feature vectors. In each plot two components of the feature vectors are plotted together. (a, b): Temporal mean of red-green and green-blue channels. (c): Temporal mean of blue channel and spatial standard deviation of red channel. (d, e): Spatial standard deviation of red-green and green-blue channels. White points correspond to acne-like regions and black points correspond to non-acne regions.

Training Stage Results

Total regions: 50 (acne-like) + 50 (non-acne) = 100
Correctly classified regions: 97/100 = 97%
Correctly classified acne-like regions: 49/50 = 98%
Correctly classified non-acne regions: 48/50 = 96%

Table 7.2: Classification results show that the classifier was successfully trained. The classifier was tested on a separate test set, and the test stage results are shown in Table 7.3.

Testing Stage Results

Total regions: 125 (acne-like) + 125 (non-acne) = 250
Correctly classified regions: 223/250 = 89.2%
Correctly classified acne-like regions: 113/125 = 90.4%
Correctly classified non-acne regions: 110/125 = 88%

Table 7.3: The classifier successfully classified acne-like and non-acne patches in the test set.

Chapter 8

Comparison with GDB-ICP Registration Algorithm

We compare the performance of our registration approach with the GDB-ICP [81, 68] feature based registration algorithm. GDB-ICP has shown remarkable ability to register wide variety of image pairs, e.g. retina, melanoma, and brain MRI PD-T1 image pairs, and under very challenging scenarios, e.g. very little overlap, and large difference in orientation and scale. However, we show that the micro-level feature based approach is better suited for registering both time-lapse skin images and multimodal skin images acquired in quick succession. We have used the GDB-ICP algorithm with quadratic model and coarse to fine alignment for time-lapse acne images and full face images with minute misregistration, and homography model without coarse to fine alignment for small quasi-planar patches with minute misregistration. In each case the registration error is measured as the RMS value of the difference between the pixel locations of corresponding micro-level feature points detected using SIFT. All experiments have been run on contrast enhanced images.

8.1 Comparison for Time-lapse Registration

Figure 8.1 (a) and (b) show a pair of 980×664 top part of two time-lapse face images with acne lesions. We register the 700×300 region within the rectangular boundary shown in Figure 8.1 (a) with the corresponding region in Figure 8.1 (b). Table 8.1 shows the registration errors for the GDB-ICP algorithm and the micro-level feature based registration approach. The first row shows the RMS error when the GDB-ICP algorithm was applied to the entire top part, the second row shows the RMS error when GDB-ICP was applied to the skin regions extracted from top part, and the third row shows the RMS error obtained using the micro-level feature based approach. The RMS

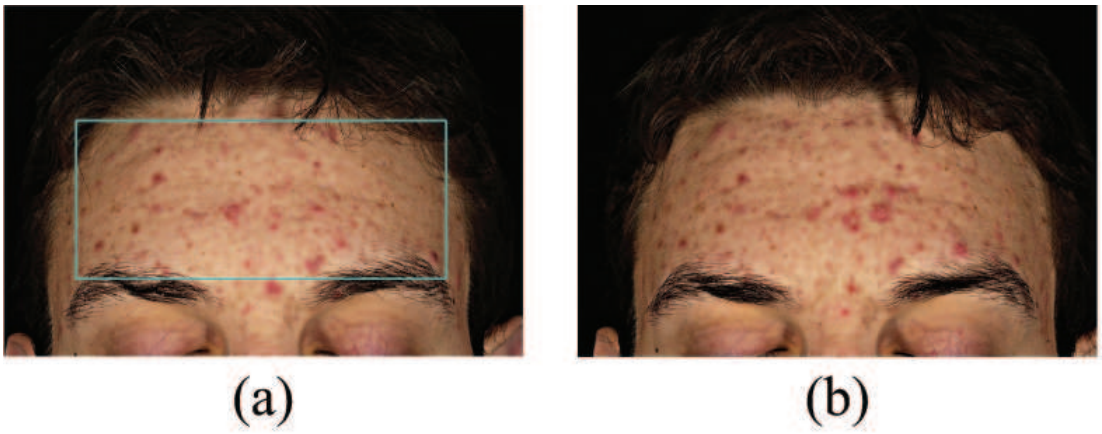


Figure 8.1: (a, b): Top part of two time-lapse face images with acne lesions. We register the skin region within the rectangular boundary shown in (a) with the corresponding region in (b)

Registration Errors for Time Lapse Images

Registration Approach	RMS
GDB-ICP Global	2.99
GDB-ICP Direct	1.98
Micro-Level	0.63

Table 8.1: The registration accuracy of the micro-level feature based approach is higher than that of GDB-ICP.

error obtained using the micro-level feature based approach is significantly lower than the error obtained using the GDB-ICP registration algorithm.

8.2 Comparison for Multimodal Registration

Table 8.2 shows the RMS error when GDB-ICP and the micro-level approach were applied to full face images acquired in quick succession under polarized and visible light. The RMS error is much smaller for the micro-level feature based approach.

Figure 8.2 compares the performance of the GDP-ICP algorithm and the micro-level feature based registration approach on 50 sets of 400×400 quasi-planar parallel-polarized and visual patch images extracted from 2000×2400 full face images. Figure 8.2 (a) shows the RMS error for the GDB-ICP registration algorithm in black points and the RMS for the micro-level feature based approach as white points. In each set

RMS Errors Between Full Face Polarized Images and Corresponding Visual Image

Images	Method	RMS	Micro-Level / GDB-ICP
Fig 6.9 (a)	Input	15.43	0.304
	GDB-ICP	3.19	
	Micro-Level	0.75	
Fig 6.13 (b)	Input	5.85	0.46
	GDB-ICP	1.84	
	Micro-Level	0.79	
Fig 6.13 (a)	Input	10.16	0.23
	GDB-ICP	3.77	
	Micro-Level	0.86	

Table 8.2: The registration accuracy of the micro-level feature based approach is higher than that of GDB-ICP.

the RMS error for the micro-level feature based approach is less than that of GDB-ICP. Figure 8.2 (b) shows the ratio of time taken by the micro-level feature based approach to the time taken by the GDB-ICP registration approach. In each set, the micro-level feature based approach required less than 20% of the time required by the GDB-ICP algorithm to register the images. GDB-ICP requires additional steps of estimating face and corner features at multiple scales, iteratively increase the region within which the estimated model is valid, transform and match corner and face features for each region, and possible model estimation at multiple SIFT keypoints.

Comparisons between GDB-ICP and micro-level feature-based approach shows that micro-level feature-based approach is better suited for skin registration.

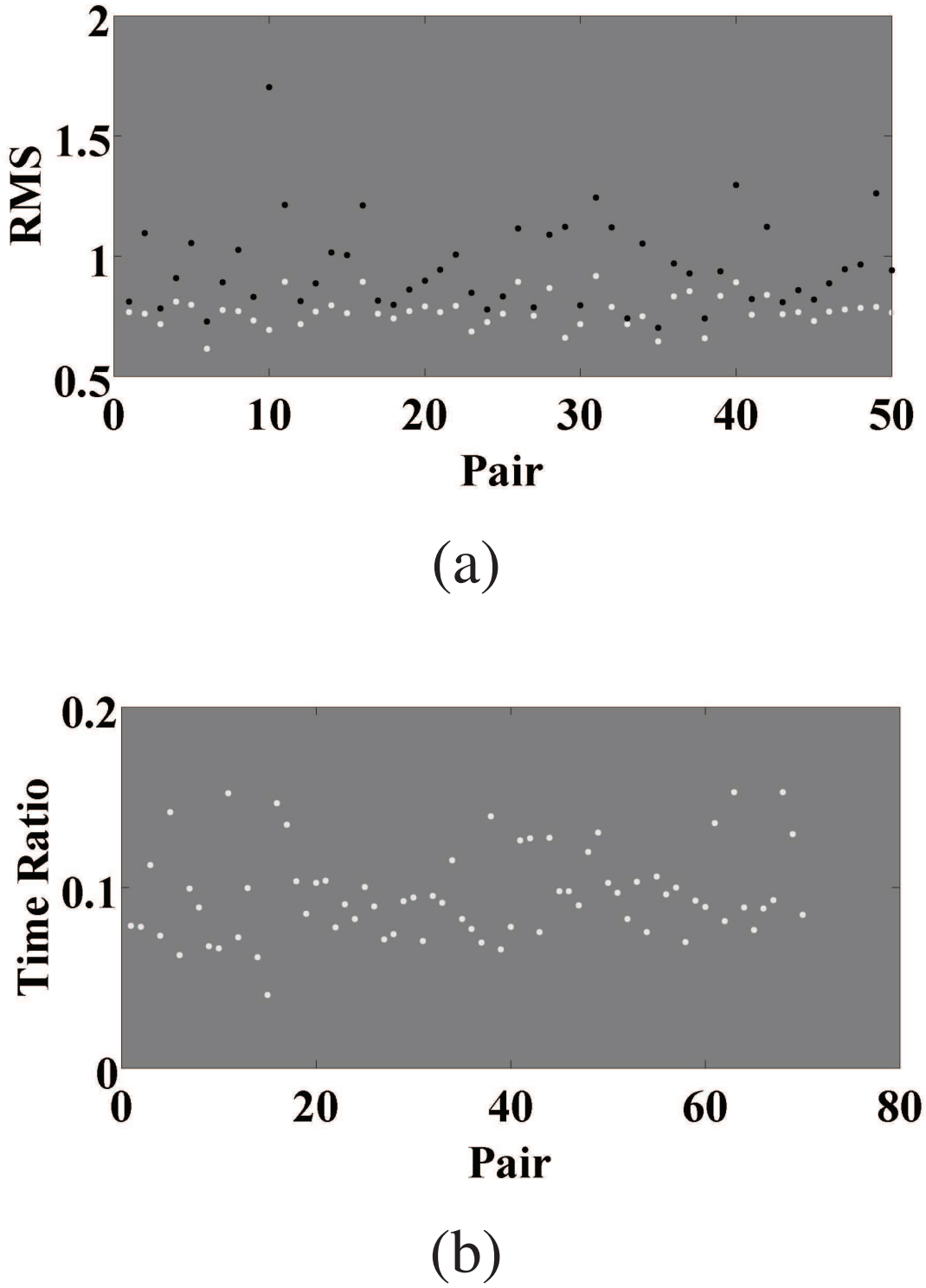


Figure 8.2: (a) RMS error for GDB-ICP and micro-level feature based registration approach. Black points indicate performance of GDB-ICP and white points indicate performance of micro-level feature-based approach. (b) Ratio of registration time required by the micro-level feature based approach to the registration time required by the GDB-ICP algorithm.

Chapter 9

Future Work

We discuss the contributions of the m-BIRCH online clustering algorithm and the micro-level feature based registration algorithm. We discuss the potential extensions and future applications of the two algorithms.

9.1 m-BIRCH: Contributions and Future Extensions

We have introduced a modified version of a classic online clustering algorithm called BIRCH, which has not been widely used in computer vision. The modified version, m-BIRCH, is able to perform automatic data driven parameter selection. The m-BIRCH algorithm uses a multi-tree data summarization method, to effectively handle different density regions in the feature space. We have used m-BIRCH to robustly and incrementally cluster large datasets of features commonly used in computer vision: 840K 384-dimensional color SIFT descriptors, 1.09 million 108-dimensional color patches, 60K outlier corrupted 50-dimensional grayscale patches, and 700K 128-dimensional grayscale SIFT descriptors. We have obtained incremental versions of spectral, GMM, and K-means clustering. To the best of our knowledge, this is the first time that BIRCH has been used with spectral and GMM based clustering. We have shown that m-BIRCH effectively handles outliers.

In future we plan to increase the scale of the experiments. Currently we have conducted all experiments on a desktop computer with four gigabytes of RAM. In recent years super computers and computer clusters are becoming commonly available. A computer cluster containing 100 nodes, each with four gigabytes of RAM, would generate resources to scale the experiments by a factor of 100, and larger gains can be obtained using larger computer clusters. The use of super computers makes possible

the processing of terabytes of data. The high computing power of computer clusters would enable us to run retrieval experiments on much larger number of categories. In non-convex clustering with spectral clustering we would be able to handle much larger matrices by distributing their computations on the different nodes. Handling larger matrices on computer clusters generates the possibility of performing spectral clustering on hundreds of millions of data points, which has never been attempted before. The use of supercomputers for processing large datasets is currently an active area of research in computer vision, and numemrous algorithms are being developed to handle large webscale datasets [74, 21].

9.2 Micro-level Feature Based Registration: Contributions and Future Extensions

We have used a novel combination of micro-level features, interest point detection, and homography estimation to precisely register high resolution multimodal and time-lapse skin images. We have registered multimodal images, with significant difference in appearance, up to pore level features. Such level of precision in skin registration has not been attempted before even with images acquired under same modality. We have shown that registration of multimodal images acquired in quick succession is essential for quantitative dermatology applications like surface component recovery. Our approach precisely registered time-lapse skin images with acne lesions acquired over a three month period. The registered time-lapse set is the first of its kind and clearly brings out the evolution of acne lesions with time. We have used the registered time-lapse set to automatically detect acne-like regions. Computational detection of acne-like regions is an important step toward automating the cumbersome and time consuming process of lesion counting.

We plan to adapt the registration approach by using patches of different sizes. Note that the registration algorithm assumes the patches to be planar. We are aiming to develop an approach to automatically determine for each patch the maximum size for which the planar assumption is valid. We plan to use global information while

determining the homography of each patch. Currently the homography of each patch is determined independently of the homography of other patches. However, the patches do not get transformed independently. We aim to take into account the transformation of remaining patches while determining the homography of each patch. Further, while accounting for global information we aim to assign weights depending on the accuracy of homography estimation and distance from the current patch.

We plan to expand the experiments on automatic lesion detection. Currently the experiments have been run on a single subject and the results have been evaluated on a distinct test set. In future we plan to train a classifier using multiple subjects. We plan to evaluate the results by stepping in the image and detecting the acne-like regions. We plan to use information from all modalities, instead of just the cross-polarized modality, to train the classifier.

We plan to incorporate our work in a tele-dermatology [16] system where remote images are obtained at a remote site or at home and the dermatologist receives a registered sequence for viewing and assessing temporal change. The dermatologist can use the precisely registered sequence to detect even the most minute changes in appearance. The registered sequence also provides a permanent record regarding the evolution of disease and the efficacy of the prescribed treatment.

References

[1] R. Anderson. Polarized light examination and photography of the skin. *Archives of Dermatology*, 127(7):1000–1005, 1991.

[2] B. Awwad, S. Hasan, and J. Q. Gan. Sequential em for unsupervised adaptive gaussian mixture model based classifier. In *Machine Learning and Data Mining in Pattern Recognition*, pages 96–106, 2009.

[3] E. J. Bae, S. H. Seo, Y. C. Kye, and H. H. Ahn. A quantitative assessment of the human skin surface using polarized light digital photography and its dermatologic significance. *Skin Research and Technology*, 16(3):270–274, 2010.

[4] Y. Bae, J. S. Nelson, and B. Jung. Multimodal facial color imaging modality for objective analysis of skin lesions. *Journal of Biomedical Optics*, 13(6), 2008.

[5] S. Baker and I. Matthews. Lucas–kanade 20 years on: A unifying framework. *International Journal of Computer Vision*, 56(3):221–255, 2004.

[6] D. Blei and M. Jordan. Variational inference for dirichlet process mixtures. *Bayesian Analysis*, 1(1):121–144, 2006.

[7] A. Bosch, A. Zisserman, and X. Muñoz. Scene classification using a hybrid generative/discriminative approach. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 30(4):712–727, 2008.

[8] K. Burbeck and Nadjm-Tehrani. Adwice-anomaly detection with real-time incremental clustering. In *Information security and cryptology*, pages 46–51, 2004.

[9] T. Butz and J.-P. Thiran. Affine registration with feature space mutual information. In *International Conference on Medical Image Computing and Computer-Assisted Intervention*, pages 549–556, 2001.

[10] T. Butz and J.-P. Thiran. Multi-view face alignment using direct appearance models. In *International Conference on Automatic Face and Gesture Recognition*, pages 324–329, 2002.

[11] L. O. Callaghan, N. M. A. M. Erson, S. Guha, and R. Motwani. Streaming data algorithms for high quality clustering. In *18^{th}s International Conference on Data Engineering*, pages 685–695, 2002.

[12] D. Capel and A. Zisserman. Automated mosaicing with super-resolution zoom. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 885–891, 1998.

[13] J. Christiansen, P. Holm, and F. Reymann. Treatment of acne vulgaris with the retinoic acid derivative ro 11-1430. a controlled clinical trial against retinoic acid. *Dermatologica*, 153(3):172–176, 1976.

[14] O. Chum, J. Phiblin, J. Sivic, M. Isard, and A. Zisserman. Total recall: Automatic query expansion with a generative feature model for object retrieval. In *International Conference on Computer Vision*, pages 1–8, 2007.

[15] O. G. Cula, K. J. Dana, F. P. Murphy, and B. K. Rao. Bidirectional imaging and modeling of skin texture. *IEEE Transactions on Biomedical Engineering*, 51(12):2148–2159, 2004.

[16] C. Ebner, E. Wurm, B. Binder, H. Kittler, G. Lozzi, C. Massone, G. Gabler, R. Hofmann-Wellenhof, and P. Soyer. Mobile teledermatology: A feasibility study of 58 subjects using mobile phones. *Jounal of Telemedicine and Telecare*, pages 2–7, 2008.

[17] F. Ercal, A. Chawla, W. V. Stoecker, H.-C. Lee, and R. H. Moss. Neural network diagnosis of malignant melanoma from color images. *IEEE Transactions on Biomedical Engineering*, 41(9):837–845, 1994.

[18] M. A. Fischler and R. C. Bolles. Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. *Communications of the ACM*, 24(6):381–395, 1981.

[19] R. Gomes, M. Welling, and P. Perona. Incremental learning of nonparametric bayesian mixture models. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 1–8, 2008.

[20] B. Han, B. Jung, J. S. Nelson, and E.-H. Choi. Analysis of facial sebum distribution using a digital fluorescent imaging system. *Journal of Biomedical Optics*, 12(1), 2007.

[21] Z. Harchaoui, M. Douze, M. Paulin, M. Dudik, and J. Malick. Large-scale image classification with lifted coordinate descent. In *First International Workshop on Large Scale Visual Recognition and Retrieval*, 2012.

[22] R. Hartley and A. Zisserman. *Multiple View Geometry in Computer Vision*. Cambridge University Press, 2000.

[23] T. Hastie, R. Tibshirani, and J. Friedman. *The elements of statistical learning: data mining, inference,and prediction*. Springer, 2001.

[24] T. Hastie, R. Tibshirani, and J. Friedman. *The Elements of Statistical Learning*. Springer, second edition, 2009.

[25] W. Hongcharu, C. R. Taylor, Y. Chang, D. Aghassi, K. Suthamjariya, and R. R. Anderson. Topical ala-photodynamic therapy for the treatment of acne vulgaris. *Journal of Investigative Dermatology*, 115(2):183–192, 2000.

[26] S. L. Jacques, J. C. Ramella-Roman, and K. Lee. Imaging superficial tissues with polarized light. *Lasers in Surgery and Medicine*, 26(2):119–129, 2000.

[27] S. L. Jacques, J. C. Ramella-Roman, and K. Lee. Imaging superficial tissues with polarized light. *Lasers in Surgery and Medicine*, 26(2):119–129, 2000.

[28] S. L. Jacques, J. C. Ramella-Roman, and K. Lee. Imaging skin pathology with polarized light. *Journal of Biomedical Optics*, 7(3):278–523, 2002.

[29] H. Jegou, M. Douze, and C. Schmid. Hamming embedding and weak geometric consistency for large scale image search. In *European Conference on Computer Vision*, pages 304–317, 2008.

[30] H. Jégou, F. Perronnin, M. Douze, J. Sánchez, P. Pérez, and C. Schmid. Aggregating local image descriptors into compact codes. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 34(9):1704–1716, 2012.

[31] F. Jurie and B. Triggs. Creating efficient codebooks for visual recognition. In *International Conference on Computer Vision*, pages 604–610, 2005.

[32] I.-S. Kang, T. wan Kim, and K.-J. Li. A spatial data mining method by delaunay triangulation. In *5^{th} ACM International Workshop on Advances in Geographic Information Systems*, pages 35–39, 1997.

[33] N. Kollias. *Bioengineering of the Skin*, chapter 7, pages 95–104. CRC Press, 1997.

[34] N. Kollias and G. N. Stamatas. Optical non-invasive approaches for diagnosis of skin diseases. *Journal of Investigative Dermatology*, (7):64–75, 2002.

[35] K. Kurihara, M. Welling, and N. Vlassis. Accelerated variational dirichlet process mixtures. *Advances in Neural Information Processing Systems*, pages 761–768, 2006.

[36] F. Lauer and C. Schnorr. Spectral clustering of linear subspaces for motion segmentation. In *International Conference on Computer Vision*, pages 678–685, 2009.

[37] Y. LeCun and C. Cortes. The mnist database of handwritten digits. `http://yann.lecun.com/exdb/mnist`.

[38] J. Lee, S. Kwak, B. Han, and S. Choi. Online video segmentation by bayesian split-merge clustering. In *European Conference on Computer Vision*, pages 856–869, 2012.

[39] D. Lowe. Distinctive image features from scale-invariant keypoints. *International Journal of Computer Vision*, 60(2):91–110, 2004.

[40] S. Madan and K. J. Dana. m-birch: An online clustering approach for computer vision applications. *Submitted to IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2012.

[41] S. Madan, K. J. Dana, and O. G. Cula. Quasiconvex alignment of multimodal skin images for quantitative dermatology. In *IEEE Computer Society Workshop on Mathematical Methods in Biomedical Analysis*, pages 117–124, 2009.

[42] S. Madan, K. J. Dana, and O. G. Cula. Learning based detection of acne-like regions using time-lapse features. In *IEEE Signal Processing in Medicine and Biology Symposium*, 2011.

[43] S. Madan, K. J. Dana, and O. G. Cula. Multimodal and time-lapse skin registration. *Submitted to Image and Vision Computing*, 2013.

[44] F. Maes, A. Collignon, D. Vandermeulen, G. Marchal, and P. Suetens. Multimodality image registration by maximization of mutual information. *IEEE Transations on Medical Imaging*, 16(2):187–198, 1997.

[45] J. Mairal, F. Bach, J. Ponce, and G. Sapiro. Online detection of unusual events in videos via dynamic sparse coding. In *International Conference on Computer Vision*, pages 604–610, 2005.

[46] J. Mairal, F. Bach, J. Ponce, and G. Sapiro. Online dictionary learning for sparse coding. In *International Conference on Computer Vision*, pages 604–610, 2005.

[47] J. Mairal, F. Bach, J. Ponce, and G. Sapiro. Online dictionary learning for sparse coding. In *International Conference on Machine Learning*, pages 689–696, 2009.

[48] D. Martin, C. Fowlkes, D. Tal, and J. Malik. A database of human segmented natural images and its application to evaluating segmentation algorithms and measuring ecological statistics. In *International Conference on Computer Vision*, pages 416–423, 2001.

[49] D. Mattes, D. R. Haynor, H. Vesselle, T. K. Lewellen, and W. Eubank. Pet-ct image registration in the chest using free-form deformations. *IEEE Transations on Medical Imaging*, 22(1):120–128, 2003.

[50] K. Mikolajczyk and C. Schmid. Scale and affine invariant interest point detectors. *International Journal of Computer Vision*, 60(2):91–110, 2004.

[51] T. K. Moon. The expectation-maximization algorithm. *IEEE Signal Processing Magazine*, 13(6):47–60, 1996.

[52] J. A. Muccini, N. Kollias, S. B. Phillips, R. R. Anderson, A. J. Sober, M. J. Stiller, and L. A. Drake. Polarized light photography in the evaluation of photoaging. *Journal of the American Academy of Dermatology*, 33(5):765–769, 1995.

[53] S. Nayar, X. Fang, and T. Boult. Removal of specularities using color and polarization. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 583–590, 1993.

[54] A. Y. Ng, M. I. Jordan, and Y. Weiss. On spectral clustering: Analysis and an algorithm. In *Advances in Neural Information Processing Systems*, pages 849–856, 2001.

[55] D. Ni, Y. Qu, X. Yang, Y. P. Chui, T.-T. Wong, S. S. Ho, and P. A. Heng. Volumetric ultrasound panorama based on 3d sift. In *International Conference on Medical Image Computing and Computer-Assisted Intervention*, pages 52–60, 2008.

[56] D. Nistér and H. Stewénius. Scalable recognition with a vocabulary tree. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 2161–2168, 2006.

[57] J. Nocedal and S. Wright. *Numerical Optimization*. Springer, second edition, 2006.

[58] A. Oliva and A. Torralba. Modeling the shape of the scene: A holistic representation of the spatial envelope. *International Journal of Computer Vision*, 42(3):145–175, 2001.

[59] J. Philbin, O. Chum, M. Isard, J. Sivic, and A. Zisserman. Object retrieval with large vocabularies and fast spatial matching. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 1–8, 2007.

[60] J. P. W. Pluim, J. B. A. Maintz, and M. A. Viergever. Mutual information based registration of medical images: a survey. *IEEE Transations on Medical Imaging*, 22(8):986–1004, 2003.

[61] W. H. Press, S. A. Teukolsky, W. T. Vetterling, and B. P. Flannery. *Numerical recipes in C++*. Cambridge University Press, 2002.

[62] J. C. Ramella-Roman, K. Lee, S. A. Prahl, and S. L. Jacques. Design, testing, and clinical studies of a handheld polarized light camera. *Journal of Biomedical Optics*, 9(6):1305–1310, 2004.

[63] N. Rao and F. Porikli. A clustering approach to update online dictionary learning. In *IEEE International Conference on Acoustics, Speech, and Signal Processing*, 2012.

[64] E. Rizova and A. Kligman. New photographic techniques for clinical evaluation of acne. *Journal of the European Academy of Dermatology and Venereology*, 15(3):13–18, 2001.

[65] D. Rueckert, L. I. Sonoda, C. Hayes, D. L. G. Hill, M. O. Leach, and D. J. Hawkes. Nonrigid registration using free-form deformations: application to breast mr images. *IEEE Transations on Medical Imaging*, 18(8):712–721, 1999.

[66] J. M. Saragih, S. Lucey, and J. F. Cohn. Deformable model fitting by regularized landmark mean-shift. *International Journal of Computer Vision*, 91(2):200–215, 2011.

[67] S. Sigurdsson, P. A. Philipsen, L. K. Hansen, J. Larsen, M. Gniadecka, and H. C. Wulf. Detection of skin cancer by classification of raman spectra. *IEEE Transactions on Biomedical Engineering*, 51(10):1784–1793, 2004.

[68] C. V. Stewart, C.-L. Tsai, and B. Roysam. The dual-bootstrap iterative closest point algorithm with application to retinal image registration. *IEEE transactions on medical imaging*, 22(11):1379–1394, 2003.

[69] R. C. N. Studinski and I. A. Vitkin. Methodology for examining polarized light interactions with tissues and tissue like media in the exact backscattering direction. *Journal of Biomedical Optics*, 5(3):330–337, 2000.

[70] R. Szeliski. Image alignment and stitching: a tutorial. *Foundations and Trends in Computer Graphics and Vision*, 2(1):1–104, 2006.

[71] E. L. Tanzi and T. S. Alster. Comparison of a 1450-nm diode laser and a 1320-nm nd:yag laser in the treatment of atrophic facial scars: a prospective clinical and histologic study. *Dermatologic Surgery*, 30(2):152–157, 2004.

[72] R. Tron and R. Vidal. A benchmark for the comparison of 3-d motion segmentation algorithms. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 1–8, 2007.

[73] P. Turcot and D. Lowe. Better matching with fewer features: The selection of useful features in large database recognition problems. In *ICCV Workshop on Emergent Issues in Large Amounts of Visual Data*, pages 2109–2116, 2009.

[74] M. Villegas and R. Paredes. A k-nn approach for scalable image annotation using general web data. In *First International Workshop on Large Scale Visual Recognition and Retrieval*, 2012.

[75] C. Wang, D. Blei, and F.-F. Li. Simultaneous image classification and annotation. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 1903–1910, 2009.

[76] J. A. Witkowski and L. C. Parish. The assessment of acne: an evaluation of grading and lesion counting in the measurement of acne. *Clinics in Dermatology*, 22(5):394–397, 2004.

[77] J. A. Witkowski and H. M. Simons. Objective evaluation of demethylchlortetracycline hydrocloride in treatment of acne. *The journal of the American Medical Association*, 196(5):397–400, 1966.

[78] J. Wu and J. M. Rehg. Centrist: A visual descriptor for scene categorization. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 33(8):1489–1501, 2011.

[79] J. Xiao, T. Moriyama, T. Kanade, and J. Cohn. Robust full-motion recovery of head by dynamic templates and re-registration techniques. *International Journal of Imaging Systems and Technology*, 13:85–94, 2003.

[80] J. Yan and M. Pollefeys. A general framework for motion segmentation: Independent, articulated, rigid, non-rigid, degenerate, and non-degenerate. In *European Conference on Computer Vision*, pages 94–106, 2006.

[81] G. Yang, C. V. Stewart, M. Sofka, and C.-L. Tsai. Registration of challenging image pairs: initialization, estimation, and decision. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 29(11):1973–1989, 2007.

[82] T. Zhang, R. Ramakrishnan, and M. Leivny. Birch: An efficient data clustering method for very large databases. In *ACM SIGMOD International Conference on Management of Data*, volume 25, pages 103–114, 1996.

[83] T. Zhang, R. Ramakrishnan, and M. Leivny. Birch: A new data clustering algorithm and its applications. *Data Mining and Knowledge Discovery*, 1(2):141–182, 1997.

[84] B. Zitova and J. Flusser. Image registration methods: a survey. *Image and Vision Computing*, 21(11):977–1000, 2003.