REFLECTANCE AND TEXTURE ENCODING FOR MATERIAL RECOGNITION AND SYNTHESIS

by

HANG ZHANG

A dissertation submitted to the

School of Graduate Studies

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 Dana

and approved by

_____

_____

_____

_____

_____

New Brunswick, New Jersey

October, 2017

ABSTRACT OF THE DISSERTATION

Reflectance and Texture Encoding for Material Recognition and Synthesis

by Hang Zhang

Dissertation Director: Dr. Kristin Dana

Material recognition plays an important role for a machine to understand and interact with the world. For example, an autonomous vehicle can use material recognition to determine whether the terrain is asphalt, grass, gravel, ice or snow in order to optimize the mechanical control and a robot can easily grab an object with proper pressure if the object's composition is known. This thesis is dedicated to developing compact and robust material and texture representations for fast visual recognition and synthesis.

Color and geometry are not a full measure of the richness of visual appearance. Reflectance describes the characteristics of light interaction with a surface, which depends on microscopic and mesoscopic composition. This thesis explores how reflectance can reveal the material categories and physical properties. We build representations that capture the spatial and angular variation of reflectance. These multi-layer deep learning representations provide invariance to intra-class variations for recognition. We also develop representations that capture sufficient detail for synthesis. In particular, this thesis develop the following methods:

1. Reflectance Hashing: Reflectance is challenging to measure and use for recognizing materials due to its high-dimensionality. In this work, we bypass the use of a gonioreflectometer by using a novel one-shot reflectance camera based on a parabolic

mirror design. The pixel coordinates of these reflectance disks correspond to the surface viewing angles. The reflectance has class-specific structure and angular gradients computed in this reflectance space reveal the material class. These reflectance disks encode discriminative information for efficient and accurate material recognition. We introduce a framework called reflectance hashing that models the reflectance disks with dictionary learning and binary hashing. We demonstrate the effectiveness of reflectance hashing for material recognition with a number of real-world materials.

2. Deep Reflectance Code: We introduce a framework that enables prediction of actual friction values for surfaces using one-shot reflectance measurements. This work is a first of its kind vision-based friction estimation. We develop a novel representation for reflectance disks that capture partial BRDF measurements instantaneously. Our method of deep reflectance codes combines CNN features and fisher vector pooling with optimal binary embedding to create codes that have sufficient discriminatory power and have important properties of illumination and spatial invariance. The experimental results demonstrate that reflectance can play a new role in deciphering the underlying physical properties of real-world scenes.

3. Texture Encoding Network: We propose a Deep Texture Encoding Network (DeepTEN) with a novel Encoding Layer integrated on top of convolutional layers, which ports the entire dictionary learning and encoding pipeline into a single model. The features, dictionaries and the encoding representation for the classifier are all learned simultaneously. The representation is orderless and therefore is particularly useful for material and texture recognition. The Encoding Layer generalizes robust residual encoders such as VLAD and Fisher Vectors, and has the property of discarding domain specific information which makes the learned convolutional features easier to transfer. The experimental results show superior performance as compared to state-of-the-art methods using gold-standard databases such as MINC-2500, Flickr Material Database, KTH-TIPS-2b, and two recent databases 4D-Light-Field-Material and GTOS.

4. Real-time Texture Synthesis We introduce a Multi-style Generative Network (MSG-Net) with a novel Inspiration Layer, which retains the functionality of

optimization-based approaches and has the fast speed of feed-forward networks. The proposed Inspiration Layer explicitly matches the feature statistics with the target styles at run time, which dramatically improves versatility of existing generative network, so that multiple styles can be realized within one network. The proposed MSG-Net matches image styles at multiple scales and puts the computational burden into the training. The learned generator is a compact feed-forward network that runs in real-time after training.

Acknowledgements

I would like to express my deep gratitude to my advisor Professor Kristin Dana, for her great support in the past four years. She brings me to the challenging and promising fields of computer vision, computational photography and deep learning, and directs me through it. I especially enjoy the freedom to work with her and benefit from her rigorous academic attitude, and perseverance in solving challenging problems. During my PhD, Kristin tirelessly developed my skills in research, writing, and communication. I feel greatly fortunate to be her student.

I would also like to thank Dr. Ko Nishino and Dr. Urs Muller. Ko's sharp insight and creative ideas often guides my research turns into good papers. I spent a wonderful summer working with Urs during my summer internship. He brought me into a team of talent researchers and always gave me encouragement.

Many thanks also go to my colleagues in Rutgers University, including Eric Wengrowski, Jia Xue, Dr. Parneet Kaur, Dr. Wenjia Yuan, He Zhang, Xing Di, Han Zhang, Elle Rosen, Hansi Liu, Fanpeng Kong, Li Zhu, Wenhan Zhang, Kelvin Zhiyang Wang, Iris Mengyang Guo, Zhenhua Jia and Haixin Xi. I learned a lot in many aspects from them during these years so that I would never forget the enriched and joyful days together.

My special appreciation is dedicated to my wife Stacy Xiaoqian Yang and my parents Jiabao Zhang and Yuhong Wang whom I share happiness, frustration, pain and success. Hardly can I survive my PhD life without their support and encouragement. Finally, I would like to thank all committee members (Prof. Kristin Dana, Prof. Vishal Patel, Prof. Yanyong Zhang, Prof. Ko Nishino and Dr. Urs Muller) for attending my PhD thesis defense, all staffs in the Electrical and Computer Engineer Department at Rutgers University and all the other people who helped and discussed the thesis.

This work is dedicated to my family

who love, encourage, and urge me all along.

Table of Contents

List of Tables

List of Figures

Chapter 1

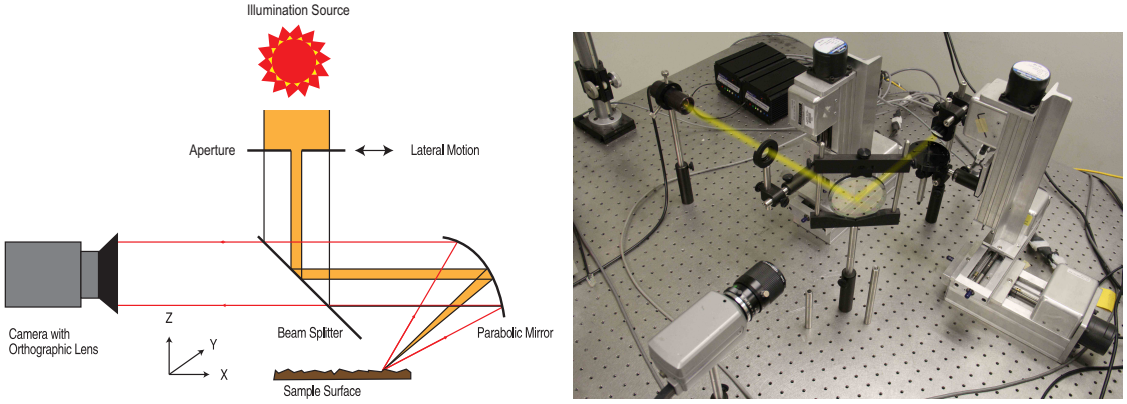
Introduction

1.1 Motivation

In the current era of artificial intelligence and computer vision, there is a rapid growth of novel applications and real-life products, such as intelligent home assistants, camera surveillance and autonomous driving. Visual understanding uses shape information as a strong cue, however it can be enhanced with the fine-grained information of material properties. Material recognition plays an important role for a machine to perceive the world and interact with the world. For example, a mobile robot or autonomous vehicle can automatically switch the mechanical control using material recognition to recognize the terrain type such as asphalt, grass, gravel, ice or snow. An indoor robot can use the proper pressure to grab an object if it knows what material the object is made of. The potential applications are promising in many areas such as home intelligent assistants, autonomous vehicles, and advanced manufacturing.

Reflectance is a property describing how light interacts with the material surface, which provides important cues about the surface microscopic and mesoscopic composition. The challenges of reflectance measurement are the high-dimensionality of the data and the time-consuming measurement procedures using traditional gonioreflectometer device that captures the material with different combinations of the viewing and illumination angles along a hemisphere.

Computational models of reflectance representation are desired to be descriptive for visual appearance and have invariance to illumination and surface tilt. The measured appearance of a surface changes under different illumination angles. The illumination direction affects the position of the specular peak within the reflectance measurements, necessitating translation invariance within the representation. The distribution



(a) Diagram of Measurement Device (b) Prototype of Measurement Device

Figure 1.1: One-shot In-field Reflectance Measurement. Reflectance disks are obtained by viewing a single point under multiple viewing directions using a concave parabolic mirror viewed by a telecentric lens.

of patterns in the reflectance measurements is similar to desired properties of texture representations. Recent work adopts robust visual encoders such as VLAD [4] and Fisher Vector [5] for texture representation, but those encoders are usually very high-dimensional. The challenge is to preserve discriminant power of the robust encoders and build a compact representation for fast retrieval.

Classic texture and material recognition typically builds on off-the-shelf representations. First hand-engineered features are extracted such as SIFT [6] or filter bank responses [7–10]. Followed by a dictionary learning and then the feature are extracted using Bag-of-Words (BoWs) representation [11–14], Finally, a classifier such as SVM is learned for classification. In recent work, hand-engineered features and filter banks are replaced by pre-trained CNNs and BoWs are replaced by the robust residual encoders such as VLAD [4] and its probabilistic version Fisher Vector (FV) [5]. For example, Cimpoi *et al.* [1] assembles different features (SIFT, CNNs) with different encoders (VLAD, FV) and have achieved state-of-the-art results. However, these methods (both classic and recent work) consists several self-contained algorithmic components (feature extraction, dictionary learning, encoding, classifier training). Consequently, the features and the encoders are fixed once built, so that the feature learning does not benefit from labeled data. Therefore, an orderless feature pooling layer is desirable for end-to-end learning to bridge the gap of robust texture encoders and deep learning.

In addition to recognition, texture modeling can be used for texture synthesis and style transfer. Style transfer can be regarded as reconstructing or synthesizing texture based on image semantic content. Prior work employing a pre-trained CNN as feature statistics for texture modeling achieves great success in texture synthesis and style transfer [15]. Recent work trains a feed-forward generative network to approximate the optimization process that transforms the image into a target style in real-time and puts the computational burden into training process [16–18]. However, these approaches require training separate networks for each different style, which extremely limits the scalability.

Despite the success of artistic style transfer in recent work, facial style transfer remains challenging due to the requirement of photo-realism and semantic consistency. Because human vision is very sensitive to facial irregularities and even small distortions can make a face look unrealistic. In addition, texture transfer can easily bring minor content information from the style target, which we want to avoid in identity preserving facial transfer. For this, an approach is desired for facial texture transfer which simultaneously preserves photo-realism and the human facial identity.

1.2 Thesis Overview

In this thesis, we are interested in angular and spatial models of reflectance. For angular variations, we study partial BRDF as measured using reflectance disks [19]. We encode the angular variation of reflectance in a framework called deep reflectance codes [20]. By using reflectance disks in a CNN framework, angular gradients are computed as a signature for recognition. Spatial variation of reflectance gives rise to image texture. We separately address spatial variations of texture by creating texture networks for both recognition (Deep-Ten) [21] and synthesis (MSG-Net, Face-Tex) [22, 23]. We explore combining both angular and spatial variations of reflectance in real world scenes using GTOS [24], a database of ground terrain that capture angular and spatial variations. Differential angular imaging is a deep learning approach to provide a combination of angular and spatial information for recognition.

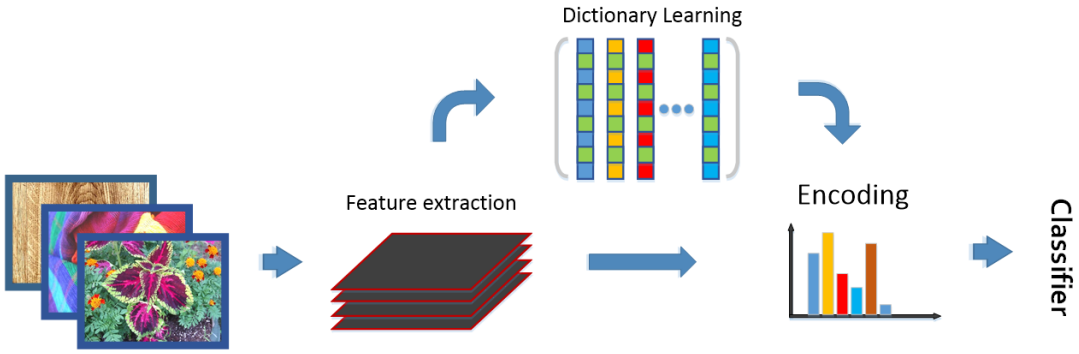


Figure 1.2: Pipelines of classic computer vision approaches. Given the input images, the local visual appearance is extracted using hang-engineered features (SIFT or filter bank responses). A dictionary is then learned off-line using unsupervised grouping such as K-means. An encoder (such as BoWs or Fisher Vector) is built on top which describes the distribution of the features and output a fixed-length representations for classification.

First, to address the reflectance measuring difficulty, we employ a novel one-shot reflectance camera (as shown in Figure 1.1) based on a parabolic mirror design [25] in Chapter 2 which bypasses the use of a gonioreflectometer. The output of this camera is a reflectance disk, a dense sampling of the surface reflectance of the material projected into a single image. We refer to the images captured using proposed one-shot in-field measurement as *reflectance disk*. We go beyond the task of material recognition and explore how reflectance can be used to estimate tactile properties of the surface: determine the touch from appearance in Chapter 3. We build the first approach for *friction-from-reflectance*. For this, we collect a first-of-its-kind database of friction coefficients and reflectance disks measured for 137 surfaces that can be grouped into 21 classes. To tackle the difficulty of compact representations with spacial invariance, we introduce a novel material representation, which we refer to as deep reflectance codes (DRC), which builds on top of the concept of state-of-the-art texture descriptor and the fast speed of binary hash codes. Deep reflectance codes achieve necessary descriptive power and invariance by applying binary embedding to fisher vector convolutional neural nets (FV-CNN) [1].

To leverage the flexibility of classic approaches and the power of feature learning using CNNs in Chapter 4, we introduce an orderless feature pooling layer on top of convolutional layers, which we refer to as the *Encoding Layer* that integrates the entire

dictionary learning and residual encoding pipeline into a single layer for CNNs. The Encoding Layer has three main properties. (1) The Encoding Layer generalizes classic robust residual encoders such as VLAD and Fisher Vector. This representation is orderless and describes the feature distribution, which is suitable for material and texture recognition. (2) The Encoding Layer acts as a global pooling layer integrated on top of convolutional layers, which can accept arbitrary input sizes and output a fixed-length representation. By allowing arbitrary size images, the Encoding Layer makes the deep learning framework more flexible and our experiments show that recognition performance is often improved with multi-size training. In addition, (3) the Encoding Layer learns an inherent dictionary and the encoding representation which is likely to carry domain-specific information and therefore is suitable for transferring pre-trained features. In this work, we transfer CNNs from object categorization (ImageNet [26]) to material recognition. Since the network is trained end-to-end as a regression progress, the convolutional features learned together with the Encoding Layer on top are easier to transfer (likely to be domain-independent).

To increase the scalability of the generative networks for real-time texture synthesis and style transfer, we introduce a Multi-style Generative Network (MSG-Net) with a novel Inspiration Layer in Chapter 5, which retains the functionality of optimization-based approaches and has the fast speed of feed-forward networks. The proposed Inspiration Layer explicitly matches the feature statistics with the target styles at run time, which dramatically improves versatility of existing generative network, so that multiple styles can be realized within one network. The proposed MSG-Net matches image styles at multiple scales and puts the computational burden into the training. The learned generator is a compact feed-forward network that runs in real-time after training. Comparing to previous work, the proposed network can achieve fast style transfer with at least comparable quality using a single network.

We also study photo-realistic facial texture transfer [23] in Chapter 6. We address the challenging problem of transferring face texture from a style face image to a content face image in a photorealistic manner simultaneously preserving the identity of the original content image. Our approach augments the prior work of MRF-CNN with a

novel facial semantic regularization incorporating a face prior regularization smoothly suppressing the changes around facial meso-structures ((e.g eyes, nose and mouth) and a facial structure loss function which implicitly preserves the facial structure so that face texture can be transferred without changing the original identity.

Chapter 2

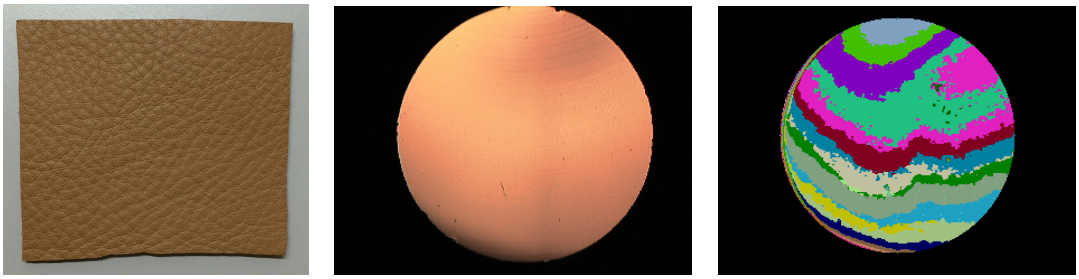
Reflectance Hashing for Material Recognition

Overview

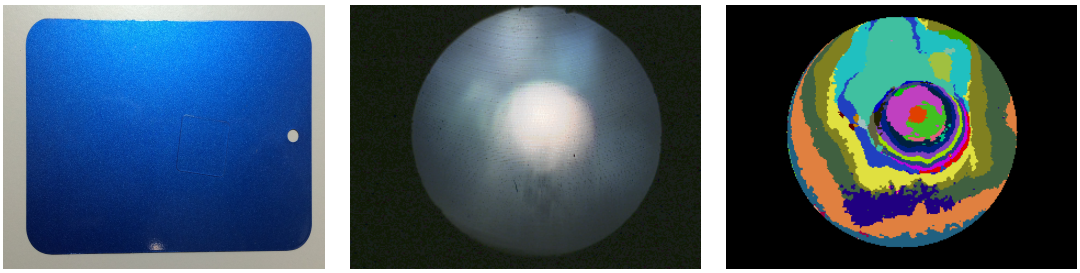
This chapter on Reflectance Hashing for Material Recognition is based on our paper [19]. We introduce a novel method for using reflectance to identify materials. Reflectance offers a unique signature of the material but is challenging to measure and use for recognizing materials due to its high-dimensionality. In this work, one-shot reflectance of a material surface which we refer to as a *reflectance disk* is capturing using a unique optical camera. The pixel coordinates of these reflectance disks correspond to the surface viewing angles. The reflectance has class-specific stucture and angular gradients computed in this reflectance space reveal the material class. These reflectance disks encode discriminative information for efficient and accurate material recognition. We introduce a framework called reflectance hashing that models the reflectance disks with dictionary learning and binary hashing. We demonstrate the effectiveness of reflectance hashing for material recognition with a number of real-world materials.

2.1 Background

Color and geometry are not a full measure of the richness of visual appearance. Material composition of a physical surface point determines the characteristics of light interaction and the reflection to an observer. In the everyday real world there are a vast number of materials that are useful to discern including concrete, metal, plastic, velvet, satin, asphalt, carpet, tile, skin, hair, wood, and marble. A computational method for identifying these materials has important implications in developing new algorithms and new technologies for a broad set of application domains. For example, a mobile robot or autonomous automobile can use material recognition to determine whether



(a) Leather Surface (b) Reflectance Disk (c) Texton Map



(d) Auto Paint Surface (e) Reflectance Disk (f) Texton Map

Figure 2.1: Reflectance disks provide a quick snapshot of the intrinsic reflectance of a surface point. Gradients of the reflectance space are captured with textons and provide a signature for material recognition.

the terrain is asphalt, grass, gravel, ice or snow in order to optimize mechanical control. An indoor mobile robot can distinguish among wood, tile, or carpet for cleaning tasks. The material composition of objects can be tagged for an e-commerce inventory or for characterizing multi-composite 3D printed objects. The potential applications are limitless in areas such as robotics, digital architecture, human-computer interaction, intelligent vehicles, and advanced manufacturing. Furthermore, just as computer vision algorithms now use depth sensors directly from RGB-D cameras, material sensors can have foundational importance in nearly all vision algorithms including segmentation, feature matching, scene recognition, image-based rendering, context-based search, object recognition, and motion estimation. Our approach uses reflectance for material recognition because of the advantages of having an intrinsic optical signature for the surface. However, we bypass the use of a gonioreflectometer by using a novel one-shot reflectance camera based on a parabolic mirror design. The output of this camera is a

reflectance disk, a dense sampling of the surface reflectance of the material projected into a single image as shown for two example surfaces in Figure 2.1. Each reflectance disk measures surface properties for a single point and can capture complex appearance such as the iridescence of a peacock feather as illustrated in Figure 2.2. We then use this convenient reflectance measurement as the discriminative characteristic for recognizing the material class. We address the issue of high dimensionality using a novel application of binary hash codes to encode reflectance information in an efficient yet discriminative representation. The key idea is to obtain sufficient sampling of the reflectance with enough discriminative power and reduce its representation size so that it can be effectively used for probing the material.

We present a database of reflectance disks comprised of twenty different diverse material classes including wood, velvet, ceramic and automotive paint with 10 spot measurements per surface and with three different surface instances per class. Measurements include three on-axis illumination angles $(-10°, 0°, 10°)$ and ten random spot measurements over the surface. Each spot measurement is a reflectance disk composed of a dense sampling of viewing angles totaling thousands of reflectance angles per disk. The database of 3600 images or reflectance disks is made publicly available. For recognition, we combine binary hash coding and texton boosting for a new framework called *reflectance hashing* for efficient and accurate recognition of materials. We compare reflectance hashing with texton boosting for the task of recognizing materials from reflectance disks.

2.2 Related work

Prior methods for material recognition use two distinct approaches. One approach assesses material identity using reflectance as an intrinsic property of the surface [27–31]. Another main approach identifies material labels using the appearance of the surface within the real world scene [32–34]. Using reflectance instead of scene appearance has the advantage that reflectance is a direct measurement of the material characteristics, instead of its phenomenological appearance [35]. Reflectance is mostly unique to the material, whereas the appearance is the convoluted end result of the interaction of all

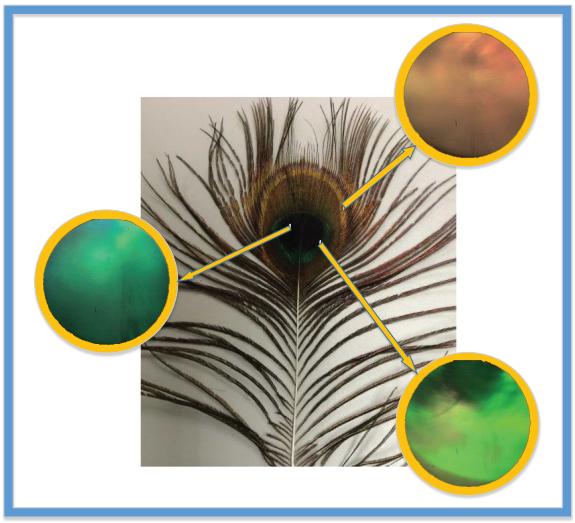


Figure 2.2: Reflectance disks provide a snapshot of the reflectance for spot samples on the surface. For this example of a peacock feather, iridescence causes a large variation of intensity with viewing direction as revealed in the three reflectance disks.

the intrinsic and extrinsic factors and thus is more difficult to decipher.

A challenge in using reflectance for material recognition is that measurements are typically difficult and cumbersome. For example, most methods require knowledge of the scene like geometry [27,36] or illumination [37,38]. Other methods require lab-based measurements of the BRDF (bidirectional reflectance distribution function) or BRDF slices such as light domes for illumination patterns [39]. Acquiring full BRDF requires gonioreflectometers that are comprised of multiple cameras and light sources covering the hemisphere of possible directions using geodesic domes or robotics as in [40–46]. Surface appearance in terms of relief texture is captured by the GelSight camera [47] using three-color photometric stereo by imprinting the surface geometry on an elastomer surface with Lambertian skin. With this device very fine geometric texture can be recovered but surface appearance is lost. An additional challenge of using reflectance for measurements is the high dimensionality, reflectance as a function of illumination and viewing direction especially if densely sampled can lead to thousands or more samples per surface point.

Specialized cameras have been developed in prior research to obtain reflectance

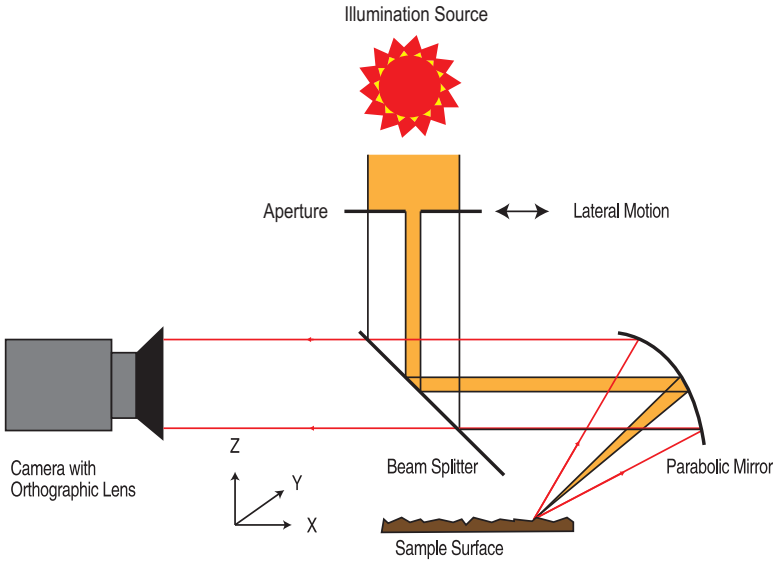


Figure 2.3: Schematic of the mirror-based camera. Reflectance disks are obtained by viewing a single point under multiple viewing directions using a concave parabolic mirror viewed by a telecentric lens.

measurements of surfaces efficiently. The mirror-based camera illustrated in Figure 2.8 is one such a device [25, 48]. This camera views an image of a single surface point by using a concave parabolic mirror with the focus coincident with the target point. In this manner, the camera records a multiview image of a surface point where each pixel records a different angle. Multiple views from mirror-based optics is also achieved with a kaleidoscope-like configuration [49]. However, the viewing angles captured in this device are discrete and sparse. Because our approach relies uses gradients in angular space a dense sampling of reflectance is needed and we use the parabolic mirror-based camera as a reflectance sensor.

In prior work, recognition of standard scene images is typically accomplished using image features that capture the spatial variation of image intensity. For example, image filtering with a set of filter banks followed by grouping the outputs into characteristic textons has been used for image texture [8, 29, 50, 51], object recognition [52, 53] and image segmentation [54]. Similarly, we encode the discriminative optical characteristics of materials captured in the reflectance disks with a texton-based representation. Texton methods for the general problem of scene and image recognition have been improved

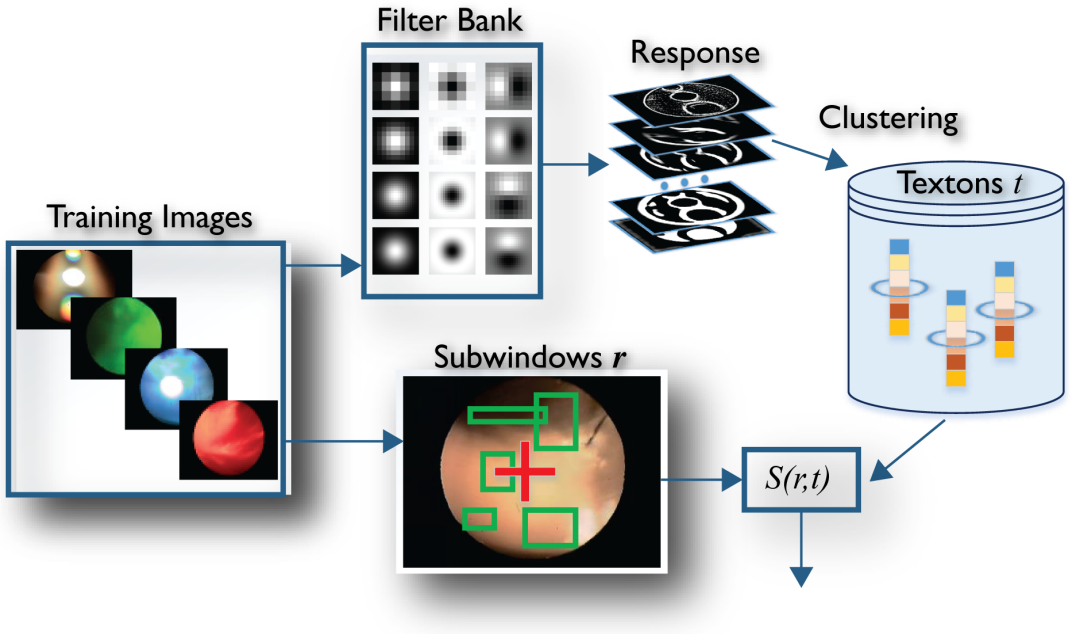


Figure 2.4: Gradients on the Reflectance Disk Extracted with Textons: Images are filtered by a filter bank comprised of Gaussian, Laplacian and oriented gradient filters. The 24×1 responses vectors are clustered to form visual words or textons. A random subsampling of rectangular subregions of the reflectance space defines regions of interest r. The feature of interest is $S(r,t)$, the count of pixels identified as texton t in region r.

significantly by incorporating boosting as in *Texton-boost* [53] and *Joint-boost* [55]. We incorporate the utility of boosting to identify a discriminative and compact representation.

The goal of efficient recognition of images for large scale search has led to numerous methods for binary hash codes [56–60]. A common approach in recent work [56, 61–63] is learning binary codes where the Hamming distance approximates the kernel distance between appearance descriptors. This method is used when the kernel distance is known to give good performance but is expensive to compute and store. Inspired by the work in image search, we develop *reflectance hashing* with binary codes that allow matching of measured reflectance disks with those in the labeled training set. The binary codes are learned using the match metric obtained from the texton-based descriptor we present in Section 2.3. We use a suite of state-of-the-art hashing techniques applied to our

measured reflectance disk dataset to evaluate recognition performance in terms of speed and recognition accuracy.

2.3 Methods

2.3.1 Reflectance Disk Measurements

We use the *texture camera* approach introduced in [25, 48, 64, 65] for measuring reflectance of surface points, The measurement camera mainly consists of a parabolic mirror, a CCD camera, a movable aperture and a beam splitter. The imaging components and their arrangement are illustrated in Figure 3.3. A parabolic mirror section is fixed so that its focus is at the surface point to be measured. The illumination source is a collimated beam of light parallel to the global plane of the surface and passing through a movable aperture. The angle of the incident ray reaching the surface is determined by the intersection point with the mirror. Therefore, the illumination direction is determined by the aperture position. Light reflected from the surface point is reflected by the mirror to a set of parallel rays directed through a beam splitter to the camera. Each pixel of this camera image corresponds to a viewing direction of the surface point. Therefore, the recorded image is an observation of a single point on the surface but from a dense sampling of viewing directions. Multiple illumination directions can be measured by planar motions of the illumination aperture. A key advantage of this approach is the potential for a handheld reflectance sensor where the mirror and light source are attached to a handheld camera.

We refer to the camera's output images as *reflectance disks* since they depict circular regions from on-axis projections of a paraboloid. Examples of these reflectance disks are shown in Figure 2.1 for two surfaces. These examples show the variation of the surface reflectance with viewing angle. Reflectance disks have an intensity variation that corresponds to angular change with the viewing direction instead of spatial change on the surface. However, the reflectance disks can be interpreted as images and filtered to find characteristic gradients. This approach of using *angular gradients in reflectance space* is a novel contribution of our work. Our goal is to represent the measurements in

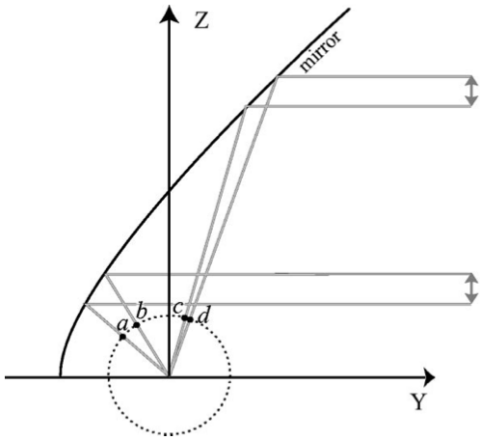
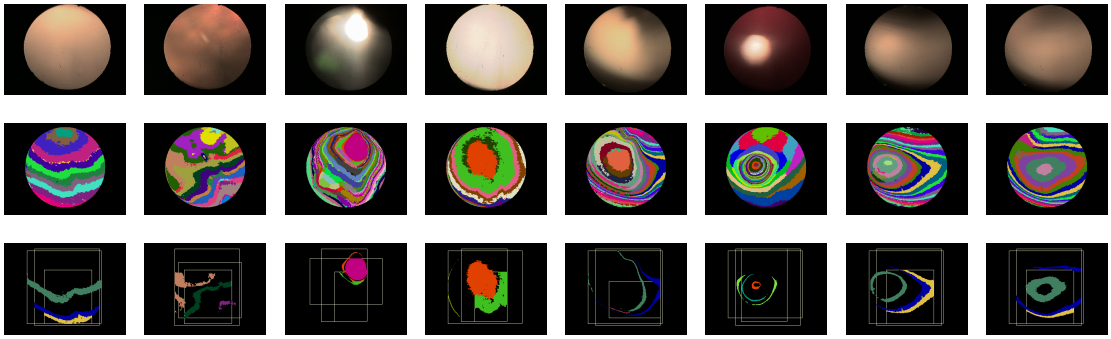


Figure 2.5: Gradients on the reflectance disk are computed using image filtering. However, the angular resolution varies over the surface as shown above. The angle at *ab* is different from *cd*. Therefore differences (or filtering) computed with a uniform spacing on the reflectance disk have a non-uniform mapping to angles.

an efficient and meaningful way that supports material recognition.

2.3.2 Textons on the Reflectance Disk

We find characteristic patterns of intensity in a reflectance disk by computing *spatial change* in the reflectance disk via image filtering. Since spatial coordinates of the reflectance disk map to angular coordinates of reflectance, filtering is a convenient way to compute angular gradients of the reflectance. A common method of quantifying intensity patterns is using *textons*. Textons are computed by first filtering the image with a set of gradient filters at different orientations and scales. We employ the same filter banks as in [29], comprised of Gaussian low-pass filters at four scales, Laplacian filters at four scales and eight gradient filters at different orientations and scales for a total of 24 filters (ranging in size from 7×7 to 25×25). The resulting 24×1 responses at each pixel are clustered using K-means clustering and a dictionary of visual words called textons is created. The underlying assumption is that the local intensity variation can be captured with the finite set of characteristic filter responses that are the centers of each of the K clusters. Each pixel in the reflectance disk is assigned a texton label and the resulting label image is called a texton map (as illustrated in Row 2 of Figure



(a) cloth (b) feather (c) glossy metal (d) paper (e) sponge (f) wax paper (g) wood (h) wood

Figure 2.6: Visualizing the reflectance-layout filters.The first row shows the reflectance disks for different materials. The second row shows the corresponding texton maps. The third row shows three of the dominant region-texton (r, t) combinations chosen for each class.

2.6) .

We use textons as a reflectance feature to provide a dense, per-pixel description of reflectance variation. Methods to detect key-points are useful for scene images, but reflectance disks do not typically have specific key-points of interest.

For texton computation, we use gradient filters that approximate the derivative operator with a discrete spatial difference filter. This discrete approximation of the gradient computes change over a non-infinitesimal distance, typically the difference of neighboring pixels. The distance between these neighboring pixels is constant over the reflectance disk, i.e. the same gradient filter is applied over the entire disk image. However, because of the mapping from spatial coordinates to angular coordinates by the parabolic mirror, the angular distance is not constant. This situation is illustrated in Figure 2.5 where the cone angle at the surface is shown for two different locations on the parabolic mirror. A constant distance d on the projected disk leads to two different cone angles. Consequently, the resolution for the angular gradient varies as a function of disk coordinates.

The equation of the parabolic mirror surface is given by,

$$y + F = \frac{z^2 + x^2}{4F}, \tag{2.1}$$

where the y-axis is the optical axis of the mirror aligned with the camera optical axis, the $x - z$ plane is aligned with the disk-shaped projection of the mirror surface on camera's image plane, and F is the mirror focal length (12.7 mm). Define the cone angle α as the set of angles subtended by arc ab or cd in Figure 2.5, where d is taken as 2 pixel units. From this equation, the variation of the cone angle α as a function of the vertical spatial coordinate z on the reflectance disk is derived in [48] as

$$\alpha(z) = \arctan \frac{z-1}{\frac{(z-1)^2}{4F} - F} - \arctan \frac{z+1}{\frac{(z+1)^2}{4F} - F}. \tag{2.2}$$

Since the geometry of the mirror is known, the spatial filters can be adaptive to ensure a uniform cone angle over the disk. However, the spatially invariant gradient filter provides a discriminative representation and uniformity of angular resolution is not required.

Reflectance-Layout Filter

Textons are a local measure of image structure, and texton histograms are used to represent a region. The histogram representation has the advantage of invariance to shifts of structures. Reflectance disks show the structure of the reflectance field, but often the distribution within that structure is the characteristic property. Histogram representations have the drawback that crucial spatial information is lost. We follow the texton boost framework [52, 53, 55] to encode the characteristic spatial structure of the reflectance disk by defining a set of randomly shaped rectangular regions over the disk. For a given texton t and region r, $S(r,t)$ is the count of pixels labelled with texton t that are located within region r as illustrated in Figure 2.4. We use this response as a key feature for describing the reflectance disks. This response is referred to as the texture-layout filter response when applied to ordinary images [53]. For material recognition using reflectance disks, we use the term *reflectance-layout filter* to indicate that the representation captures the angular layout of reflectance gradients. For computing reflectance-layout filter responses, the number of particular textons in a region, the texton count is calculated using a soft-weighting for the 8 nearest cluster centers. Soft-weighted textons as in [53, 54, 66] allows each pixel to be associated with

8 nearest clusters instead of a single texton label. This method alleviates the problem of two similar responses assigned to different texton labels.

Each reflectance layout filter characterizes a specific region in reflectance space where the change of reflectance as a function of viewing angle is a descriptive feature for the material. There are many possible combinations of (r, t) as illustrated in Figure 2.4 and we want to choose the ones that are most discriminative. The boosting approach in [52,55] is used to identify the most discriminative (r, t) combinations. After boosting, the typical numbers of reflectance layout filters is approximately 700.

2.3.3 Joint Boosting

The algorithm for *Joint Boosting* [55] provides feature selection where the feature is the reflectance-layout filter $S(r, t)$ specified by a particular (r, t) pair that indicates the number of times visual words t occurs in region r. The approach builds a strong learner iteratively by adding the strongest weak learner which is based on the dominant feature at each iteration, allowing the weak learner to help classify a subset of classes. At each step m, the method maximally reduces the weighted squared error by choosing the strongest weak learner $h_m(I_i, c)$, updating the strong learner as: $H(I_i, c) := H(I_i, c) + h_m(I_i, c)$, where I_i is the input image and c is the class. The weighted square error is given by

$$J_{wse} = \sum_{c=1}^{C} \sum_{i=1}^{N} w_i^c (z_i^c - h_m(I_i, c))^2, \tag{2.3}$$

where C is number of classes, and N is the number of training samples, $w_i^c = e^{-z_i^c H(I_i,c)}$ defines the weights for class c and sample i that emphasize the incorrectly classified samples for that iteration, z_i^c is the membership label (± 1) for class c. The optimal solution that gives the selected learner at step m has the form

$$h_m(I_i, c) = \begin{cases} a\delta(S(r, t) > \theta) + b & \text{, if } c \in T_m \\ k^c & \text{, otherwise} \end{cases} \tag{2.4}$$

with the parameters (a, b, k^c, θ) given by the minimization of Equation 2.3, $\delta(.)$ is an indicator function (0 or 1), $S(r, t)$ is the count or texton t in region r, T_m is the class subset that share this feature. For the classes sharing the feature ($c_i \in T_m$), the weak

learner gives $h(I_i, c) \in \{a + b, b\}$ determined by the comparison of reflectance-layout filter response $S(r, t)$ with the threshold θ. For the classes not sharing the feature, there is a constant k^c different for each class c that avoids the asymmetry caused by the number of positive and negative samples for each class.

This method chooses a dominant weak learner $h_m(I_i, c)$ at each iteration to minimize the error function. For each selected learner, finding the class subset T that maximally reduces the error on weighted training set is expensive. Instead of searching all possible $2^C - 1$ combinations, a greedy search heuristic [55] is used to reduce the complexity to $O(C^2)$. We first find the best feature for each single class, and pick the class that has the best error reduction. Then we add the class that has best joint error reduction until we go through all the classes. Finally, we select the set that provides the largest overall error reduction. It is also time consuming to go through all the features at each iteration, so only a fraction $\tau \ll 1$ reflectance-layout filters are examined, randomly chosen in each iteration. The bottom row of Figure 2.6 shows the selected dominant features (reflectance-layout filters) for a number of different reflectance disks

2.3.4 Reflectance Hashing

Boosting allows the selection of learners $h_m(I_i, c)$ that specify (r, t) (region, texton) pairs of interest. The selected learners can simply be viewed as the input feature vector for a basic nearest neighbor classification. Nearest neighbors in high dimensions is problematic, but the computational tools of binary hash codes enables efficient and accurate representation. We use the reflectance layout filters directly in a concatenated feature vector with $700 \times c$ component where each component is a region-texton (r, t) pair that is known to be useful in material recognition. There is likely to be redundancy in the (r, t) combinations so the number of components in the feature vector may be reduced accordingly. This high dimensional feature vector is used directly for fast computation by employing binary coding. This efficiency is highly desirable for the embedded hardware implementation.

True class \ Inferred class	cardboard	CD	woven fabric	feather	textured rubber	glossy ceramic	glossy metal	leather	linen	matte metal	automotive paint	non-ferrous metal	paper	plastic	rock	rubber	sponge	velvet	glossy paper	wood
cardboard	**96.1**	0.1					0.2	0.1			0.3	0.1								3.8
CD		**99.1**	0.1																	
woven fabric			**74.1**	0.2				5.5	0.8	0.1			0.7		8.3		1.3	6.9		2.0
feather	0.1			**92.9**				1.9		0.2							3.1			1.7
textured rubber			0.1		**72.8**	0.4			0.9	0.6			3.4		1.2	17.6	0.5	2.5		
glossy ceramic					5.0	**84.1**	2.2			5.1			1.5			0.1			2.0	
glossy metal						1.0	**98.7**	0.2		0.1										
leather	8.2		1.1					**83.7**	1.2			0.2			0.7		0.1			4.8
linen	0.6		5.9	0.2	1.5	1.3	2.4	3.5	**61.9**	2.4	3.1	0.9	0.5	0.5	12.3		0.4	0.3	0.2	2.3
matte metal					3.6	0.2				**94.8**	0.2		0.9			0.2				
automotive paint			0.4		2.0	2.8	1.0		3.3	0.4	**81.7**	3.6			0.6	0.8	1.1	2.4		
non-ferrous metal					0.3		0.8	0.5	1.9	1.5	2.3	**85.3**	0.3	0.3	1.1		0.6			5.2
paper	0.1			0.1	3.9					2.0			**89.8**	1.1		1.3		1.7		
plastic											0.1			**99.4**	0.1		0.1		0.2	
rock	0.2		10.5		2.9			4.2	4.0		0.1		0.1		**66.8**	1.4	4.2	3.9		1.6
rubber					1.1					0.1			2.0		0.1	**96.0**	0.1	0.5		
sponge			2.3	1.7	4.3				0.7						12.0	0.2	**66.1**	9.8		2.7
velvet			1.9	3.2	2.2				0.2	0.4			1.1		0.7	6.4	4.8	**78.6**	0.3	0.1
glossy paper			2.3	0.2		0.1		0.5						0.2					**96.4**	0.2
wood	9.1		1.1	0.1				5.7	2.0		0.5		1.5		3.8		1.4			**74.9**

Figure 2.7: Joint Boosting Classification. Confusion matrix with percentages row-normalized. Overall accuracy is 84.38%.

We use a suite of state-of-the-art approaches for forming binary code words including circulant binary embedding (CBE) [67], bilinear embedding [68], iterative quantization (ITQ) [69], angular quantization-based binary codes(AQBC) [70], spectral hashing (SH) [71], locality sensitive hash(LSH) [72], and locality-sensitive binary codes from shift-invariant kernels (SKLSH) [73]. We refer to this direct encoding and use of the reflectance layout filters as reflectance hashing.

2.4 Experiments

Reflectance Disk Database To evaluate the performance of our reflectance hashing framework, a database of 3600 reflectance disks is collected consisting of the following 20 surface classes: cardboard, CD, cloth, feather, textured rubber, glossy ceramic, glossy metal, leather, linen, mattel metal, automotive painting, non-ferrous metal, paper, plastic, rock, rubber, sponge, velvet, glossy paper and wood. The measurement camera shown in Figure 2.8 is arranged as in Figure 3.3 with a video camera, telecentric lens, light source, beam splitter and parabolic mirror. The database includes 3 instances per class, i.e. three different example surfaces per class, 10 spot samples per surface with 3 illumination directions $(-10^\circ, 0, 10^\circ)$ where 0° is frontal illumination aligned with the surface normal. Additionally, two exposure settings are collected for each reflectance

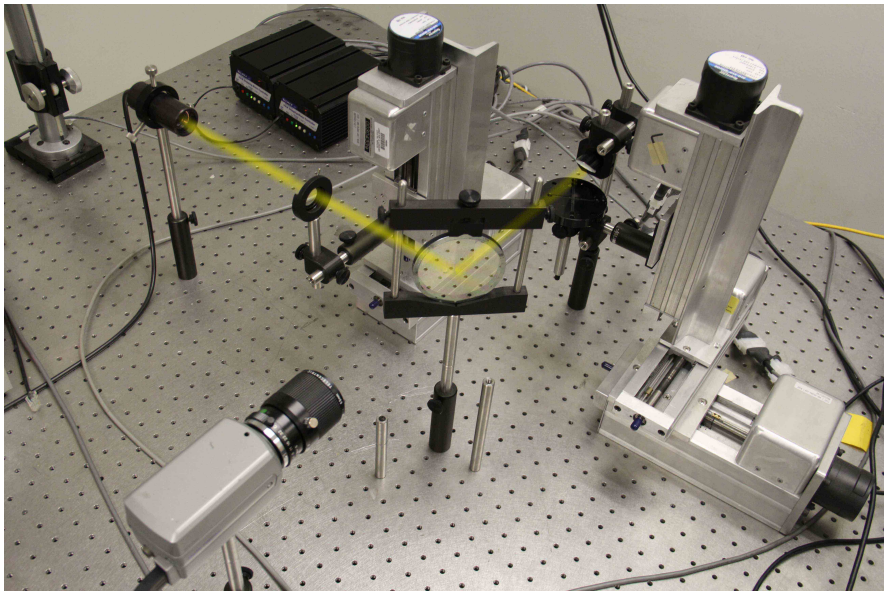


Figure 2.8: Prototype of the camera for reflectance disk capture. The components follow the arrangement in Figure 3.3.

disk. Therefore, the dataset consists of 180 reflectance disks per class, for a total dataset of 3600 reflectance disks each of size 320×240. This reflectance dataset is made publicly available.[1]

Material Recognition We compare the performance of texton boosting with reflectance hashing for material classification. We compared the classification result with a training set of N random selected images for both methods, and test the classification accuracy on a test set of $3600 - N$ random selected images. We vary N from 360 (10%) to 1500. For the large set of training images, both methods perform well; however, the recognition rate of joint boosting decreases significantly, when decreasing the size of training set. Reflectance hashing gives a more stable recognition rate as a function of the training set size as shown in Figure 2.9.

Figure 2.7 shows the confusion matrix from $N = 360$ classification with joint boosting with 700 iterations in the training stage, and mean recognition rate of 84.38%. The parameter settings were $K = 512$ textons and 200 subwindows or regions, with the random feature selection fraction $\tau = 0.01$. Figure 2.10 shows the confusion matrix

[1]Available at http://www.ece.rutgers.edu/vision

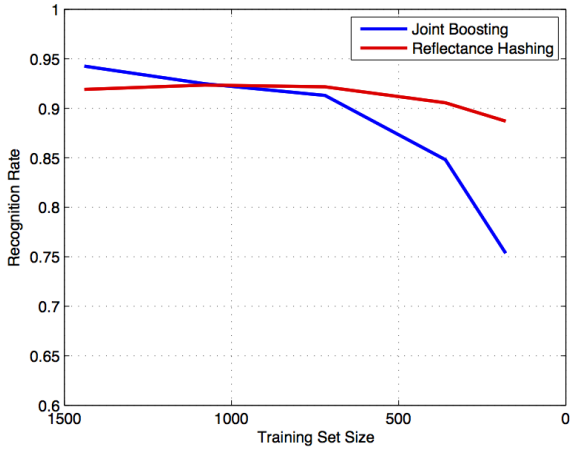


Figure 2.9: Recognition rate as a function of training set size for both boosting and reflectance hashing. Notice the performance of reflectance hashing is high even when the training set number is low.

obtained by our method of reflectance hashing, with the features selected in 700 iterations, and using iterative quantization [69] as the binary embedding method. The overall recognition rate is 92.3%, and several individual class recognition rates are significantly higher than the boosting method of Figure 2.7. For both methods, 5-fold cross validation was performed.

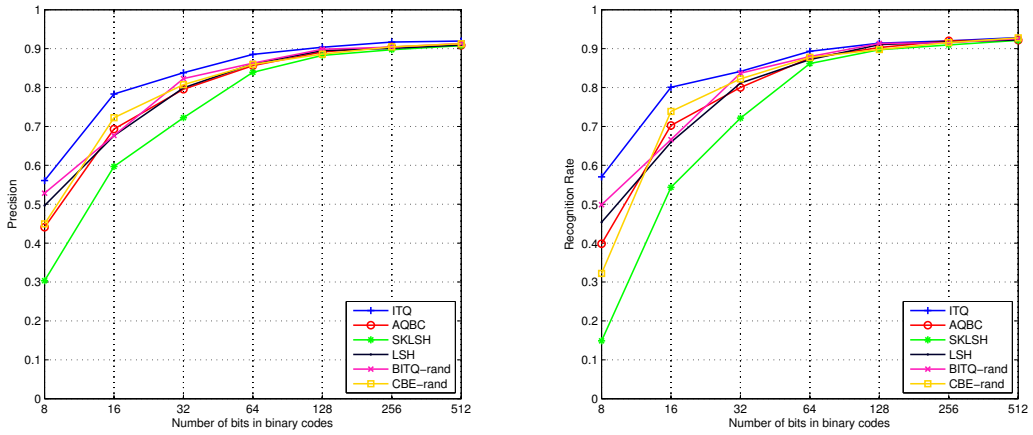
We also make a baseline comparison with texton histogram classification where no boosting is done and no region subwindows are used. The histogram is computed over the entire reflectance disk. The recognition rate with 1500 training images is 79.53% with a max rate of 100% (plastic, glossy paper) and a minimum of 45.6% (sponge) The recognition rate with 360 training images is 41.94% with a max rate of 89.2% (cardboard) and a minimum of 18.3% (linen).

From the empirical results, we see that reflectance hashing provides reliable recognition even for a small training set. This result has important implications for real time material recognition since the approach may support online training with compact training sets.

Evaluation of Binary Codes We compare the performance of recent binary code methods for use in reflectance hashing. These codes are use to represent the response

True class \ Inferred class	cardboard	CD	woven fabric	feather	textured rubber	glossy ceramic	glossy metal	leather	linen	matte metal	automotive paint	non-ferrous metal	paper	plastic	rock	rubber	sponge	velvet	glossy paper	wood
cardboard	**99.6**	0.1						0.2												
CD		**99.8**	0.2																	
woven fabric			**89.9**	0.2				3.3		0.2			0.1		4.5		0.4	1.0		0.4
feather	0.2			**96.1**				1.7		0.1			0.1				1.0	0.4		0.3
textured rubber				0.1	**82.1**	0.2			0.9	0.8	0.2		4.3		1.9	8.0	0.8	0.7		
glossy ceramic					3.0	**95.7**	1.0			0.4										
glossy metal							**99.6**	0.2	0.1											
leather	0.9		0.2					**97.3**	0.2				0.1		1.0					0.2
linen	0.1		4.5	0.5	1.2	0.2	2.9	3.8	**66.7**	2.0	3.7	2.5	1.1	0.7	6.2		0.7	0.7		2.3
matte metal					0.6	0.1	0.1			**98.4**	0.2		0.5							
automotive paint									0.1	0.3	**99.0**	0.6								
non-ferrous metal	0.1						1.2	0.6	1.2	0.4	2.5	**91.6**	0.3		0.4		0.6			1.1
paper					0.5					2.2			**95.2**	0.6		0.4		1.1		
plastic														**99.7**	0.3					
rock			4.8		0.2			2.0	1.7						**84.0**	0.4	1.8	2.0		3.1
rubber					0.1								1.5			**98.1**	0.3			
sponge			2.0	2.6	2.2										4.3	0.4	**82.3**	4.4		1.8
velvet			1.2	0.1	2.6	0.1		0.8		0.1			0.7		0.6	4.3	3.4	**85.6**	0.3	
glossy paper			0.8			0.6								0.1					**98.2**	0.2
wood	7.4		0.5	0.3				5.3					0.7		0.5		0.1			**85.2**

Figure 2.10: Reflectance Hashing Classification. Confusion matrix with percentages row-normalized. Overall accuracy is 92.3%.



(a) Nearest neighbor search precision

(b) Recognition Rate

Figure 2.11: Precision and recognition rate as a function of the number of bits for binary embedding with 10 nearest neighbors.

from the reflectance-layout filters as described in Section 2.3.4. We test the following binary embedding methods: randomized (CBE-rand) version of circulant binary embedding (CBE) [67], randomized (BITQ-rand) version of bilinear embedding [68], iterative quantization (ITQ) [69], angular quantization-based binary codes(AQBC) [70], spectral hashing (SH) [71], locality-sensitive binary codes from shift-invariant kernels (SKLSH) [73], and a baseline method locality sensitive hash (LSH) [72]. CBE and BITQ also have learned versions, but these are more appropriate for higher dimensional data and longer bit code. Recognition is accomplished using a nearest neighbor search with a Hamming distance metric and ten nearest neighbors. From Figure 2.11, we see the binary embedding recognition rate reaches around 90%, when using 128 or 256 bit binary codes. The method of ITQ gives the best results for this material recognition task.

2.5 Summary and Conclusion

We have introduced a novel framework for measurement and recognition of material classes. The approach encodes high dimensional reflectance with a compact binary code. We have compared several existing binary code methods to choose the most appropriate for this material recognition approach. The coding supports discrimination among the classes and can be realized in embedded vision implementations. Our method of reflectance hashing is compared to two popular baseline texton-based methods, boosting and texton histograms for material recognition. The results show excellent performance even for small training sets and provide a novel method based on reflectance for fast sensing and recognition of real-world materials.

Chapter 3

Friction from Reflectance: Deep Reflectance Codes for Predicting Physical Surface Properties from One-Shot In-Field Reflectance

Overview

This chapter on Deep Reflectance Code is based on our paper [20]. Images are the standard input for vision algorithms, but one-shot in-field reflectance measurements are creating new opportunities for recognition and scene understanding. In this work, we address the question of what reflectance can reveal about materials in an efficient manner. We go beyond the question of recognition and labeling and ask the question: What intrinsic physical properties of the surface can be estimated using reflectance? We introduce a framework that enables prediction of actual friction values for surfaces using one-shot reflectance measurements. This work is a first of its kind vision-based friction estimation. We develop a novel representation for reflectance disks that capture partial BRDF measurements instantaneously. Our method of *deep reflectance codes* combines CNN features and fisher vector pooling with optimal binary embedding to create codes that have sufficient discriminatory power and have important properties of illumination and spatial invariance. The experimental results demonstrate that reflectance can play a new role in deciphering the underlying physical properties of real-world scenes.

3.1 Background

Reflectance describes the characteristics of light interaction with a surface, which is uniquely determined by how the surface is made up at a microscopic (e.g., pigments in the surface medium) and mesoscopic scale (e.g., geometric 3D texture). Reflectance

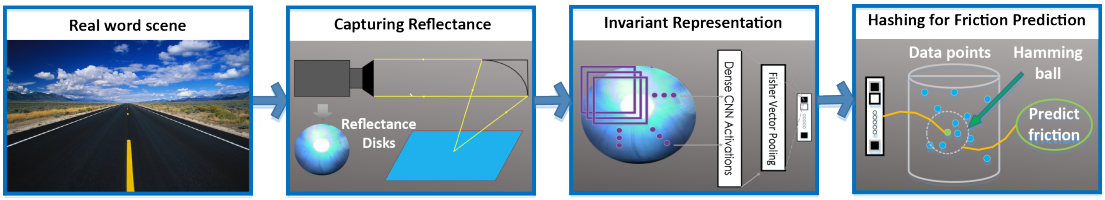


Figure 3.1: Deep reflectance codes for *friction-from-reflectance*. Reflectance is captured using one-shot in-field measurements. The binary embedding of Fisher Vector CNN preserves physical properties and provides invariant representation. The resulting hash codes is used in prediction of surface friction.

provides important cues about the surface, including what it is made of (i.e., material), and how it is shaped (i.e., surface roughness). Reflectance has recently been used to classify surfaces based on the underlying materials [19, 74]. and to recover fine-grained geometry [48, 65]. Reflectance, however, encodes richer information about the surface than just the material category and surface geometry.

In this work, we show that reflectance can be used to estimate tactile properties of the surface: determine the touch from appearance. We, in particular, build the first approach for *friction-from-reflectance*. An image-based friction estimation method has important implications in many applications. For example, robotics can plan how to grasp an object easily with an estimated friction coefficient. A mobile robot or autonomous automobile can use friction information to select an optimized driving mode. For this, we introduce a novel representation of reflectance patterns using a state-of-the-art texture representation that builds on key aspects of deep learning and binary embedding. We demonstrate the power of our approach called *deep reflectance codes*, first in conventional material recognition and then on predicting friction from one-shot measurements.

Reflectance measurements encode rich information that raw images do not. Recognizing materials using ordinary internet-mined photographs has shown great promise [32, 75–77]. Reflectance information provides a different, complementary approach [19] based on the measurement of the material characteristics, instead of phenomenological appearance. As introduced by Zhang et al. [19], reflectance disks provide a sampling of a surface BRDF without requiring a lab based dome-like measurement system. The disk images are acquired using a concave parabolic mirror positioned over a surface.

Due to the mirror geometry, each pixel corresponds to the reflected surface light from a different viewing direction. One of the key innovations, that we introduced in [19] is the use of *angular gradients* in representing surface appearance for recognition and other algorithms. While spatial filtering such as orientation gradients have been commonly used in recognizing texture and materials, angular gradients in the BRDF have been relatively unexplored and our prior work was the first to show that angular filtering is critical for material recognition. Spatial filtering the images comprising reflectance disks corresponds to angular filtering of the surface reflectance. The representation is learned in a multilayer deep learning approach.

These computational models of reflectance disks need to be sufficiently descriptive and yet have invariance to illumination and surface tilt. The measured appearance of a surface changes under multiple illumination angles. The illumination direction affects the position of the specular peak within the reflectance disk, necessitating translation invariance within the representation. The distribution of patterns in the reflectance disk also necessitates a shift-invariant representation similar to desired properties of texture representations. Prior work in representing reflectance disks [19] used hashing to create a binary code for fast recognition and retrieval in high dimensional spaces. Classic methods of texton histograms and texton maps have been replaced in recent years with deep representations for texture such as deep filter banks [1]. We introduce a novel material representation, which we refer to as *deep reflectance codes (DRC)*, that builds on these two concepts. Deep reflectance codes achieve necessary descriptive power and invariance by applying binary embedding to fisher vector convolutional neural nets (FV-CNN) [1]. To demonstrate the general discriminatory power of this new material representation, we test it in both image-based and reflectance disk-based material recognition and show that their performance surpasses past representations on several existing material datasets. Figure 3.1 shows an overview of our approach.

Deep reflectance codes provide a compact binary representation of the intrinsic phyiscal characteristics of the underlying material. We demonstrate this by showing, for the first time, that reflectance can be used to estimate the friction coefficient of a surface. For this, we collect a first-of-its-kind database of friction coefficients and

reflectance disks measured for 137 surfaces that can be grouped into 21 classes (shown in Figure 3.2). For exploratory analysis of how reflectance disks encode surface friction, we find a manifold showing the proximal layout of deep reflectance codes using t-SNE [78] as well as a corresponding friction map in t-SNE space. The experimental results show that the reflectance disks and their deep codes encode sufficient information to accurately predict the friction of a surface from its one-shot in-field reflectance measurements. The practical implications of this novel vision-based friction estimation is significant. Current friction sensors are tactile such as tire sensors that measure slip [79–81]. Appearance modeling in a fine-grained precise manner allows non-contact friction estimation (e.g. before the tire hits the road patch), which would be useful for various applications such as road surface ice detection, robot locomotion control, and the integration of haptics and graphics.

In summary, there are three main novel contributions: 1) deep reflectance codes for material representation, 2) friction estimation from reflectance, 3) and a database of friction coefficients and reflectance disks.

3.2 Related Work

Friction from reflectance is an entirely new area. Prior work in friction measurement requires surface contact. These contact friction sensors have been used for real-world friction estimates in applications such as haptics for textiles [82, 83], automobile tire sensors [79–81], and sheet metal rolling in manufacturing [84]. The ability to estimate friction with vision-based methods will have significant impact on these application areas and many others. Non-contact enables higher speeds and larger distances. Furthermore, material characterization based on friction does not depend on a semantic label and may have greater utility in applications that interact with surfaces.

In prior work [19], reflectance disks have been used for material recognition. Their work combines material descriptor of textons distribution with similarity preserving binary embedding methods such as circulant binary embedding (CBE) [67], bilinear embedding [68], iterative quantization (ITQ) [69], angular quantization-based binary

Figure 3.2: (Top) Images of the 137 material surfaces for the friction-material database ordered by friction coefficient. (Bottom) Example reflectance disks for corresponding material surfaces. The names for all material classes and the number of instances captured per class are shown in Table 3.1. As examples of material class names, the list of names for the first surface in each row are as follows: Smooth Ceramic Tile, Automotive Paint, Marble, Composite Flooring, Asphalt, Nylon, Linen, Leather and Sand Paper. As expected, samples such as Smooth Ceramic Tile have lower coefficients of friction (0.2) than samples such as Sand Paper (0.53).

codes (AQBC) [70], spectral hashing (SH) [71], locality-sensitive binary codes from shift-invariant kernels (SKLSH) [73], and locality sensitive hash (LSH) [72]. Traditional methods of texton histograms [7,8,29,85] have a weakness in their discriminatory power.

Fisher Vector pooling as part of deep learning frameworks has been shown to have excellent performance in texture recognition problems [1]. Binary embedding combined with Fisher Vector pooling [86] has been explored for image retrieval. Deep learning is known for discovering discriminant and robust feature representations. The combination of deep learning and binary compact hash codes is an interesting path that combines the efficiency of binary codes (hamming distance is fast) with the robust performance of deep learning. Semantic hashing [87, 88] builds a deep binary model by quantizing a pre-trained and fine-tuned stack of Restricted Boltzmann Machines (RBMs). Deep hashing [89] develops a neural network to learn multiple hierarchical non-linear transformations mapping raw images to compact binary hash codes. However, these hashing techniques rely on hand-crafted visual features as input.

Recent work shows that deep convolutional neural networks trained on a sufficiently large dataset such as ImageNet [90] can be transferred to other computer vision tasks [1, 2, 91, 92]. Lower convolutional layers remain similar on different datasets [93]. CNN activations are still sensitive to translation, rotation and scale [2]. Recognizing reflectance disks from unknown illumination and surface tilts is very similar to texture recognition. It is the distribution of features and visual structures, not their particular spatial location, that is most important. Therefore, dense pooling methods are essential. Multi-scale orderless pooling [2] and deep filter banks [1] using VLAD pooling [4] and Fisher Vector pooling of CNN activations achieve state-of-the-art results on texture, material and scene recognition tasks. We leverage Fisher vector pooling of CNN activations for exploring compact hash codes as binary representations of data.

3.3 One-Shot In-Field Reflectance Disks

We follow Zhang et al. [19] and use a mirror-based camera [25,48] to measure reflectance of surface points. The camera components are an off-axis parabolic mirror, a CCD

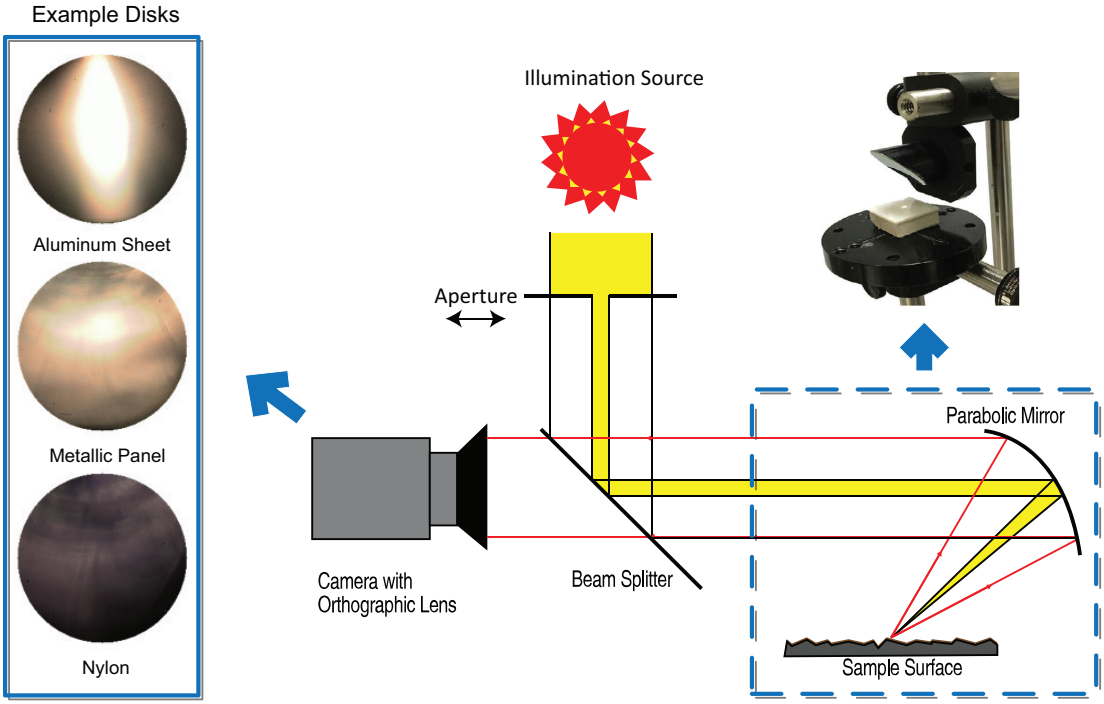


Figure 3.3: Schematic of the mirror-based camera. Reflectance disks are obtained by viewing a single point under multiple viewing directions using a concave parabolic mirror viewed by a telecentric lens. The off-axis concave parabolic mirror is shown in the upper right.

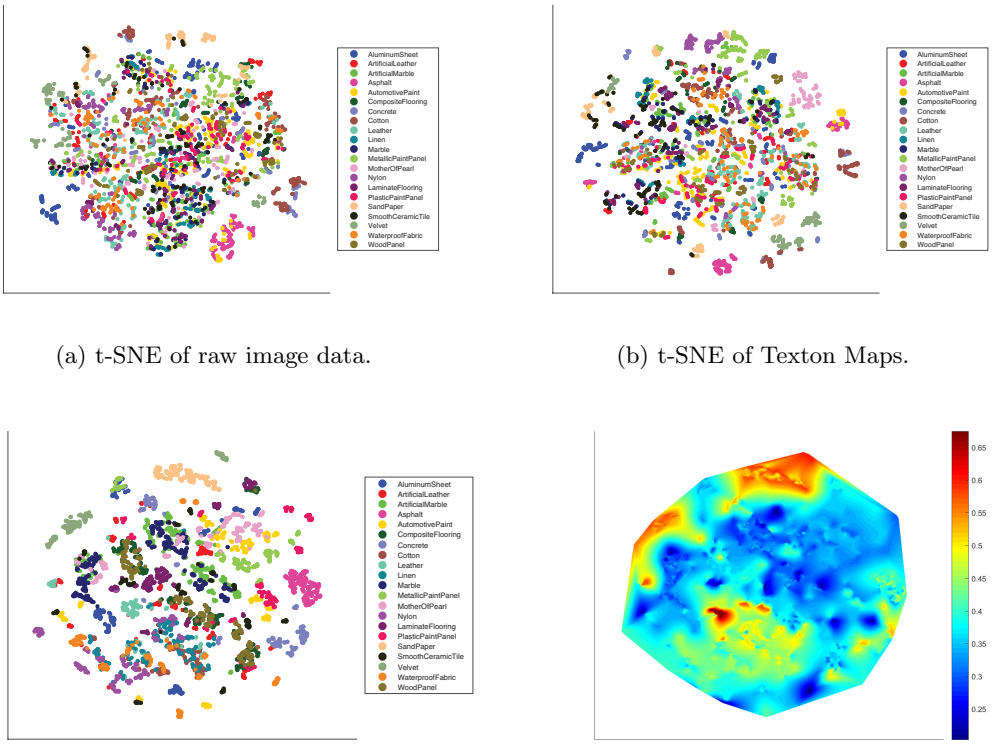
camera, a movable aperture and a beam splitter (as shown in Figure 3.3). The parabolic mirror is fixed so that its focus is at the surface point to be measured. The illumination source is a collimated beam of light parallel to the global plane of the surface passing through a movable aperture. The angle of the incident ray at the surface is determined by the intersection point with the mirror. Therefore, the illumination direction can be controlled by planar translations of the aperture. Similarly, light reflected from the surface point is reflected by the mirror to a set of parallel rays directed through a beam splitter to the camera. Each pixel of this camera image corresponds to a viewing direction of the surface point. Therefore, the recorded image, referred to as a *reflectance disk*, is an observation of a single point on the surface but from a dense sampling of viewing directions.

3.4 Deep Reflectance Codes

Reflectance naturally encodes the intrinsic physical properties of the surface, but its measurement is sensitive to illumination changes. For reflectance disks, the illumination changes correspond to translational shifts of image features; e.g. the specularity position shifts. We explore translation invariant representation to extract the physical information. Pre-trained CNN features have been applied to a number of computer vision tasks with great success [94–96], but CNN features are fairly sensitive to translation and rotation. Dense pooling of CNN activations removes the globally spatial information and therefore is robust to pattern translation and rotation. VLAD-CNN and FV-CNN achieved the state-of-the-art results in scene understanding and material recognition [1, 2]. We leverage the shift-invariance representation of dense pooling CNN for exploring compact hash codes. The resulting hash codes potentially preserve the similarities of surface physical characteristics, which we refer to as *deep reflectance codes*. A key difference from Cimpoi *et al.* [1] is that we introduce a unsupervised hashing approach rather than a material descriptor for supervised SVM material classification.

Visualizing the effect of the FV-CNN representation can be done with t-SNE [78]. Figure 3.4 shows reflectance disk representations using (a) raw reflectance disk data and (b) texton maps. The representation of FV-CNN in Figure 3.4 (c) shows significantly better class discrimination. This figure depicts many material instances under multiple illumination directions and we can see that our reflectance descriptor provides sufficient invariance and allows grouping of similar materials.

The challenge of using FV-CNN for retrieval is the high dimensionality of the Fisher vector representation ($64K$ dimensions for our implementation described in Section 3.6.1). To achieve a more compact representation, we integrate a binary embedding strategy which preserves the similarity of the Fisher vectors. The resulting hash codes are suitable for both material recognition and friction estimation. In this work, the recognition experiments are used to evaluate the quality of the representation in order to use it for friction estimation.



(a) t-SNE of raw image data.

(b) t-SNE of Texton Maps.

(c) Material Manifold of t-SNE FV-CNN.

(d) Friction Manifold in t-SNE space.

Figure 3.4: (a) and (b) shows t-SNE of raw image data and traditional texton representation. (c) t-SNE embedding of deep reflectance codes representation. Classes are color-coded (21 classes). There are 5-10 instances per class (137 material surfaces). For some classes there is significant intra-class reflectance variation, but most group well within the t-SNE manifold. (d) Friction map generated in the t-SNE space. Each instance can have a different friction value, as shown in Table 3.1.

DRC

For binary embedding, projection to a lower dimensional subspace that preserves the similarity is a key component. The Johnson-Lindenstrauss (J-L) Lemma implies that with high probability, the relative distances between all pair of points are approximately preserved under random projection [97]. Therefore, by randomly projecting the high-dimensional data into lower dimension, we can still achieve a comparative results with a linear classifier. Additionally, learning a kernel classifier using Fisher kernel is equivalent to learning a linear classifier on the Fisher vector [5]. By quantizing the randomly projected Fisher vector to binary hash codes, we are able to approximate the

behavior of linear classifier using Fisher Vectors, approximating a kernel classifier using Fisher Kernel. The dot product has been shown as a good measurement of similarity for Fisher vectors and we can use Local Sensitivity Hashing (LSH) [72] to binarize the Fisher Vector as in [86] quantizing the randomly projected data using the hyper-planes across the origin. The cosine similarities of all pairs of data are preserved under random projection [98]. Therefore the hamming distance of generated hash codes preserves the similarity of Fisher vectors. We use quantized random projection of Fisher representation in one variation of our approach, and we refer to this method as *DRC - deep reflectance codes.*

DRC-opt

LSH directly quantizes the randomly projected data into hash codes. Let $v \in \mathbb{R}^d$ represent a projected data point and the hash function $\text{sgn}(v)$ maps the data to the vertex of the hyper-cube $\{-1, 1\}^d$. The quantization error can be written as $\|\text{sgn}(v) - v\|^2$. Following the iterative quantization method [69], in order to preserve the local similarity of the data, a better hash code can be learned by rotating the data to minimize the quantization error:

$$\sum_{i=1}^{N} \|b_i - v_i R\|_2^2, \tag{3.1}$$

where N is the number of training data, the binary code is given by $b_i = \text{sgn}(v_i R)$, $R \in \mathbb{R}^{d \times d}$ is a rotation matrix. We use this combination of fisher vector CNN with this optimized binary embedding and we refer to it as *DRC-opt optimized deep reflectance codes.*

3.5 Friction from Reflectance

Deep reflectance codes give us a compact representation that encodes the reflectance disks which we expect to encode rich information about the physical properties of the material itself. We are particularly interested in probing tactile properties of a surface from its reflectance appearance. For this, we focus on estimating friction from reflectance disks. In this section, we introduce the first-of-its-kind friction-reflectance

Friction	AVG.	STD.	1	2	3	4	5	6	7	8	9	10
AluminumSheet	**0.354**	**0.018**	0.354	0.344	0.364	0.384	0.344	0.335				
ArtificialLeather	**0.463**	**0.099**	0.325	0.510	0.577	0.404	0.499					
ArtificialMarble	**0.296**	**0.013**	0.287	0.306	0.306	0.306	0.296	0.296	0.315	0.296	0.277	0.277
Asphalt	**0.398**	**0.043**	0.384	0.404	0.414	0.335	0.466	0.384				
AutomotivePaint	**0.342**	**0.019**	0.344	0.306	0.335	0.344	0.335	0.354	0.344	0.374		
CompositeFlooring	**0.371**	**0.016**	0.384	0.374	0.384	0.354	0.344	0.384	0.384	0.354	0.384	0.364
Concrete	**0.453**	**0.060**	0.424	0.364	0.435	0.456	0.521	0.521				
Cotton	**0.449**	**0.006**	0.445	0.445	0.456	0.445	0.456					
Leather	**0.494**	**0.108**	0.466	0.477	0.466	0.675	0.384					
Linen	**0.467**	**0.031**	0.499	0.510	0.466	0.435	0.445	0.445				
Marble	**0.320**	**0.028**	0.344	0.306	0.335	0.325	0.277	0.287	0.296	0.344	0.364	0.325
MetallicPaintPanel	**0.331**	**0.018**	0.344	0.335	0.335	0.315	0.306	0.354				
MotherOfPearl	**0.333**	**0.023**	0.306	0.306	0.344	0.344	0.335	0.364				
Nylon	**0.412**	**0.015**	0.435	0.424	0.424	0.394	0.404	0.404	0.414	0.394		
LaminateFlooring	**0.394**	**0.018**	0.394	0.384	0.374	0.384	0.404	0.424				
PlasticPaintPanel	**0.343**	**0.042**	0.325	0.325	0.414	0.315	0.374	0.306				
SandPaper	**0.547**	**0.039**	0.510	0.532	0.499	0.577	0.577	0.589				
SmoothCeramicTile	**0.224**	**0.021**	0.259	0.203	0.222	0.213	0.222					
Velvet	**0.564**	**0.026**	0.601	0.532	0.554	0.577	0.554					
WaterproofFabric	**0.394**	**0.014**	0.404	0.414	0.394	0.394	0.384	0.374				
WoodPanel	**0.346**	**0.031**	0.364	0.354	0.354	0.344	0.374	0.287				

Table 3.1: The measured friction coefficients of the material dataset; the friction sensor is shown at the right bottom corner. The collection of 137 material surfaces are grouped into classes. Each class has multiple distinct instances numbered from 1 to 10.

database and then apply deep reflectance codes to estimate friction from one-shot in-field reflectance disks.

3.5.1 Friction-Reflectance database

We collect 137 different materials (shown in Figure 3.2) and group them into 21 categories: aluminum sheet, artificial leather, artificial marble, asphalt, automotive paint, composite flooring, concrete, cotton, leather, linen, marble, metallic paint panel, mother of pearl, nylon, painted wood flooring, plastic paint panel, sand paper, smooth ceramic tile, velvet, waterproof fabric, wood panel. Our database includes 5-10 different material samples (instances) per category, and we measure 2 different surface spots per sample with 7 illumination directions $(-20°, -10°, 0, 10°, 20°)$ along axis and $(-10°, 10°)$ off axis, where $0°$ is frontal illumination aligned with the surface normal. Additionally, images with 3 exposure settings are collected for high dynamic range imaging. The total number of reflectance disks are 5754.

Friction of a surface is an intrinsic physical property of a surface that can readily be measured non-visually with a contact device. The coefficient of kinetic friction

multiplied by the force normal (due to gravity) is the force of friction which act to hold an object in place on a surface. We adopt a simple approach for measuring kinetic friction coefficient by using an inclined plane. In the experiments, we hold the object just above the surface of the inclined plane and release it. At low angles, the object should does not move. As the angle increases, the object begins to slide down at a constant velocity. The angle θ is recorded, and the kinetic friction coefficient is calculated as $\mu = \tan\theta$. For a thin material such as cloth, we attach it to a solid object for the friction measurement. The measured kinetic friction coefficients for all 137 surfaces and the inclined plane friction measurement device are shown in the Table 3.1. The measured kinetic friction coefficients are shown in Table 3.1.

3.5.2 Hashing for Friction Prediction

Deep reflectance codes encode rich information of physical properties of material surface from reflectance disks. The similarities of material physical properties are preserved in Hamming distance by compact hash codes. Figure 3.4 (c,d) visualizes the correspondence between the friction distribution and the representation using t-SNE, and we are able to see the material surfaces with similar friction coefficients are likely to have small distance. Therefore, our approach of deep reflectance codes can be used for friction estimation. We build a binary hash table with the DRC as the key and the corresponding surface friction coefficient as the hash value. The binary representations are compact and amenable for fast nearest neighbor retrieval. The predicted friction coefficient of a sample is given by the average of friction coefficients of retrieved samples.

3.6 Experimental Results

To evaluate the proposed approach of deep reflectance codes in spatial invariance representation and friction prediction, we conduct two groups of experiments. We first apply deep reflectance codes on retrieval for material recognition as a litmus test of the representation and then evaluate the performance of friction prediction from one-shot measurements.

dataset	FV-SIFT-Hash	CNN-ITQ	VLAD-CNN-KBE	FV-CNN-KBE	DRC	DRC-opt
reflectance	**64.5**$_{\pm 5.8}$	51.9$_{\pm 7.0}$	60.1$_{\pm 6.7}$	58.8$_{\pm 7.1}$	59.9$_{\pm 4.8}$	60.2$_{\pm 5.1}$
FMD	48.3$_{\pm 4.5}$	65.0$_{\pm 1.7}$	59.4$_{\pm 2.8}$	57.7$_{\pm 2.7}$	64.8$_{\pm 2.8}$	**65.5**$_{\pm 3.0}$
DTD	43.6$_{\pm 1.4}$	52.6$_{\pm 1.3}$	52.3$_{\pm 1.4}$	53.1$_{\pm 0.9}$	55.4$_{\pm 1.2}$	**55.8**$_{\pm 1.4}$
KTH	72.0$_{\pm 4.3}$	73.7$_{\pm 3.4}$	75.6$_{\pm 6.2}$	74.4$_{\pm 5.2}$	76.6$_{\pm 5.2}$	**77.2**$_{\pm 6.4}$

Table 3.2: Comparison of encodings and hashing methods (using 1024-bit hash codes) on different material datasets. The recognition precision over 10 runs, large standard deviation of reflectance and KTH dataset due to randomly selecting one sample surface from each class as test set in each iteration (described in Section 3.6.1).

3.6.1 Hashing for Material Recognition

Material and texture are typically modeled by orderless pooling of visual patterns, *i.e.* texton distributions, as the shape and spatial information are not important. We apply deep reflectance codes on image retrieval for material recognition to demonstrate the discriminative power and the spatial invariance of our representation.

Additional Datasets

We also consider three other material datasets to evaluate our DRC approach for general material recognition. The first one is Flickr Material Dataset (FMD) [32], which contains 10 material classes. The second one is the Describable Texture Datasets (DTD) [77], containing texture images labeled with 47 describable attributes. The third texture dataset is KTH-TIPS-2b (KTH) [99], which contains 11 material categories with 4 samples per category and a number of example images for each sample. For this unsupervised hashing for material recognition experiments, the training set means the retrieval set for nearest neighbor search. For the DTD and FMD, the test set are given by randomly selected 20% images, and the rest are for training. The test set of reflectance database and KTH-TIPS are given by randomly selecting one sample of each material category, and the remaining samples are used for training. The experimental results are averaged in 10 runs.

Baseline Methods

For general evaluation of our DRC in material recognition, we compare it with the following methods and settings: FV-SIFT-Hash [86], CNN-ITQ [69], VLAD-CNN-KBE [100], FV-CNN-KBE. FV-SIFT-Hash is binarized fisher vector pooling of SIFT feature as in [86]. CNN-ITQ is iterative quantization binary hashing [69] on 4096-dimensional CNN activations, which is typically compared as a baseline for deep hashing algorithm. KBE represents a latest work Kronecker Binary Embedding [100] (randomized orthogonal version with order 2 is used in this experiment, which is good for high-dimension data). We define ground truth as the given class label of material, and use the mode of labels from 10 retrieved nearest neighbors of hash codes in Hamming Distance as the predicted label. The mean recognition precision is computed over all inquiries from test set averaged over 10 runs.

In our experiments, a pre-trained VGG-M model [101] is used for computing CNN activations. CNN activations for VLAD are taken by the output on the first fully connected layer (full6) with a 4096 dimension. We extract activations for 128×128 paches sampled with a stride of 32 pixels, and use PCA to reduce them to 512 dimensions [2]. We use 64 k-means centers for computing VLAD pooling, the vector is given by assigning each patch to its nearest clustering center and aggregating the residuals of patches minus the center. Hence, the dimension of VLAD pooling CNN is $32K$. The CNN activations for FV pooling are taken from the output of the first convolutional layer with 512 dimensions (conv5 of VGG-M). We use 64 Gaussian components for the FV representation, resulting in $65K$ dimensions FV-CNN descriptors. The implementations are based on MatConvNet [102] and VLFeat [103].

Table 3.2 shows material recognition results comparing different encodings and hashing methods. We can see the randomized and optimized version of DRC hashing outperform the other methods in FTD, DTD and KTH datasets. Note that our compact unsupervised binary hashing methods can achieve comparable results with training SVM on FV-CNN as reported in [77]. FV pooling provides domain transfer of CNN activations

and we have found it make the use of the property that FV pooling essentially discarding the influence the CNN activations corresponding to frequent background structures that are often domain specific. DRC-opt (60.2%) is much better than directly using CNN activations (CNN-ITQ 51.9%) for the reflectance dataset. The methods using CNN features are slightly worse than hashing with FV pooling SIFT feature, since the reflectance disks dataset is very different from the CNN pre-trained dataset, ImageNet, which consists real world images. Reflectance Hashing with textons [19] has an overall recognition precision of 56.69% on this reflectance dataset. We test this method on the reflectance dataset only since it is specifically developed for reflectance disks. Our evaluation indicates the utility of this binary code representation both for general material recognition and specifically for reflectance disk representation.

3.6.2 Friction Prediction

For friction prediction experiments, we use 1024-bit deep reflectance codes to estimate the surface friction from one-shot in-field reflectance disks. For reflectance-friction dataset, each surface instance (from same or different material class) can have a different friction coefficient, as shown in Table 3.1. We use leave-one-out cross-validation as training test splits. Each time the disks of one surface is selected as test set and the disks of the rest 136 surfaces are used for training (retrieval). The predicted friction coefficient is given by calculating the average of retrieved 10 neighbors.

Baseline Method

To evaluate our hash codes on friction prediction, we use a regression network as a baseline method. The feedforward network takes the material reflectance representation to directly predict the friction coefficients by fitting a parameterized functional form using FV-CNN representation as input (projected into 1024 dimension). Each neuron in the network nonlinearly transforms the weighted received inputs into outputs. Our regression net has two hidden layers with 16 and 8 neurons respectively and apply a sigmoid function as activation function. The optimal values for parameters are learned by minimizing the sum of the square errors between measured friction coefficients and

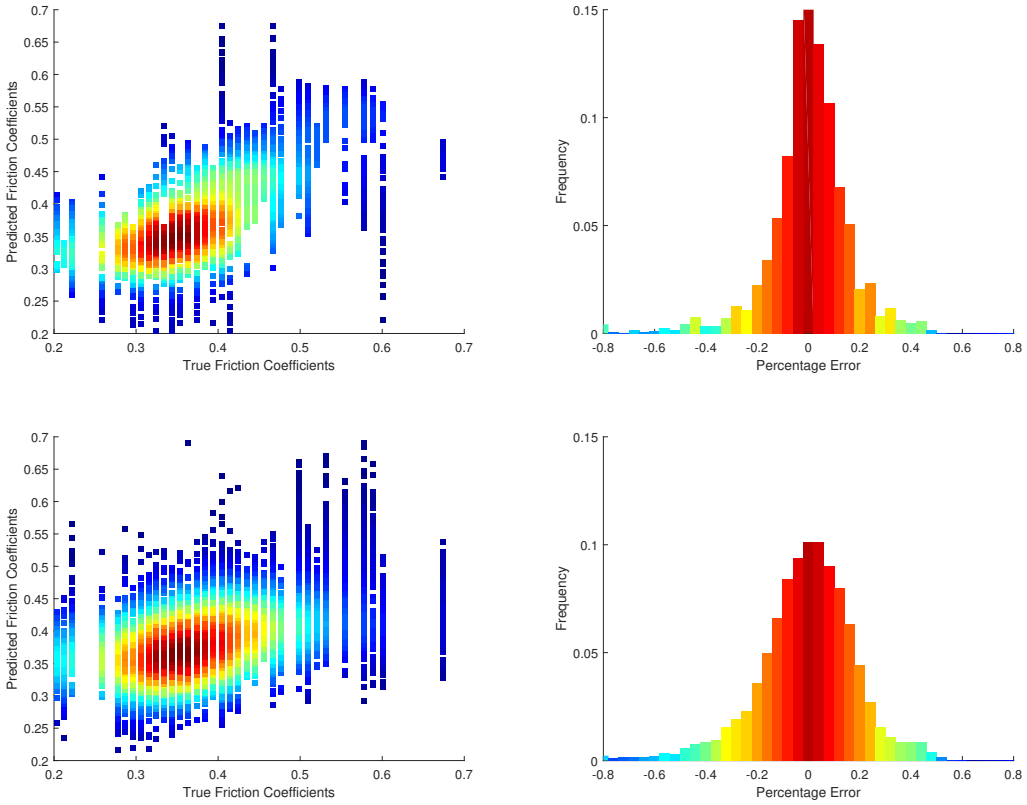


Figure 3.5: (Top-left) Friction prediction scatter plot, colored by density. (Top-right) Percentage error distribution, the average percentage error is 11.15%. (Bottom row) The friction prediction with neural net regression has the average percentage error of 14.74%.

those predicted by the network.

Figure 3.5 shows the prediction results of DRC and the baseline method. The scatter plot (top-left) is color coded and the colors represent the local density around each point. The plot shows the largest density of points have low prediction error. The top-right figure shows a histogram of the error percentage. The predicted data mainly fall into small error bins and the overall mean percentage error is 11.15%. The results of baseline method are shown at the bottom and the overall mean percentage error is 14.74%.

Figure 3.6 provides a more detailed view of the friction-from-reflectance prediction by showing the per-class recognition performance. The red line is the ground truth of

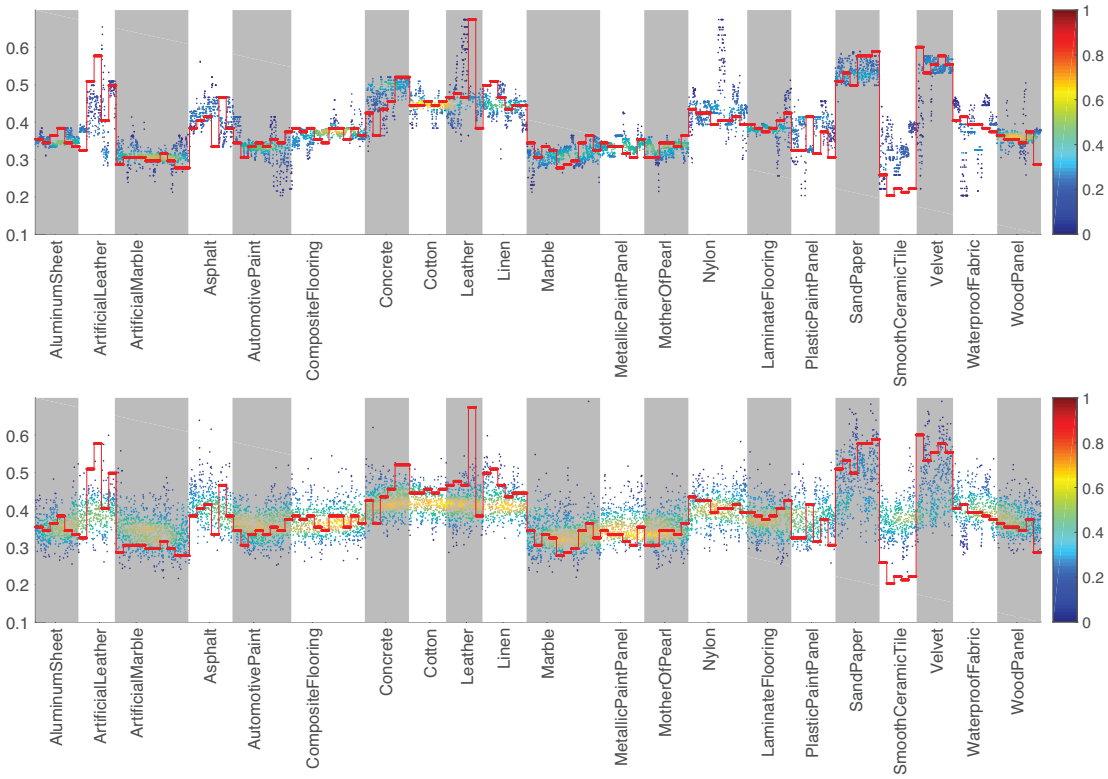


Figure 3.6: (Top) Per-class friction prediction results, colored by local density. The red line is the ground truth of friction coefficients. The mean percentage error is 11.15%. The predicted coefficients match the overall friction variation with material class. Friction-from-reflectance has never been attempted to our knowledge and these results show great promise for this challenging task. (Bottom) The friction prediction with neural net regression has the average percentage error of 14.74%.

friction coefficients, and the points are predicted values colored by the local density of points. Our DRC approach outperforms the baseline method. However, the DRC as a hashing approach has the limitation for predicting the surface with extremely high or low friction due to absence of certain data from retrieval set.

3.7 Summary and Conclusion

We have presented a framework of reflectance codes to encode sampled reflectance functions. There are three main contributions: 1) first of its kind friction-from-reflectance method, 2) reflectance codes for material recognition and 3) a new combination of FV-CNN with binary embedding that can be applied to image-based material recognition as well as reflectance disks. The database of reflectance disks and corresponding friction

measurements are publicly available for future research.

Chapter 4

Deep TEN: Texture Encoding Network

Overview

This chapter on Deep TEN: Texture Encoding Network is based on our paper [21]. We propose a Deep Texture Encoding Network (Deep-TEN) with a novel Encoding Layer integrated on top of convolutional layers, which ports the entire dictionary learning and encoding pipeline into a single model. Current methods build from distinct components, using standard encoders with separate off-the-shelf features such as SIFT descriptors or pre-trained CNN features for material recognition. Our new approach provides an end-to-end learning framework, where the inherent visual vocabularies are learned directly from the loss function. The features, dictionaries and the encoding representation for the classifier are all learned simultaneously. The representation is orderless and therefore is particularly useful for material and texture recognition. The Encoding Layer generalizes robust residual encoders such as VLAD and Fisher Vectors, and has the property of discarding domain specific information which makes the learned convolutional features easier to transfer. Additionally, joint training using multiple datasets of varied sizes and class labels is supported resulting in increased recognition performance. The experimental results show superior performance as compared to state-of-the-art methods using gold-standard databases such as MINC-2500, Flickr Material Database, KTH-TIPS-2b, and two recent databases 4D-Light-Field-Material and GTOS. The source code for the complete system are publicly available[1].

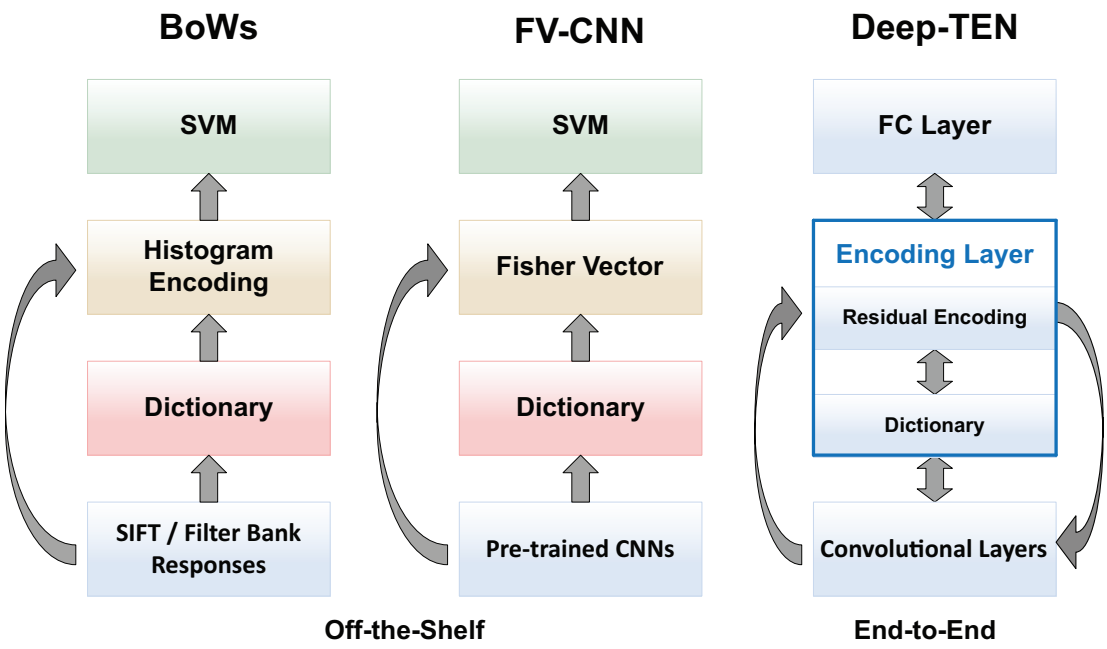


Figure 4.1: A comparison of classic approaches and the proposed Deep Texture Encoding Network. Traditional methods such as bag-of-words BoW (left) have a structural similarity to more recent FV-CNN methods (center). Each component is optimized in separate steps as illustrated with different colors. In our approach (right) the entire pipeline is learned in an integrated manner, tuning each component for the task at hand (end-to-end texture/material/pattern recognition).

4.1 Background

With the rapid growth of deep learning, convolutional neural networks (CNNs) has become the de facto standard in many object recognition algorithms. The goals of material and texture recognition algorithms, while similar to object recognition, have the distinct challenge of capturing an orderless measure encompassing some spatial repetition. For example, distributions or histograms of features provide an orderless encoding for recognition. In classic computer vision approaches for material/texture recognition, hand-engineered features are extracted using interest point detectors such as SIFT [6] or filter bank responses [7–10]. A dictionary is typically learned offline and then the feature distributions are encoded by Bag-of-Words (BoWs) [11–14], In the final step, a classifier such as SVM is learned for classification. In recent work,

[1] http://ece.rutgers.edu/vision

hand-engineered features and filter banks are replaced by pre-trained CNNs and BoWs are replaced by the robust residual encoders such as VLAD [4] and its probabilistic version Fisher Vector (FV) [5]. For example, Cimpoi *et al.* [1] assembles different features (SIFT, CNNs) with different encoders (VLAD, FV) and have achieved state-of-the-art results. These existing approaches have the advantage of accepting arbitrary input image sizes and have no issue when transferring features across different domains since the low-level features are generic. However, these methods (both classic and recent work) are comprised of stacking self-contained algorithmic components (feature extraction, dictionary learning, encoding, classifier training) as visualized in Figure 6.2 (left, center). Consequently, they have the disadvantage that the features and the encoders are fixed once built, so that feature learning (CNNs and dictionary) does not benefit from labeled data. We present a new approach (Figure 6.2, right) where the entire pipeline is learned in an end-to-end manner.

Deep learning [94] is well known as an end-to-end learning of hierarchical features, so what is the challenge in recognizing textures in an end-to-end way? The convolution layer of CNNs operates in a sliding window manner acting as a local feature extractor. The output featuremaps preserve a relative spatial arrangement of input images. The resulting globally ordered features are then concatenated and fed into the FC (fully connected) layer which acts as a classifier. This framework has achieved great success in image classification, object recognition, scene understanding and many other applications, but is typically not ideal for recognizing textures due to the need for an spatially invariant representation describing the feature distributions instead of concatenation. Therefore, an orderless feature pooling layer is desirable for end-to-end learning. The challenge is to make the loss function differentiable with respect to the inputs and layer parameters. We derive a new back propagation equation series (see Appendix A.1). In this manner, encoding for an orderless representation can be integrated within the deep learning pipeline.

As the **first contribution** of this work, we introduce a novel *learnable residual encoding layer* which we refer to as the *Encoding Layer*, that ports the entire dictionary learning and residual encoding pipeline into a single layer for CNN. The Encoding

Layer has three main properties. (1) The Encoding Layer generalizes robust residual encoders such as VLAD and Fisher Vector. This representation is orderless and describes the feature distribution, which is suitable for material and texture recognition. (2) The Encoding Layer acts as a pooling layer integrated on top of convolutional layers, accepting arbitrary input sizes and providing output as a fixed-length representation. By allowing arbitrary size images, the Encoding Layer makes the deep learning framework more flexible and our experiments show that recognition performance is often improved with multi-size training. In addition, (3) the Encoding Layer learns an inherent dictionary and the encoding representation which is likely to carry domain-specific information and therefore is suitable for transferring pre-trained features. In this work, we transfer CNNs from object categorization (ImageNet [26]) to material recognition. Since the network is trained end-to-end as a regression progress, the convolutional features learned together with Encoding Layer on top are easier to transfer (likely to be domain-independent).

The **second contribution** of this work is a new framework for end-to-end material recognition which we refer to as *Texture Encoding Network - Deep TEN*, where the feature extraction, dictionary learning and encoding representation are learned together in a single network as illustrated in Figure 6.2. Our approach has the benefit of gradient information passing to each component during back propagation, tuning each component for the task at hand. Deep-Ten outperforms existing modular methods and achieves the **state-of-the-art** results on material/texture datasets such as MINC-2500 and KTH-TIPS-2b. Additionally, this Deep Encoding Network performs well in general recognition tasks beyond texture and material as demonstrated with results on MIT-Indoor and Caltech-101 datasets. We also explore how convolutional features learned with Encoding Layer can be transferred through joint training on two different datasets. The experimental result shows that the recognition rate is significantly improved with this joint training.

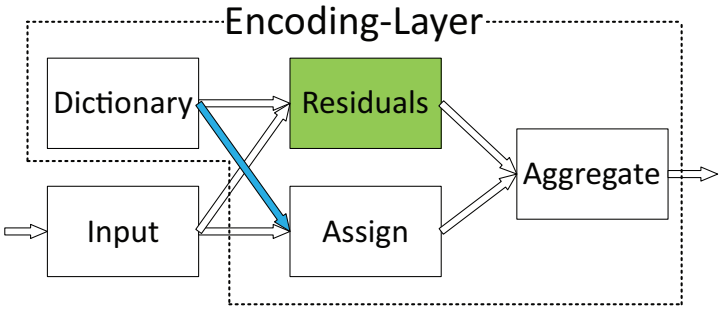


Figure 4.2: The *Encoding Layer* learns an inherent *Dictionary*. The *Residuals* are calculated by pairwise difference between visual descriptors of the input and the codewords of the dictionary. Weights are *assigned* based on pairwise distance between descriptors and codewords. Finally, the residual vectors are *aggregated* with the assigned weights.

	Deep Features	Dictionary Learning	Residual Encoding	Any-size	Fine-tuning	End-to-end Classification
BoWs		✓		✓		
Fisher-SVM [104]		✓	✓	✓		
Encoder-CNN [1, 2]	✓	✓	✓	✓		
CNN	✓				✓	✓
B-CNN [105]	✓				✓	
SPP-Net [106]	✓			✓	✓	✓
Deep TEN (ours)	✓	✓	✓	✓	✓	✓

Table 4.1: Methods Overview. Compared to existing methods, Deep-Ten has several desirable properties: it integrates deep features with dictionary learning and residual encoding and it allows any-size input, fine-tuning and provides end-to-end classification. (Encoder CNN (FV [1] VLAD [2])

4.2 Learnable Residual Encoding Layer

Residual Encoding Model

Given a set of N visual descriptors $X = \{x_1, ..x_N\}$ and a learned codebook $C = \{c_1, ...c_K\}$ containing K codewords that are D-dimensional, each descriptor x_i can be assigned with a weight a_{ik} to each codeword c_k and the corresponding residual vector is denoted by $r_{ik} = x_i - c_k$, where $i = 1, ...N$ and $k = 1, ...K$. Given the assignments and the residual vector, the residual encoding model applies an aggregation operation for every single codeword c_k:

$$e_k = \sum_{i=1}^{N} e_{ik} = \sum_{i=1}^{N} a_{ik} r_{ik}. \tag{4.1}$$

The resulting encoder outputs a fixed length representation $E = \{e_1, ...e_K\}$ (independent of the number of input descriptors N).

Encoding Layer

The traditional visual recognition approach can be partitioned into feature extraction, dictionary learning, feature pooling (encoding) and classifer learning as illustrated in Figure 6.2. In our approach, we port the dictionary learning and residual encoding into a single layer of CNNs, which we refer to as the *Encoding Layer*. The Encoding Layer simultaneously learns the encoding parameters along with with an inherent dictionary in a fully supervised manner. The inherent dictionary is learned from the distribution of the descriptors by passing the gradient through assignment weights. During the training process, the updating of extracted convolutional features can also benefit from the encoding representations.

Consider the assigning weights for assigning the descriptors to the codewords. Hard-assignment provides a single non-zero assigning weight for each descriptor x_i, which corresponds to the nearest codeword. The k-th element of the assigning vector is given by $a_{ik} = \delta(\|r_{ik}\|^2 = min\{\|r_{i1}\|^2, ...\|r_{iK}\|^2\})$ where δ is the indicator function (outputs 0 or 1). Hard-assignment doesn't consider the codeword ambiguity and also makes the model non-differentiable. Soft-weight assignment addresses this issue by assigning a descriptor to each codeword [66]. The assigning weight is given by

$$a_{ik} = \frac{\exp(-\beta\|r_{ik}\|^2)}{\sum_{j=1}^{K}\exp(-\beta\|r_{ij}\|^2)}, \tag{4.2}$$

where β is the smoothing factor for the assignment.

Soft-assignment assumes that different clusters have equal scales. Inspired by guassian mixture models (GMM), we further allow the smoothing factor s_k for each cluster center c_k to be learnable:

$$a_{ik} = \frac{\exp(-s_k\|r_{ik}\|^2)}{\sum_{j=1}^{K}\exp(-s_j\|r_{ij}\|^2)}, \tag{4.3}$$

which provides a finer modeling of the descriptor distributions. The Encoding Layer concatenates the aggregated residual vectors with assigning weights (as in Equation 4.1). As is typical in prior work [5, 107], the resulting vectors are normalized using the $L2$-norm.

End-to-end Learning

The Encoding Layer is a directed acyclic graph as shown in Figure 4.2, and all the components are differentiable $w.r.t$ the input X and the parameters (codewords $C = \{c_1, ...c_K\}$ and smoothing factors $s = \{s_1, ...s_k\}$). Therefore, the Encoding Layer can be trained end-to-end by standard SGD (stochastic gradient descent) with backpropagation. We provide the details and relevant equation derivations in the Appendix A.1.

4.2.1 Relation to Other Methods

Relation to Dictionary Learning

Dictionary Learning is usually learned from the distribution of the descriptors in an unsupervised manner. K-means [108] learns the dictionary using hard-assignment grouping. Gaussian Mixture Model (GMM) [109] is a probabilistic version of K-means, which allows a finer modeling of the feature distributions. Each cluster is modeled by a Gaussian component with its own mean, variance and mixture weight. The Encoding Layer makes the inherent dictionary differentiable $w.r.t$ the loss function and learns the dictionary in a supervised manner. To see the relationship of the Encoding Layer to K-means, consider Figure 4.2 with omission of the residual vectors (shown in green of Figure 4.2) and let smoothing factor $\beta \to \infty$. With these modifications, the Encoding Layer acts like K-means. The Encoding Layer can also be regarded as a simplified version of GMM, that allows different scaling (smoothing) of the clusters.

Relation to BoWs and Residual Encoders

BoWs (bag-of-word) methods typically hard assign each descriptor to the nearest codeword and counts the occurrence of the visual words by aggregating the assignment vectors $\sum_{i=1}^{N} a_i$ [110]. An improved BoW employs a soft-assignment weights [111]. VLAD [4] aggregates the residual vector with the hard-assignment weights. NetVLAD [4] makes two relaxations: (1) soft-assignment to make the model differentiable and (2) decoupling the assignment from the dictionary which makes the assigning weights depend only on the input instead of the dictionary. Therefore, the codewords are not

learned from the distribution of the descriptors. Considering Figure 4.2, NetVLAD drops the link between visual words with their assignments (the blue arrow in Figure 4.2). Fisher Vector [5] concatenates both the 1st order and 2nd order aggregated residuals. FV-CNN [1] encodes off-the-shelf CNNs with pre-trained CNN and achieves good result in material recognition. Fisher Kernel SVM [104] iteratively update the SVM by a convex solver and the inner GMM parameters using gradient descent. A key difference from our work is that this Fisher Kernel method uses hand-crafted instead of learning the features. VLAD-CNN [2] and FV-CNN [1] build off-the-shelf residual encoders with pre-trained CNNs and achieve great success in robust visual recognition and understanding areas.

Relation to Pooling

In CNNs, a pooling layer (Max or Avg) is typically used on top of the convolutional layers. Letting $K = 1$ and fixing $c = 0$, the Encoding Layer simplifies to Sum pooling ($e = \sum_{i=1}^{N} x_i$ and $\frac{d_\ell}{d_{x_i}} = \frac{d_\ell}{d_e}$). When followed by $L2$-normalization, it has exactly the same behavior as Avg pooling. The convolutional layers extract features as a sliding window, which can accept arbitrary input image sizes. However, the pooling layers usually have fixed receptive field size, which lead to the CNNs only allowing fixed input image size. SPP pooling layer [106] accepts different size by fixing the pooling bin number instead of receptive field sizes. The relative spatial orders of the descriptors are preserved. Bilinear pooling layer [105] removes the globally ordered information by summing the outer-product of the descriptors across different locations. Our Encoding Layer acts as a pooling layer by encoding robust residual representations, which converts arbitrary input size to a fix length representation. Table 5.2 summarizes the comparison our approach to other methods.

4.3 Deep Texture Encoding Network

We refer to the deep convolutional neural network with the Encoding Layer as *Deep Texture Encoding Network (Deep-TEN)*. In this section, we discuss the properties of

	output size	Deep-TEN 50
Conv1	176×176×64	7×7, stride 2
Res1	88×88×256	3 × 3 max pool, stride 2
		$\begin{bmatrix} 1\times1, 64 \\ 3\times3, 64 \\ 1\times1, 256 \end{bmatrix} \times 3$
Res2	44×44×512	$\begin{bmatrix} 1\times1, 128 \\ 3\times3, 128 \\ 1\times1, 512 \end{bmatrix} \times 4$
Res3	22×22×1024	$\begin{bmatrix} 1\times1, 256 \\ 3\times3, 256 \\ 1\times1, 1024 \end{bmatrix} \times 6$
Res4	11×11×2048	$\begin{bmatrix} 1\times1, 512 \\ 3\times3, 512 \\ 1\times1, 2048 \end{bmatrix} \times 3$
Projection + Reshape	121×128	conv 1×1, 2048⇒128
		W×H×D⇒N×D
Encoding	32×128	32 codewords
$L2$-norm + FC	n classes	1×1 FC

Table 4.2: Deep-TEN architectures for adopting 50 layer pre-trained ResNet. The 2nd column shows the featuremap sizes for input image size of 352×352. When multi-size training for input image size 320×320, the featuremap after Res4 is 10×10. We adopt a 1×1 convolutional layer after Res4 to reduce number of channels.

the Deep-TEN, that is the property of integrating Encoding Layer with an end-to-end CNN architecture.

Domain Transfer

Fisher Vector (FV) has the property of discarding the influence of frequently appearing features in the dataset [5], which usually contains domain specific information [20]. FV-CNN has shown its domain transfer ability practically in material recognition work [1]. Deep-TEN generalizes the residual encoder and also preserves this property. To see this intuitively, consider the following: when a visual descriptor x_i appears frequently in the data, it is likely to be close to one of the visual centers c_k. Therefore, the resulting residual vector corresponding to c_k, $r_{ik} = x_i - c_k$, is small. For the residual vectors of r_{ij} corresponding to c_j where $j \neq k$, the corresponding assigning weight a_{ij} becomes small as shown in Equation 4.3. The Encoding Layer aggregates the residual vectors

with assignment weights and results in small values for frequently appearing visual descriptors. This property is essential for transferring features learned from different domain, and in this work we transfer CNNs pre-trained on the object dataset ImageNet to material recognition tasks.

Traditional approaches do not have domain transfer problems because the features are usually generic and the domain-specific information is carried by the dictionary and encoding representations. The proposed Encoding Layer generalizes the dictionary learning and encoding framework, which carries domain-specific information. Because the entire network is optimized as a regression progress, the resulting convolutional features (with Encoding Layer learned on top) are likely to be domain-independent and therefore easier to transfer.

Multi-size Training

CNNs typically require a fixed input image size. In order to feed into the network, images have to be resized or cropped to a fixed size. The convolutional layers act as in sliding window manner, which can allow any input sizes (as discussed in SPP [106]). The FC (fully connected) layer acts as a classifier which take a fix length representation as input. Our Encoding Layer act as a pooling layer on top of the convolutional layers, which converts arbitrary input sizes to a fixed length representation. Our experiments show that the classification results are often improved by iteratively training the Deep Encoding Network with different image sizes. In addition, this multi-size training provides the opportunity for cross dataset training.

Joint Deep Encoding

There are many labeled datasets for different visual problems, such as object classification [26, 112, 113], scene understanding [114, 115], object detection [116, 117] and material recognition [19, 118]. An interesting question to ask is: how can different visual tasks benefit each other? Different datasets have different domains, different labeling strategies and sometimes different image sizes (e.g. CIFAR10 [112] and ImageNet [26]). Sharing convolutional features typically achieves great success [106, 119]. The concept

	MINC-2500	FMD	GTOS	KTH	4D-Light	MIT-Indoor	Caltech-101
FV-SIFT	46.0	47.0	65.5	66.3	58.4	51.6	63.4
FV-CNN (VGG-VD)	61.8	75.0	77.1	71.0	70.4	67.8	83.0
Deep-TEN (ours)	**80.6**	**80.2**$_{\pm 0.9}$	**84.3**$_{\pm 1.9}$	**82.0**$_{\pm 3.3}$	**81.7**$_{\pm 1.0}$	**71.3**	**85.3**

Table 4.3: The table compares the recognition results of Deep-TEN with off-the-shelf encoding approaches, including Fisher Vector encoding of dense SIFT features (FV-SIFT) and pre-trained CNN activations (FV-CNN) on different datasets using single-size training. Top-1 test accuracy mean±std % is reported and the best result for each dataset is marked bold. (The results of Deep-TEN for FMD, GTOS, KTH datasets are based on 5-time statistics, and the results for MINC-2500, MIT-Indoor and Caltech-101 datasets are averaged over 2 runs. The baseline approaches are based on 1-time run.)

	MINC-2500	FMD	GTOS	KTH	4D-Light	MIT-Indoor
FV-CNN (VGG-VD) multi	63.1	74.0	79.2	77.8	76.5	67.0
FV-CNN (ResNet) multi	69.3	78.2	77.1	78.3	77.6	76.1
Deep-TEN (ours)	80.6	**80.2**$_{\pm 0.9}$	84.3$_{\pm 1.9}$	82.0$_{\pm 3.3}$	**81.7**$_{\pm 1.0}$	71.3
Deep-TEN (ours) multi	**81.3**	78.8$_{\pm 0.8}$	**84.5**$_{\pm 2.9}$	**84.5**$_{\pm 3.5}$	81.4$_{\pm 2.6}$	**76.2**

Table 4.4: Comparison of single-size and multi-size training.

of *multi-task learning* [120] was originally proposed in [121], to jointly train cross different datasets. An issue in joint training is that features from different datasets may not benefit from the combined training since the images contain domain-specific information. Furthermore, it is typically not possible to learn deep features from different image sizes. Our Encoding Layer on top of convolution layers accepts arbitrary input image sizes and learns domain independent convolutional features, enabling convenient joint training. We present and evaluate a network that shares convolutional features for two different dataset and has two separate Encoding Layers. We demonstrate joint training with two datasets and show that recognition results are significantly improved.

4.4 Experimental Results

Datasets

The evaluation considers five material and texture datasets. *Materials in Context Database* (MINC) [118] is a large scale material in the wild dataset. In this work, a publicly available subset (MINC-2500, Sec 5.4 of original paper) is evaluated with provided train-test splits, containing 23 material categories and 2,500 images per-category.

	MINC-2500	FMD	GTOS	KTH	4D-Light
Deep-TEN* (ours)	**81.3**	80.2$_{\pm 0.9}$	**84.5**$_{\pm \mathbf{2.9}}$	**84.5**$_{\pm \mathbf{3.5}}$	**81.7**$_{\pm \mathbf{1.0}}$
State-of-the-Art	76.0$_{\pm 0.2}$ [118]	**82.4**$_{\pm \mathbf{1.4}}$ [1]	N/A	81.1$_{\pm 1.5}$ [122]	77.0$_{\pm 1.1}$ [123]

Table 4.5: Comparison with state-of-the-art on four material/textures dataset (GTOS is a new dataset, so SoA is not available). Deep-TEN* denotes the best model of Deep Ten and Deep Ten multi.

Flickr Material Dataset (FMD) [32], a popular benchmark for material recognition containing 10 material classes, 90 images per-class used for training and 10 for test. *Ground Terrain in Outdoor Scenes Dataset* (GTOS) [24] is a dataset of ground materials in outdoor scene with 40 categories. The evaluation is based on provided train-test splits. *KTH-TIPS-2b* (KTH)- [99], contains 11 texture categories and four samples per-category. Two samples are randomly picked for training and the others for test. *4D-Light-Field-Material* (4D-Light) [123] is a recent light-field material dataset containing 12 material categories with 100 samples per-category. In this experiment, 70 randomly picked samples per-category are used as training and the others for test and only one angular resolution is used per-sample. For general classification evaluations, two additional datasets are considered. *MIT-Indoor* [124] dataset is an indoor scene categorization dataset with 67 categories, a standard subset of 80 images per-category for training and 20 for test is used in this work. *Caltech 101* [125] is a 102 category (1 for background) object classification dataset; 10% randomly picked samples are used for test and the others for training.

Baselines

In order to evaluate different encoding and representations, we benchmark different approaches with single input image sizes without ensembles, since we expect that the performance is likely to improve by assembling features or using multiple scales. We fix the input image size to 352×352 for SIFT, pre-trained CNNs feature extractions and Deep-TEN. FV-SIFT, a non-CNN approach, is considered due to its similar encoding representations. SIFT features of 128 dimensions are extracted from input images and a GMM of 128 Gaussian components is built, resulting in a 32K Fisher Vector

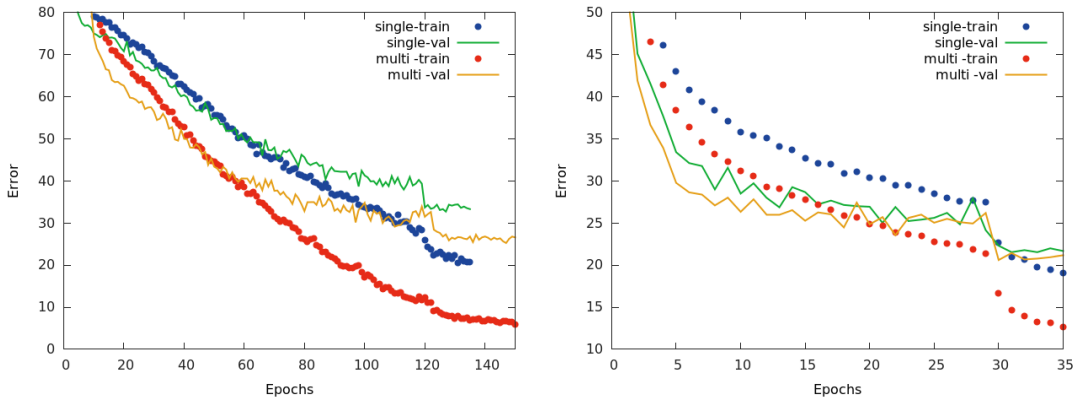


Figure 4.3: Comparison between single-size training and multi-size training. Iteratively training the Deep-TEN with two different intput sizes (352×352 and 320×320) makes the network converging faster and improves the performance. The top figure shows the training curve on MIT-Indoor and the bottom one shows the first 35 epochs on MINC-2500.

encoding. For FV-CNN encoding, the CNN features of input images are extracted using pre-trained 16-layer VGG-VD model [126]. The feature maps of conv5 (after ReLU) are used, with the dimensionality of 14×14×512. Then a GMM of 32 Gaussian components is built and resulting in a 32K FV-CNN encoding. To improve the results further, we build a stronger baseline using pre-trained 50-layers ResNet [127] features. The feature maps of the last residual unit are used. The extracted features are projected into 512 dimension using PCA, from the large channel numbers of 2048 in ResNet. Then we follow the same encoding approach of standard FV-CNN to build with ResNet features. For comparison with multi-size training Deep-TEN, multi-size FV-CNN (VD) is used, the CNN features are extracted from two different sizes of input image, 352×352 and 320×320 (sizes determined empirically). All the baseline encoding representations are reduced to 4096 dimension using PCA and $L2$-normalized. For classification, linear one-vs-all Support Vector Machines (SVM) are built using the off-the-shelf representations. The learning hyper-parameter is set to $C_{svm} = 1$, since the features are $L2$-normalized. The trained SVM classifiers are recalibrated as in prior work [1, 105], by scaling the weights and biases such that the median prediction score of positive and negative samples are at $+1$ and -1.

Deep-TEN Details

We build Deep-TEN with the architecture of an Encoding Layer on top of 50-layer pre-trained ResNet (as shown in Table 4.2). Due to high-dimensionality of ResNet feature maps on Res4, a 1×1 convolutional layer is used for reducing number of channels (2048⇒128). Then an Encoding Layer with 32 codewords is added on top, followed by $L2$-normalization and FC layer. The weights (codewords C and smoothing factor s) are randomly initialized with uniform distribution $\pm\frac{1}{\sqrt{K}}$. For data augmentation, the input images are resized to 400 along the short-edge with the per-pixel mean subtracted. For in-the-wild image database, the images are randomly cropped to 9% to 100% of the image areas, keeping the aspect ratio between 3/4 and 4/3. For the material database with in-lab or controlled conditions (KTH or GTOS), we keep the original image scale. The resulting images are then resized into 352×352 for single-size training (and 320×320 for multi-size training), with 50% chance horizontal flips. Standard color augmentation is used as in [94]. We use SGD with a mini-batch size of 64. For fine-tuning, the learning rate starts from 0.01 and divided by 10 when the error plateaus. We use a weight decay of 0.0001 and a momentum of 0.9. In testing, we adopt standard 10-crops [94].

Multi-size Training

Deep-TEN ideally can accept arbitrarily input image sizes (larger than a constant). In order to learn the network without modifying the standard optimization solver, we train the network with a pre-defined size in each epoch and iteratively change the input image size for every epoch as in [106]. A full evaluation of combinatorics of different size pairs have not yet been explored. Empirically, we consider two different sizes 352×352 and 320×320 during the training and only use single image size in testing for simplicity (352×352). The two input sizes result in 11×11 and 10×10 feature map sizes before feeding into the Encoding Layer. Our goal is to evaluate how multi-size training affects the network optimization and how the multi-scale features affect texture recognition.

4.4.1 Recognition Results

We evaluate the performance of Deep-TEN, FV-SIFT and FV-CNN on aforementioned golden-standard material and texture datasets, such as MINC-2500, FMD, KTH and two new material datasets: 4D-Light and GTOS. Additionally, two general recognition datasets MIT-Indoor and Caltech-101 are also considered. Table 4.3 shows overall experimental results using single-size training,

Comparing with Baselines

As shown in Table 4.3, Deep-TEN and FV-CNN always outperform FV-SIFT, which shows that pre-trained CNN features are typically more discriminant than hand-engineered SIFT features. FV-CNN usually achieves reasonably good results on different datasets without fine-tuning pre-trained features. We can observe that the performance of FV-CNN is often improved by employing ResNet features comparing with VGG-VD as shown in Table 4.4. Deep-TEN outperforms FV-CNN under the same settings, which shows that the Encoding Layer gives the advantage of transferring pre-trained features to material recognition by removing domain-specific information as described in Section 4.3. The Encoding Layer's property of representing feature distributions is especially good for texture understanding and segmented material recognition. Therefore, Deep-TEN works well on GTOS and KTH datasets. For the small-scale dataset FMD with less training sample variety, Deep-TEN still outperforms the baseline approaches that use an SVM classifier. For MINC-2500, a relatively large-scale dataset, the end-to-end framework of Deep TEN shows its distinct advantage of optimizing CNN features and consequently, the recognition results are significantly improved (61.8%⇒80.6% and 69.3%⇒81.3, compared with off-the-shelf representation of FV-CNN). For the MIT-Indoor dataset, the Encoding Layer works well on scene categorization due to the need for a certain level of orderless and invariance. The best performance of these methods for Caltech-101 is achieved by FV-CNN(VD) multi (85.7% omitted from the table). The CNN models VGG-VD and ResNet are pre-trained on ImageNet, which is also an object classification dataset like Caltech-101. The pre-trained features are discriminant

to target datasets. Therefore, Deep-TEN performance is only slightly better than the off-the-shelf representation FV-CNN.

Impact of Multi-size

For in-the-wild datasets, such as MINC-2500 and MIT-Indoor, the performance of all the approaches are improved by adopting multi-size as expected. Remarkably, as shown in Table 4.4, Deep-TEN shows a performance boost of 4.9% using multi-size training and outperforms the best baseline by 7.4% on MIT-Indoor dataset. For some datasets such as FMD and GTOS, the performance decreases slighltly by adopting multi-size training due to lack of variety in the training data. Figure 4.3 compares the single-size training and multi-size (two-size) training for Deep-TEN on MIT-Indoor and MINC-2500 dataset. The experiments show that multi-size training helps the optimization of the network (converging faster) and the learned multi-scale features are useful for the recognition.

Comparison with State-of-the-Art

As shown in Table 4.5, Deep-TEN outperforms the state-of-the-art on four material/texture recognition datasets: MINC-2500, KTH, GTOS and 4D-Light. Deep-TEN also performs well on two general recognition datasets. Notably, the prior state-of-the-art approaches either (1) relies on assembling features (such as FV-SIFT & CNNs) and/or (2) adopts an additional SVM classifier for classification. Deep-TEN as an end-to-end framework neither concatenates any additional hand-engineered features nor employe SVM for classification. For the small-scale datasets such as FMD and MIT-Indoor (subset), the proposed Deep-TEN gets compatible results with state-of-the-art approaches (FMD within 2%, MIT-indoor within 4%). For the large-scale datasets such as MINC-2500, Deep-TEN outperforms the prior work and baselines by a large margin demonstrating its great advantage of end-to-end learning and the ability of transferring pre-trained CNNs. We expect that the performance of Deep-TEN can scale better than traditional approaches when adding more training data.

	STL-10	CIFAR-10
Deep-TEN (Individual)	76.29	91.5
Deep-TEN (Joint)	**87.11**	91.8
State-of-the-Art	74.33 [128]	-

Table 4.6: Joint Encoding on CIFAR-10 and STL-10 datasets. Top-1 test accuracy %. When joint training with CIFAR-10, the recognition result on STL-10 got significantly improved. (Note that traditional network architecture does not allow joint training with different image sizes.)

4.4.2 Joint Encoding from Scratch

We test joint training on two small datasets CIFAR-10 [112] and STL-10 [113] as a litmus test of Joint Encoding from scratch. We expect the convolutional features learned with Encoding Layer are easier to transfer, and can improve the recognition on both datasets.

CIFAR-10 contains 60,000 tiny images with the size 32×32 belonging to 10 classes (50,000 for training and 10,000 for test), which is a subset of tiny images database. STL-10 is a dataset acquired from ImageNet [26] and originally designed for unsupervised feature learning, which has 5,000 labeled images for training and 8,000 for test with the size of 96×96. For the STL-10 dataset only the labeled images are used for training. Therefore, learning CNN from scratch is not supposed to work well due to the limited training data. We make a very simple network architecture, by simply replacing the 8×8 Avg pooling layer of pre-Activation ResNet-20 [129] with Encoding-Layer (16 codewords). We then build a network with shared convolutional layers and separate encoding layers that is jointly trained on two datasets. Note that the traditional CNN architecture is not applicable due to different image sizes from this two datasets. The training loss is computed as the sum of the two classification losses, and the gradient of the convolutional layers are accumulated together. For data augmentation in the training: 4 pixels are padded on each side for CIFAR-10 and 12 pixels for STL-10, and then randomly crop the padded images or its horizontal flip into original sizes 32×32 for CIFAR-10 and 96×96 for STL-10. For testing, we only evaluate the single view of the original images. The model is trained with a mini batch of 128 for each dataset. We start with a learning rate of 0.1 and divide it by 10 and 100 at 80^{th} and 120^{th} epoch.

The experimental results show that the recognition result of STL-10 dataset is significantly improved by joint training the Deep TEN with CIFAR-10 dataset. Our approach achieves the recognition rate of 87.11%, which outperforms previous the state of the art 74.33% [128] and 72.8% [130] by a large margin.

4.5 Conclusion

In summary, we developed an Encoding Layer and built the network Deep-TEN and demonstrated the effectiveness on various material and texture recognition datasets. we developed an Encoding Layer which bridges the gap between classic computer vision approaches and the CNN architecture. This layer has two main advantages: (1) the resulting deep learning framework is more flexible by allowing arbitrary input image size, and (2) the learned convolutional features are easier to transfer since the Encoding Layer is likely to carry domain-specific information. The Encoding Layer shows superior performance in transferring pre-trained CNN features. Deep-TEN outperforms traditional off-the-shelf methods and achieves state-of-the-art results on MINC-2500, KTH and two recent material datasets: GTOS and 4D-Lightfield.

The Encoding Layer is efficient using GPU computations and our Torch [131] implementation of 50-layer Deep-Ten (as shown in Table 4.2) takes the input images of size 352×352, runs at 55 frame/sec for training and 290 frame/sec for inference on 4 Titan X Maxwell GPUs.

Chapter 5

Multi-style Generative Network for Real-time Transfer

Overview

This chapter on multi-style generative network for real-time transfer is based on our paper [22]. Recent work in style transfer learns a feed-forward generative network to approximate the prior optimization-based approaches, resulting in real-time performance. However, these methods require training separate networks for different target styles which greatly limits the scalability. We introduce a Multi-style Generative Network (MSG-Net) with a novel Inspiration Layer, which retains the functionality of optimization-based approaches and has the fast speed of feed-forward networks. The proposed Inspiration Layer explicitly matches the feature statistics with the target styles at run time, which dramatically improves versatility of existing generative network, so that multiple styles can be realized within one network. The proposed MSG-Net matches image styles at multiple scales and puts the computational burden into the training. The learned generator is a compact feed-forward network that runs in real-time after training. Comparing to previous work, the proposed network can achieve fast style transfer with at least comparable quality using a single network. The experimental results have covered (but are not limited to) simultaneous training of twenty different styles in a single network. The complete software system and pre-trained models are publicly available upon publication[1].

Figure 5.1: The problem of multi-style transfer in real-time using a **single** network is solved in this thesis. Examples of transfered images and the corresponding styles.

5.1 Background

Style transfer can be approached as reconstructing or synthesizing texture based on the target image semantic content [132]. Many pioneering works have achieved success in classic texture synthesis starting with methods that resample pixels [133–136] or match multiscale feature statistics [137–139]. These methods employ traditional image pyramids obtained by handcrafted multiscale linear filter banks [140, 141] and perform texture synthesis by matching the feature statistics to the target style at multiple scales. In recent years, the concepts of texture synthesis and style transfer have been revisited within the context of deep learning. Feature histograms at pyramid levels in traditional methods are replaced with Gram Matrix representation from convolutional neural nets (CNN). Gatys *et al.* [142,143] first adopts a pre-trained CNN as a descriptive representation of image statistics and provides an explicit representation that separates

[1] https://github.com/zhanghang1989/MSG-Net

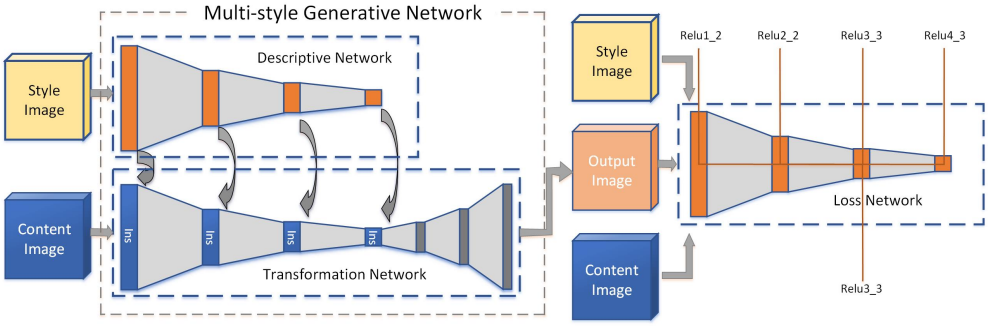


Figure 5.2: An overview of MSG-Net, Multi-style Generative Network. The transformation network as part of the generator explicitly matches the features statistics captured by a descriptive network of the style targets using the proposed Inspiration Layer denoted as Ins (introduced in Section 5.3). Detailed architecture of the transformation network is shown in Table 5.1. A pre-trained loss network provides the supervision of MSG-Net learning by minimizing the content and style differences with the targets as discussed in Section 5.4.2.

image style and content information. This framework has achieved great success in both texture synthesis and style transfer. This method is *optimization-based* because the new texture image is generated by applying gradient descent that manipulates a white noise image to match the Gram Matrix representation of the target image. Optimization-based approaches have no scalability problem but require expensive computation to generate images using gradient descent.

Recent work [16, 17] train a feed-forward generative network to approximate the optimization process that transforms the image into a target style in real-time by moving the computational burden into training process. However, these approaches require training separate networks for each different style, which extremely limits the scalability. Chen *et al.* [144] adopt a hybrid solution by separating mid-level convolutional filters individually for each style like a hard switch, and share the down-sampling and up-sampling parts across different styles. Nevertheless, the size of the network is still proportional to the number of styles, which will be problematic for hundreds of styles.

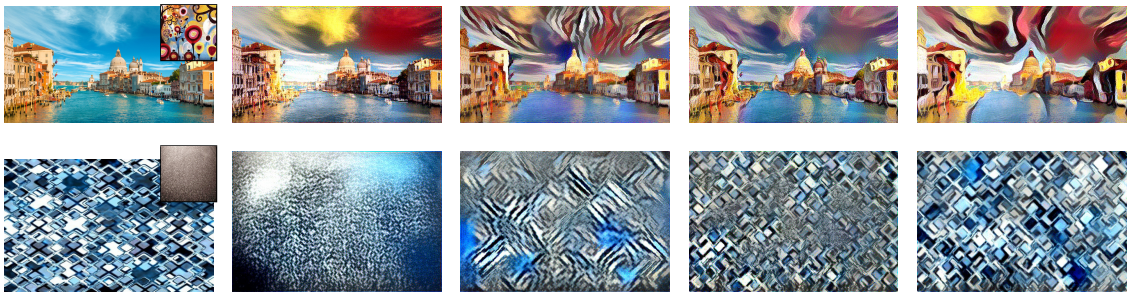
What limits the diversity of styles in existing generative network? Existing work learns a generative network taking an input image x_c and providing the transferred output $G(x_c)$, in which the feature statistics of the style image are implicitly learned from the loss function without informing the network about the style target x_s [16, 17, 145].

For existing approaches, there is a fundamental difficulty in deriving a representation that simultaneously preserves the semantic content of input image x_c and matches the style of the target image x_s (see extended discussion in Section 5.4.3). In order to build a multi-style generative network $G(x_c, x_s)$, where x_s can be chosen from a diverse set of styles, the generator network should explicitly match the feature statistics of the style target images at run time.

As the **first contribution** of the work, we introduce an *Inspiration Layer* which matches the feature statistics (Gram Matrix) of the target and preserves the input semantic contents at run time. The Inspiration Layer is end-to-end learnable with existing generative network architectures and puts the computational burden into the training process to achieve real-time style matching. The proposed Inspiration Layer enables multi-style generation from a single network which dramatically improves the versatility of existing generative network architectures.

The **second contribution** of this work is learning a novel feed-forward generative network for real-time multi-style matching, which we refer to as *Multi-style Generative Network (MSG-Net)*. The Inspiraton Layer is a component of MSG-Net. This proposed network explicitly matches the feature statistics at multiple scales in order to retain the performance of optimization-based methods and achieve the speed of feed-forward networks. The network design benefits from recent advances in CNN architecture called residual block [146], which reduces computational complexity without losing style versatility by preserving larger number of channels. We further design an *upsampling residual block* to allow passing identity all the way through the generative network, enabling the network to extend deeper and converge faster. The experimental results show that MSG-Net can achieve real-time style transfer with comparable image fidelity compared to previous work.

The work is organized as follows. We briefly describe the prior work on content and style representation using a CNN framework in Section 5.2. We introduce our proposed Inspiration Layer in Section 5.3 and our novel generative architecture Multi-style Generative Network in Section 5.4. The comparison to other approaches is discussed in Section 5.4.3. Finally, the experimental results and comparisons are presented in



(a) targets (b) scale 1 (c) scale 2 (d) scale 3 (e) all scales

Figure 5.3: Visualizing the effects of multi-scale feature statistics for style transfer (top) and texture synthesis (bottom). (First column) Input targets; (Center columns) Inverted representations at each individual scale; (Last Column) Inverted representation combining the multiple scales. We use pre-trained 16-layer VGG as the descriptive network and consider the size of 256×256 as the original scale and reduce the size by dividing 2^{i-1} for i-th scale. The featuremaps after ReLU at each scale are used.

Section 5.5.

5.2 Content and Style Representation

CNNs pre-trained on a very large dataset such as ImageNet can be regarded as descriptive representations of image statistics containing both semantic content and style information. Gatys *et al.* [143] provides explicit representations that independently model the image content and style from CNNs, which we briefly describe in this section for completeness.

The semantic content of the image can be represented as the activations of the descriptive network at i-th scale $\mathcal{F}^i(x) \in \mathbb{R}^{C_i \times H_i \times W_i}$ with a given the input image x, where the C_i, H_i and W_i are the number of feature map channels, feature map height and width. The texture or style of the image can be represented as the distribution of the features using Gram Matrix $\mathcal{G}(\mathcal{F}^i(x)) \in \mathbb{R}^{C_i \times C_i}$ given by

$$\mathcal{G}\left(\mathcal{F}^i(x)\right) = \sum_{h=1}^{H_i} \sum_{w=1}^{W_i} \mathcal{F}^i_{h,w}(x) \mathcal{F}^i_{h,w}(x)^T. \tag{5.1}$$

The Gram Matrix is orderless and describes the feature distributions. For zero-centered data, the Gram Matrix is the same as the covariance matrix scaled by the number of elements $C_i \times H_i \times W_i$. It can be calculated efficiently by first reshaping the feature map

$\Phi\left(\mathcal{F}^i(x)\right) \in \mathbb{R}^{C_i \times (H_i W_i)}$, where $\Phi()$ is a reshaping operation. Then the Gram Matrix can be written as $\mathcal{G}\left(\mathcal{F}^i(x)\right) = \Phi\left(\mathcal{F}^i(x)\right) \Phi\left(\mathcal{F}^i(x)\right)^T$.

5.3 Inspiration Layer

In this section, we introduce *Inspiration Layer*, which explicitly matches multi-scale feature statistics based on the given styles. For a given content target x_c and a style target x_s, the content and style representations at the i-th scale using the descriptive network can be written as $\mathcal{F}^i(x_c)$ and $\mathcal{G}(\mathcal{F}^i(x_s))$, respectively. A direct solution $\hat{\mathcal{Y}^i}$ is desirable which preserves the semantic content of input image and matches the target style feature statistics:

$$\begin{aligned} \hat{\mathcal{Y}^i} = &\underset{\mathcal{Y}^i}{\operatorname{argmin}}\{\|\mathcal{Y}^i - \mathcal{F}^i(x_c)\|_F^2 \\ &+\alpha\|\mathcal{G}(\mathcal{Y}^i) - \mathcal{G}\left(\mathcal{F}^i(x_s)\right)\|_F^2\}. \end{aligned} \tag{5.2}$$

where α is a trade-off parameter that balancing the contribution of the content and style targets.

The minimization of the above problem is solvable by using an iterative approach, but it is infeasible to achieve it in real-time or make the model differentiable. However, we can still approximate the solution and put the computational burden to the training stage. We introduce an approximation which tunes the feature map based on the target style:

$$\hat{\mathcal{Y}^i} = \Phi^{-1}\left[\Phi\left(\mathcal{F}^i(x_c)\right)^T W \mathcal{G}\left(\mathcal{F}^i(x_s)\right)\right]^T, \tag{5.3}$$

where $W \in \mathbb{R}^{C_i \times C_i}$ is a learnable weight matrix and $\Phi()$ is a reshaping operation to match the dimension, so that $\Phi\left(\mathcal{F}^i(x_c)\right) \in \mathbb{R}^{C_i \times (H_i W_i)}$. For intuition on the functionality of W, suppose $W = \mathcal{G}\left(\mathcal{F}^i(x_s)\right)^{-1}$, then the first term in Equation 5.2 is minimized. Now let $W = \Phi\left(\mathcal{F}^i(x_c)\right)^{-T} \mathcal{L}(\mathcal{F}^i(x_s))^{-1}$, where $\mathcal{L}\left(\mathcal{F}^i(x_s)\right)$ is obtained by the Cholesky Decomposition of $\mathcal{G}\left(\mathcal{F}^i(x_s)\right) = \mathcal{L}\left(\mathcal{F}^i(x_s)\right) \mathcal{L}\left(\mathcal{F}^i(x_s)\right)^T$, then the second term of Equation 5.2 is minimized. We let W be learned directly from the loss function to dynamically balance the trade-off.

End-to-end Learning

The Inspiration Layer is differentiable with respect to both layer input and the layer weights. Therefore, the Inspiration Layer can be learned by standard Stochastic Gradient Decent (SGD) solver. We provide explicit expression for the derived backpropagation equations in the supplementary material.

5.4 Multi-style Generative Network

5.4.1 Network Architecture

Existing feed-forward based style transfer work learns a generator network that takes only the content image as the input and outputs the transferred image, i.e. the generator network can be expressed as $G(x_c)$, which implicitly learns the feature statistics from the loss function. We introduce a *Multi-style Generative Network* which takes both content and style target as inputs. i.e. $G(x_c, x_s)$. The proposed network explicitly matches the feature statistics of the style targets at multiple scales.

As part of the Generator Network, we adopt a 16-layer pre-trained VGG network [126] as a descriptive network $\mathcal{F}$ which captures the feature statistics of the style image x_s at different scales, and outputs the Gram Matrices $\{\mathcal{G}(\mathcal{F}^i(x_s))\}(i = 1, ...K)$ where K is total number of scales. Then a transformation network takes the content image x_c and matches the feature statistics of the style image at multiple scales with Inspiration Layers. We adopt the VGG network that is pre-trained on ImageNet [147] as the descriptive network, because the network features learned from a diverse set of images are likely to be generic and informative.

Multi-scale Processing

Figure 5.3 illustrates the impact of multi-scale representation by comparing the reconstruction result from the feature statistics at individual scales with the result of combining multiple scales. We use a pre-trained 16-layer VGG as a descriptive network and use Gatys' approach to invert the representation [142]. This experiment suggests that feature statistics at individual scales are not informative enough to reconstruct

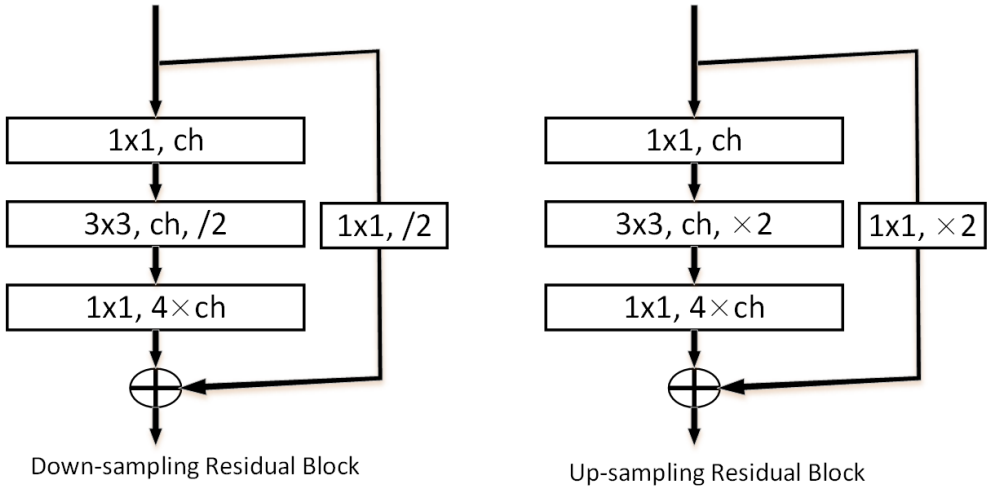


Figure 5.4: We extend the original down-sampling residual architecture (left) to an up-sampling version (right). We use a 1×1 fractionally-strided convolution as a shortcut and adopt reflectance padding.

textures or styles. Multi-scale feature statistics provide a comprehensive representation of the textures and styles. Therefore, we introduce a multi-scale Inspiration architecture to match the feature statistics at four different scales as shown in Figure 6.2.

Up-sample Residual Block.

Deep residual learning has achieved great success in visual recognition [129,146]. Residual block architecture plays an important role by reducing the computational complexity without losing diversity by preserving the large number of feature map channels. We extend the original architecture with an up-sampling version as shown in Figure 5.4 (right), which has a fractionally-strided convolution [148] as the shortcut and adopts reflectance padding to avoid artifacts of the generative process. This up-sampling residual architecture allows us to pass identity all the way through the network (as shown in Table 5.1), so that the network converges faster and extends deeper.

Other Details.

We only use in-network down-sample (convolutional) and up-sample (fractionally-strided convolution) in the transformation network as in previous work [17, 149]. We use reflectance padding to avoid artifacts at the border. Instance normalization [150] and

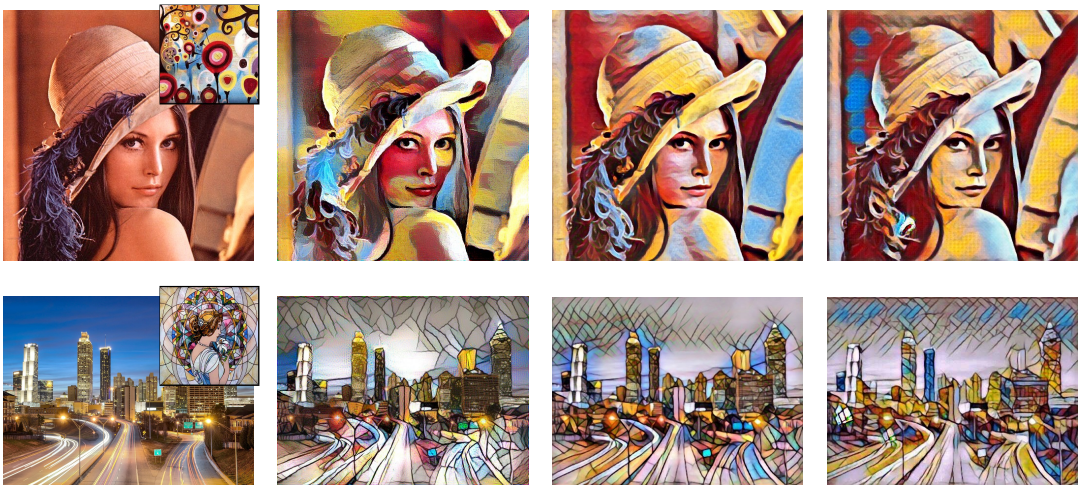
layer name	output size	MSG-Net
Conv1	256×256	7×7,64
Inspiration1	256×256	C=64
Res1	128×128	$\begin{bmatrix} 1\times1, 32 \\ 3\times3, 32 \\ 1\times1, 128 \end{bmatrix}\times$k
Inspiration2	128×128	C=128
Res2	64×64	$\begin{bmatrix} 1\times1, 64 \\ 3\times3, 64 \\ 1\times1, 256 \end{bmatrix}\times$k
Inspiration3	64×64	C=256
Res3	32×32	$\begin{bmatrix} 1\times1, 128 \\ 3\times3, 128 \\ 1\times1, 512 \end{bmatrix}\times$k
Inspiration4	32×32	C=512
Up-Res1	64×64	$\begin{bmatrix} 1\times1, 64 \\ 3\times3, 64 \\ 1\times1, 256 \end{bmatrix}\times$k
Up-Res2	128×128	$\begin{bmatrix} 1\times1, 32 \\ 3\times3, 32 \\ 1\times1, 128 \end{bmatrix}\times$k
Up-Res3	256×256	$\begin{bmatrix} 1\times1, 16 \\ 3\times3, 16 \\ 1\times1, 64 \end{bmatrix}\times$k
Conv	256×256	7×7,3

Table 5.1: The architecture of the transformation network with (18k+6) layers, which is the core part of Multi-style Generative Network (MSG-Net).

ReLU are used after weight layers (convolution, fractionally-strided convolution and the Inspiration Layer), which improves the generated image quality and is robust to the image contrast changes.

5.4.2 Network Learning

Style transfer is an open problem, since there is no gold-standard ground-truth to follow. We follow previous work to minimize a weighted combination of the style and content differences of the generator network outputs and the targets for a given pre-trained loss network $\mathcal{F}$ [16, 17]. Let the generator network be denoted by $G(x_c, x_s)$ parameterized by weights W_G. Learning proceeds by sampling content images $x_c \sim X_c$ and style



(a) input (b) Gatys [143] (c) Johnson [17] (d) MSG-Net(ours)

Figure 5.5: Comparison to existing approaches. Our proposed MSG-Net has dramatically improved the scalability of the generative network and achieves comparable results with existing work for each individual style.

images $x_s \sim X_s$ and then adjusting the parameters W_G of the generator $G(x_c, x_s)$ in order to minimize the loss:

$$\begin{aligned} \hat{W}_G = \underset{W_G}{\operatorname{argmin}} E_{x_c,x_s}\{ & \\ & \lambda_c \|\mathcal{F}^c\left(G(x_c, x_s)\right) - \mathcal{F}^c(x_c)\|_F^2 \\ & + \lambda_s \sum_{i=1}^{K} \|\mathcal{G}\left(\mathcal{F}^i(G(x_c, x_s))\right) - \mathcal{G}(\mathcal{F}^i(x_s))\|_F^2 \\ & + \lambda_{TV} \ell_{TV}\left(G(x_c, x_s)\right)\}, \end{aligned} \tag{5.4}$$

where λ_c and λ_s are the balancing weights for content and style losses. We consider image content at scale c and image style at scales $i \in \{1, ..K\}$. $\ell_{TV}()$ is the total variation regularization as used prior work for encouraging the smoothness of the generated images [17, 151, 152].

5.4.3 Relation to Other Methods

Relation to Pyramid Matching.

Early methods for texture synthesis were developed using multi-scale image pyramids [135, 137–139]. The discovery in these earlier methods was that realistic texture images

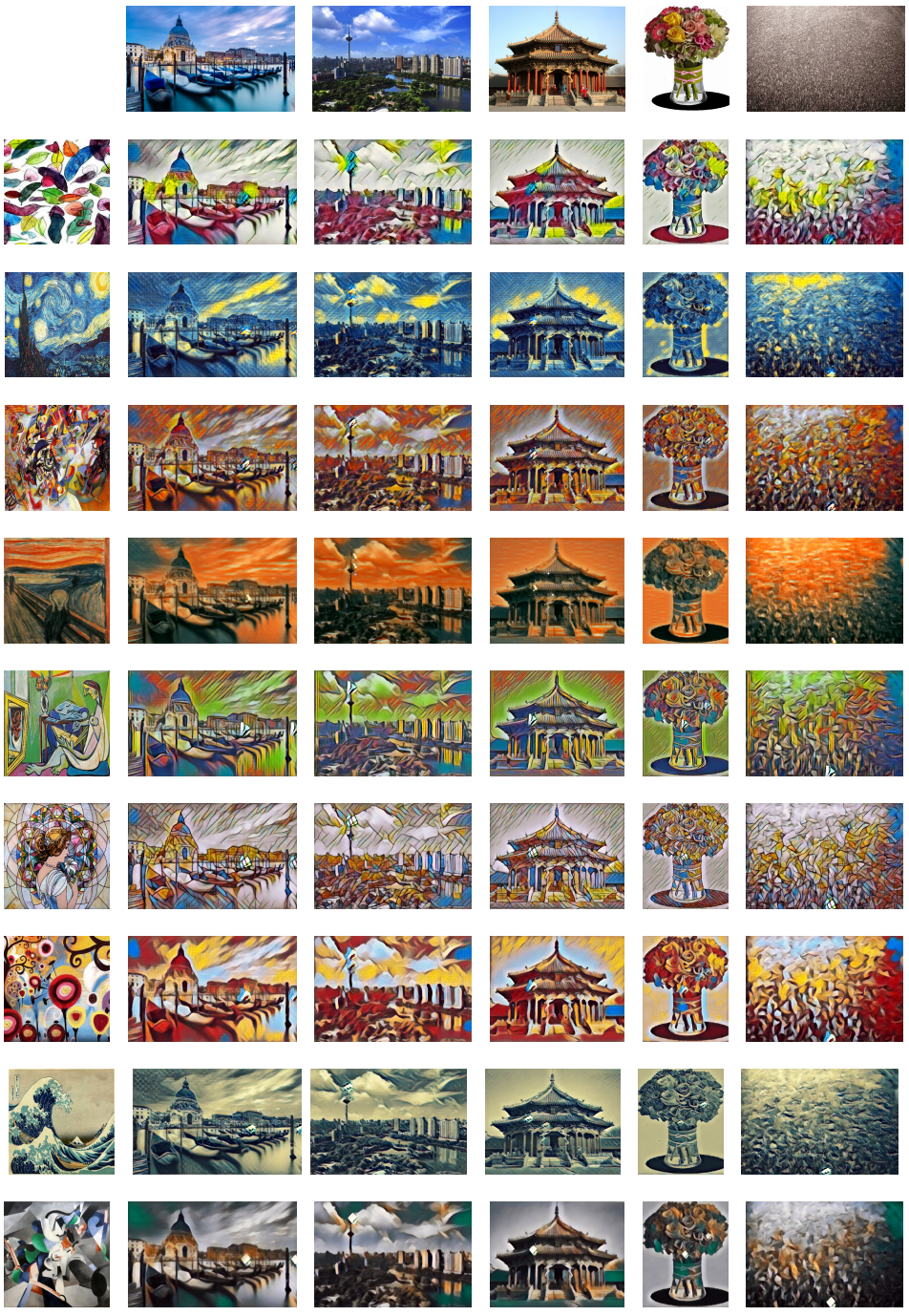


Figure 5.6: Diverse of images that are generated using a single MSG-Net. First row shows the input content images and the other rows are generated images with different style targets (first column).

could be synthesized from manipulating a white noise image so that its feature statistics were matched with the target at each pyramid level. Our approach is inspired by classic methods, which matches feature statistics at multiple image scales, but it leverages the advantages of deep learning networks while placing the computational costs into the training process.

Relation to Fusion Layers

Our proposed Inspiration Layer is a kind of fusion layer that takes two inputs (content and style representations). Current work in fusion layers with CNNs include feature map concatenation and element-wise sum [153–155]. However, these approaches are not directly applicable, since there is no separation of style style from content. For style transfer, the generated images should not carry semantic information of the style target nor styles of the content image. In addition, input representation size must match in prior fusion methods; but for style transfer, the content representation has the dimension of $C_i \times H_i \times W_i$ and the orderless style representation (Gram Matrix) has the dimension of $C_i \times C_i$.

Relation to Generative Networks and Adversarial Training

Generative Adversarial Network (GAN) [156], which jointly trains an adversarial generator and discriminator simultaneously, has catalyzed a surge of interest in the study of image generation [149, 153, 154, 157, 158]. Recent work on image-to-image GAN [153] adopts a conditional GAN to provide a general solution for some image-to-image generation problems. For those problems, it was previously hard to define a loss function. However, the style transfer problem cannot be tackled using the conditional GAN framework, due to missing ground-truth image pairs. Instead, we follow the work [16, 17] to adopt a discriminator / loss network that minimizes the perceptual difference of synthesized images with content and style targets and provides the supervision of the generative network learning. The initial idea of employing Gram Matrix to trigger the styles synthesis is inspired by a recent work [157] that suggests using an encoder instead of random vector in GAN framework.

	Storage	Training Time	Test Time
Optimization Based [132, 143]	$\mathcal{O}(1)$	N/A	slow
Feed-forward [16, 17, 144]	$\mathcal{O}(N)$	$\mathcal{O}(N)$	real-time
MSG-Net (ours)	$\mathcal{O}(1)$	$\mathcal{O}(N)$	real-time

Table 5.2: Compared to existing methods, MSG-Net has the benefit of real-time style-transfer of feed-forward based approaches as well as the scalability of classic approaches.

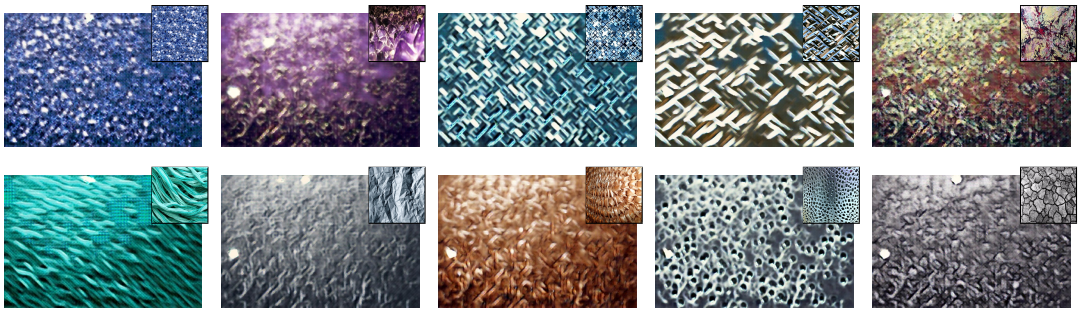


Figure 5.7: Texture synthesis examples that are generated using a single MSG-Net and the corresponding texture targets.

Concurrent with our work

Concurrent work [3, 159, 160] explores arbitrary style transfer. A style swap layer is proposed in [159], but gets lower quality and slower speed (compared to existing feed-forward approaches). An adaptive instance normalization is introduced in [3] to match the mean and variance of the feature maps with the style target. Instead, our Inspiration Layer matches the second order statistics of Gram Matrices for the feature maps at multiple scales. We also explore applying our method to new styles (not seen during training) in Figure 5.8.

5.5 Experimental Results

In this section, we make qualitative comparison of the proposed MSG-Net with existing approaches for style transfer task. We consider the gold-standard optimization based work of Gatys *et al.* [143] and the state-of-the-art feed-forward approach of Johnson *et al.* [17] with Instance Normalization [150]. Additionally, we show MSG-Net can be applied to texture synthesis task.

5.5.1 Style Transfer

Baselines.

We adopt a publicly available implementation [161] of Gatys *et al.* [143] as a golden standard baseline of optimization based approaches. Given the content image x_c, style image x_s and a pre-trained 16-layer VGG [126] as a descriptive network $\mathcal{F}$. Considering the content reconstruction at c-th scale and the style reconstruction at scales $i \in \{1, ...K\}$, an image $\hat{y}$ is initialized with white noise and updated by iteratively minimizing the objective function:

$$\begin{aligned}\hat{y} = \underset{y}{\operatorname{argmin}}\{&\lambda_c \|\mathcal{F}^c(y) - \mathcal{F}^c(x_c)\|_F^2 \\ &+ \lambda_s \sum_{i=1}^{K} \|\mathcal{G}(\mathcal{F}^i(y)) - \mathcal{G}(\mathcal{F}^i(x_s))\|_F^2 \\ &+ \lambda_{TV} \ell_{TV}(y)\}.\end{aligned} \tag{5.5}$$

The optimization is performed using L-BFGS solver for 500 iterations. The method is slow due to forwarding and backwarding the image y at each iteration.

We also compare our approach with an improved version of recent feed-forward work [17] using instance normalization [150] as the state-of-the-art approach, where we train each feed-forward network individually for different style images using the loss function same as Equation 5.4.

Method Details.

16-layer VGG network [126] are used in the descriptive network as part of the MSG-Net and the loss network in Equation 5.4. For both networks, we consider the style representation at 4 different scales using the layers ReLU1_2, ReLU2_2, ReLU3_3 and ReLU4_3. For loss network, we consider the content representation at the layer ReLU2_2. The Microsoft COCO dataset [117] is used as the content image image set X_c, which has around 80,000 natural images. We collect 20 style images, choosing those that are typically used in previous work. 42-layer MSG-Net is used in this experiment ($k = 2$ as shown in Table 5.1). We follow the work [16,17] and adopt Adam [162] to train the network with a learning rate of 1×10^{-3}. For learning the network, we use the loss function

Figure 5.8: Test a model with the style targets that have **NOT** been covered during the training.

as described in Equation 5.4 with the balancing weights $\lambda_c = 1, \lambda_s = 5, \lambda_{TV} = 1 \times 10^{-6}$ for content, style and total regularization. We resize the content images $x_c \sim X_c$ to 256×256 and learn the network with a batch size of 4 for 4,000$\times N_{style}$ iterations. We iteratively update the style image x_s every 20 iterations[2] with size of 512×512[3]. After training, the MSG-Net can accept arbitrary input image size, and we resize the input images to 512 along the long edge before fed into the network during the test in this experiment. Our implementation is based on Torch [163], which takes roughly 8 hours for training 20-style MSG-Net model on a single Titan X Pascal GPU, which is 10 times faster than Johnson's approach for the same number of styles because the joint optimization across different styles benefits from each other.

Qualitative Comparison

We keep the same hyper parameters for MSG-Net and baselines, such as balancing weights. For optimization based Gatys' approach [143], we stop the optimization after 500 iterations that is typically more than enough. The feed-forward baseline model using Johnson's approach [17] is learned for 40,000 iterations for each style model as same as in the original paper. We train the proposed 20-style MSG-Net for 80,000 iterations (4,000$\times N_{style}$). A standard histogram matching is added as a post-processing

[2]The number of 20 is not empirically chosen, we did a grid search varying from 4 to 20 and chose the one with best quality.

[3]We use a large style image, which provides more texture details and improves the quality compared to the size of 256×256.

to all the approaches, which adds a slight improvement of the perceived color. Figure 5.5 shows the comparison of three approaches using popular pictures of Lena and Atlanta city with two popular styles. We can see that the optimization based approach is more colorful and has sharper textures than feed-forward approaches (Johnson's and MSG-Net), such as the buildings and the roads in the pictures of Atlanta (bottom row). Feed-forward approaches have the advantages of preserving semantic consistencies, such as human face and hairs in the picture of Lena (top row), because the models are trained on MS-COCO dataset which contains a lot of context information of real-word images. More examples of the transfered images using MSG-Net are shown in Figure 5.6. In general, our proposed MSG-Net dramatically improve the scalability of the network for style transfer and has at least comparable quality comparing to existing work.

5.5.2 Texture Synthesis

The texture synthesis can be regarded as a special case of style transfer, in which the content image is not involved and the goal is to reconstruct the textures. In this section, we explore how our approach can be applied on texture synthesis task. 10 textures are selected from Describable Texture Dataset (DTD) [164] as the targets. We adopt the same network and training strategy as in style transfer task. We follow previous work [16, 145] and feed the MSG-Net with random noise to trigger the texture synthesis. Li et al. [145] suggests that using Brown noise that containing spectrum of frequencies produces the textures with better quality than using white noise. We further discover that triggering the network using a structured noise that containing both different frequencies and various intensities results in textures with even better quality. Examples of texture synthesis are shown in Figure 5.7 and the right column of Figure 5.6

5.6 Conclusion and Discussion

The problem of multi-style transfer in real-time using a single network has been addressed in this paper. We tackle the technical difficulties of existing approaches by

introducing a novel Inspiration Layer, which explicitly matches the target styles at run time. We have demonstrated that the Inspiration Layer embedded in our proposed Multi-style Generative Network enables 20 styles transfer without losing quality. The proposed MSG-Net puts the computational burden of matching feature statistics in the training process, which enables real-time transfer. It runs at 17.8 frame/sec for the input image of size 256×256 on a single Titan X Pascal GPU.

However, dealing with unknown style is still an unsolved problem for feed-forward approaches. The strategy of putting burden into training limits the performance on unknown style images as shown in Figure 5.8. This can be potentially solved by large varieties of training styles and better style representation model, so that the interpolation between different styles can be learned by the generator network.

Chapter 6

Photo-realistic Facial Texture Transfer

6.1 Overview

This section on Photo-realistic Facial Texture Transfer is based on a joint work with Parneet Kaur [23]. Style transfer methods have achieved significant success in recent years with the use of convolutional neural networks. However, many of these methods concentrate on artistic style transfer with few constraints on the output image appearance. We address the challenging problem of transferring face texture from a style face image to a content face image in a photorealistic manner without changing the identity of the original content image. Our framework for face texture transfer (FaceTex) augments the prior work of MRF-CNN with a novel facial semantic regularization that incorporates a *face prior regularization* smoothly suppressing the changes around facial meso-structures (e.g eyes, nose and mouth) and a *facial structure loss function* which implicitly preserves the facial structure so that face texture can be transferred without changing the original identity. We demonstrate results on face images and compare our approach with recent state-of-the-art methods. Our results demonstrate superior texture transfer because of the ability to maintain the identity of the original face image.

6.2 Background

Recent work in texture synthesis and style transfer has achieved great success using convolutional neural networks [15, 165]. Despite the success of artistic style transfer, facial style transfer remains challenging due to the requirement of photo-realism and semantic consistency. Human vision is very sensitive to facial irregularities and even small distortions can make a face look unrealistic [166, 167]. In this work, we address

the problem of photo-realistic facial style transfer, which transfers *facial texture* from a new style image while preserving most of the original *facial structure* and identity (Figure 6.1). Facial texture comprises skin texture details like wrinkles, pigmentation and pores, while facial structure consists of the meso-structures such as eyes, nose, mouth and face shape. Our approach has important implications in commercial applications and dermatology, such as visualizing the effects of age, sun exposure, or skin treatments (e.g. anti-aging, acne).

Style transfer of artistic work is typically approached by synthesizing a style texture based on the semantic content of the input image [133–136]. Classic algorithms match the feature statistics of multi-scale representations [137–139]. Gatys et al. [165, 168] first adopted a pre-trained CNN [169] as a statistical feature representation to provide an explicit representation of image content and style. The output image is generated by solving an optimization problem which minimizes both content and style differences and iteratively passes the gradient directly to the image pixels. Recent work also explores real-time style transfer by training feed-forward networks while approximating the optimization process which outputs the style transferred images directly [170–172], and has been extended to multi-style [22, 173].

Despite the rapid growth of artistic style transfer work, photo-realistic facial style transfer remains challenging due to the need of preserving local semantic consistency while transferring skin texture. The Gram matrix is often used as a gold-standard style representation. Minimizing the difference of a global representation of Gram matrix does not sufficiently enforce local semantic consistency at meso-structures such as lower facial contour, eyes and mouth as shown in Figure 6.3 (last column). A recent method [174] incorporates the Gram matrix with semantic segmentation and achieves high quality results for photo-realistic style transfer in scene images. This approach removes distortions in architectural scenes but is not designed for facial texture transfer and has no mechanism for retaining facial structure. Our approach is developed with the specific goal of maintaining the content face identity.

Markov random field (MRF) models have been used widely for representing image

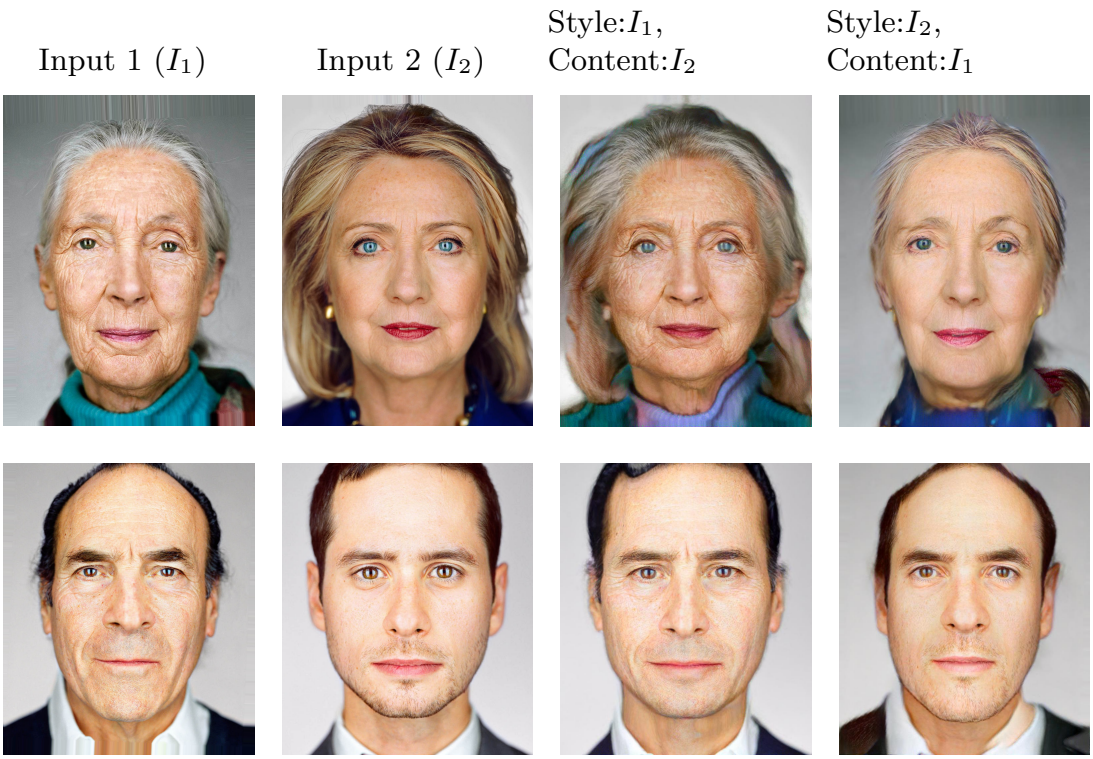


Figure 6.1: Identity-preserving Facial Texture Transfer (FaceTex). The textural details are transferred from style image to content image while preserving its identity. FaceTex outperforms existing methods perceptually as well as quantitatively. Column 3 uses input 1 as the style image and input 2 as the content. Column 4 uses input 1 as the content image and input 2 as the style image. Figure 6.3 shows more examples and comparison with existing methods. Input photos: Martin Scheoller/Art+Commerce.

texture [175] by modeling the image statistic at a pixel or patch level and the dependence between neighbors. Classic texture synthesis methods using MRF [133] [135] provide new texture instances using an MRF texture model. A recent work called MRF-CNN [171] leverages the local representation of MRF and the descriptive power of CNN for style transfer. However, this method also transfers meso-structures from the style image. For faces, this facial structure sourced from the style image leads to an undesirable change in facial identity during the texture transfer as in Figure 6.3 (column 3).

As the **first contribution** of this work, we introduce *Facial Semantic Regularization* that consists of a *Facial Prior Regularization* and *Facial Structural Loss* for preserving identity during the texture transfer. Facial identity incorporates facial structure and

shapes. We suppress the changes around the meso-structures by introducing the Facial Prior Regularization that smoothly slows down the updating. Additionally, we tackle the challenge of preserving facial shape by minimizing a Facial Structure Loss which we define as an identity loss from a pre-trained face recognition network that implicitly preserves the facial structure.

The **second contribution** of this work is the development of an algorithm for *Identity Preserving Facial Texture Transfer* which we call *FaceTex* along with a complete benchmark of facial texture transfer with a novel metric for quantitative evaluation. Our approach augments the MRF-CNN framework with the Facial Semantic Regularization and faithfully transfers facial textures and preserves the facial identity. We provide a complete benchmark that evaluates style transfer algorithms on facial texture transfer task. Prior methods typically rely on perceptual evaluation of results, which makes it difficult to quantitatively compare them. We propose metrics that quantify the facial structure consistency as well as texture similarity. The experimental results show that the proposed FaceTex outperforms the existing approaches for identity-preserving texture transfer perceptually as well as quantitatively.

6.3 Methods

6.3.1 Texture Representation

We follow prior work of MRF-CNN [171] for texture representation and briefly describe it for completeness [171]. A pre-trained VGG-19 [169] is used as a descriptive representation of image statistics, and the feature-maps at layer l for input image x is denoted as $\Phi^l(x)$. For a given content image x_c and a style image x_s, the facial texture is transferred from x_s to the output/target image x_t by minimizing the difference of local patches. Let $\Psi(\Phi^l(x))$ denote the set of the local patches on the featuremaps. For each patch $\Psi_i(\Phi^l(x_t))$, the difference with the most similar patch in the style image $\Psi_{NN(i)}(\Phi^l(x_s))$ (among N_s patches) is minimized. The distance of the nearest neighbor is defined using normalized cross-correlation as

$$NN(i) = \underset{j=1,\dots N_s}{\operatorname{argmin}} \frac{\Psi_i(\Phi^l(x_t))\Psi_j(\Phi^l(x_s))}{|\Psi_i(\Phi^l(x_t))| \cdot |\Psi_j(\Phi^l(x_s))|}. \tag{6.1}$$

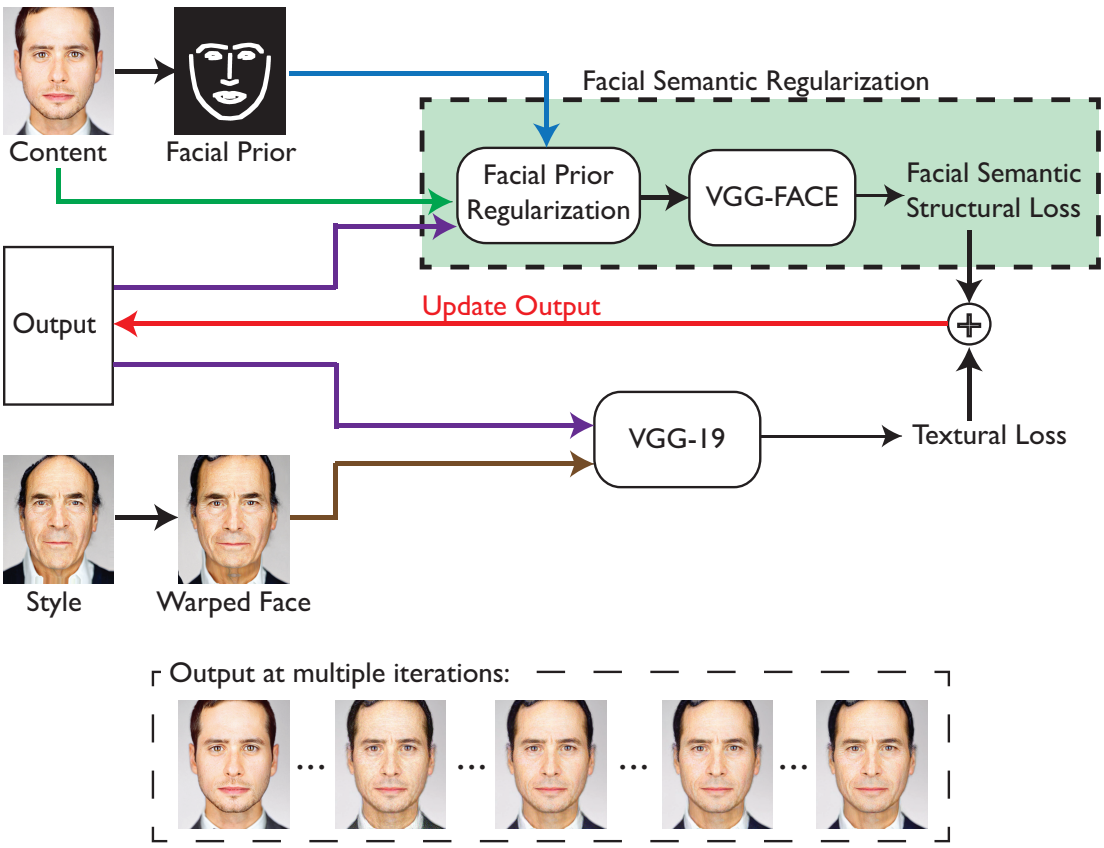


Figure 6.2: Overview of our method. Facial identity is preserved using Facial Semantic Regularization which regularizes the update of meso-structures using a facial prior and facial semantic structural loss. Texture loss regularizes the update of local textures from the style image. The output image is initialized with the content image and updated at each iteration by back-propagating the error gradients for the combined losses. Content/style photos: Martin Scheoller/Art+Commerce.

The texture loss is the sum of the difference for all the N_t patches in the generated image and is given by

$$\ell^l_{tex}(x_t, x_s) = \sum_{i=1}^{N_t} \|\Psi_i(\Phi^l(x_t)) - \Psi_{NN(i)}(\Phi^l(x_s))\|^2. \tag{6.2}$$

In contrast to the Gram Matrix that gives global impact to the image, MRF-CNN is good for preserving local textural structures. However, it also carries the semantic information from the style image, which violates the goal of preserving facial identity. For this, we augment the MRF-CNN framework with additional regularizations.

6.3.2 Facial Semantic Regularization

Facial identity consists of meso-structures including eyes, noise, eyebrow, lips and face contour. We tackle the problem of preserving facial identity by suppressing local changes around these meso-structures and minimizing the identity loss from face recognition network, which implicitly preserves the semantic facial structure.

Facial Prior Regularization

Inspired by the dropout regularization [176] which randomly drops some units and blocks the gradient during the optimization, we build a facial prior regularization that smoothly slows down the updating around the meso-structures. For generating the facial prior mask, we follow the prior work [177] to generate 66 landmark points and draw contours for meso-structures. Then we build a landmark mask by applying a Gaussian blur to the facial contour and normalize the output between 0 and 1, which provides a smooth transition between meso-structures and rest of the face. For implementation, we build a CNN layer that performs an identity mapping during the forward pass of the optimization, and scales the gradient with an element-wise product with the face prior mask during back-propagation.

Facial Semantic Structure Loss

Deep learning is well known for learning hierarchical representations directly from data. Instead of manually tackling preservation of facial structure, we minimize the perceptual difference of a face recognition network to force the output image to be recognized as the same person depicted in the input/content image. VGG-Face [178] is trained on millions of faces and has superior discriminative power for face recognition, which captures the facial meso-structures for identifying the person. Instead of minimizing the final classification error, we minimize the difference of mid-level feature-maps, because the mid-level features are already discriminative for preserving facial identity. Let $\delta^i(x)$ denote the feature-maps at a i-th layer of a pre-trained VGG-Face for input image x.

The structure loss is the L^2-distance of the feature-maps and is given by

$$\ell_{face}(x_t, x_c) = \sum_{i=1}^{N_l} \frac{1}{C_i H_i W_i} \|\delta(x_t) - \delta(x_c)\|^2, \tag{6.3}$$

where N_l is total number of layers for calculating structure loss, and C_i, H_i and W_i are the number of channels, height and width of the feature-map, respectively.

6.3.3 Identity Preserving Facial Texture Transfer

Pre-processing

To maintain facial structural consistency and avoid artifacts, we warp the style image to the facial structure of the content image. First, 66 facial landmark points are generated for the content and style images using an existing facial landmark detection algorithm [177]. The style image is then morphed and aligned to the content image [179]. To further align the face contour we apply sift-flow, which uses dense SIFT feature correspondences for alignment while preserving spacial discontinuities [180].

Loss functions

Reconstructing the image from the loss of highly abstracted pre-trained networks makes the image look unrealistic and noisy. We follow the prior work [22,170,171] which uses total variation regularization (TV loss) to encourage the smoothness of the output image x, which is given by the squared norm of the gradients:

$$\ell_{TV}(x) = \sum_{i,j} \left((x_{i,j+1} - x_{i,j})^2 + (x_{i+1,j} - x_{i,j})^2 \right). \tag{6.4}$$

We use a weighted combination of texture loss, facial structure loss and TV loss to find the output estimate $\hat{x_t}$ as follows

$$\hat{x_t} = \underset{x_t}{\operatorname{argmin}} \sum_{l=1}^{L} \lambda_{tex}^l \ell_{tex}^l(x_t, x_s) + \lambda_{face} \ell_{face}(x_t, x_c) + \lambda_{TV} \ell_{TV}(x_t), \tag{6.5}$$

where L is total number of layers for texture loss and λ_{tex}^l, λ_{face} and λ_{TV} are the balancing weights for texture loss, facial structure loss and TV loss. The optimization is performed by manipulating the the content image x_c by iteratively updating the image pixels using an L-BFGS solver.

6.4 Experimental Results

6.4.1 Facial Style Transfer Benchmark

Baseline Approaches. We use the publicly available implementation of *Neural Style transfer* for comparison [165,181]. Gatys et al. [165] generates an output image x_t from content image x_c and style image x_s by jointly minimizing the content loss and the style loss iteratively. The content loss is given by the L^2-distance of the feature-maps at each convolution layers for the output and the content images. The style loss is the Frobenius norm of the Gram matrix difference of the feature-maps of output image and the style images at each layer. The weighted combination of the losses is minimized to obtain the output image as

$$\hat{x_t} = \underset{x_t}{\operatorname{argmin}} \lambda_c ||\Phi(x_t) - \Phi(x_c)||^2 + \lambda_s \sum_{l=1}^{L} ||\mathcal{G}^l(x_t) - \mathcal{G}^l(x_s)^l||_F^2 + \lambda_{TV} \ell_{TV}(x_t), \quad (6.6)$$

where $\Phi(x_t)$ and the $\Phi(x_c)$ are the feature-maps of output and style images, $\mathcal{G}^l(x_t)$ and $\mathcal{G}^l(x_s)$ are the Gram matrices of the feature-maps of output and style images at layer l; L is the total number of layers; λ_c, λ_s and λ_{TV} are the weights for content loss, style loss and TV loss. In these experiments, we use $\lambda_c = 5$, $\lambda_s = 100$ and $\lambda_{TV} = 10^{-3}$. We use the L-BFGS solver for 1000 iterations. VGG-19 [169] pre-trained network is used for computing feature-maps. Layer relu4_2 is used for content loss while layers relu1_1,relu2_1,relu3_1,relu4_1 and relu5_1 are used for style loss.

We also compare our work with it MRF-CNN [171]. The output image is generated by minimizing the patch difference with the style image and preserving the high-level structure the same as in the content image. The loss function consists texture loss, content loss and TV losses:

$$\hat{x_t} = \underset{x_t}{\operatorname{argmin}} \lambda_c ||\Phi(x_t) - \Phi(x_c)||^2 + \lambda_s^l \sum_{l=1}^{L} \ell_{tex}^l(x_t, x_s) + \lambda_{TV} \ell_{TV}(x_t), \quad (6.7)$$

where $\ell_{tex}(x_t, x_s)$ is the texture loss as in Section 6.3.1, λ_c, λ_s and λ_{TV} are the wights for content loss, style loss and TV loss. Layers relu3_1 and relu4_1 of VGG-19 are used for texture loss and layer relu4_2 for content loss. Neural patches of size 3×3 are used to find the best matching patch. Three resolutions with 100 iterations each are used.

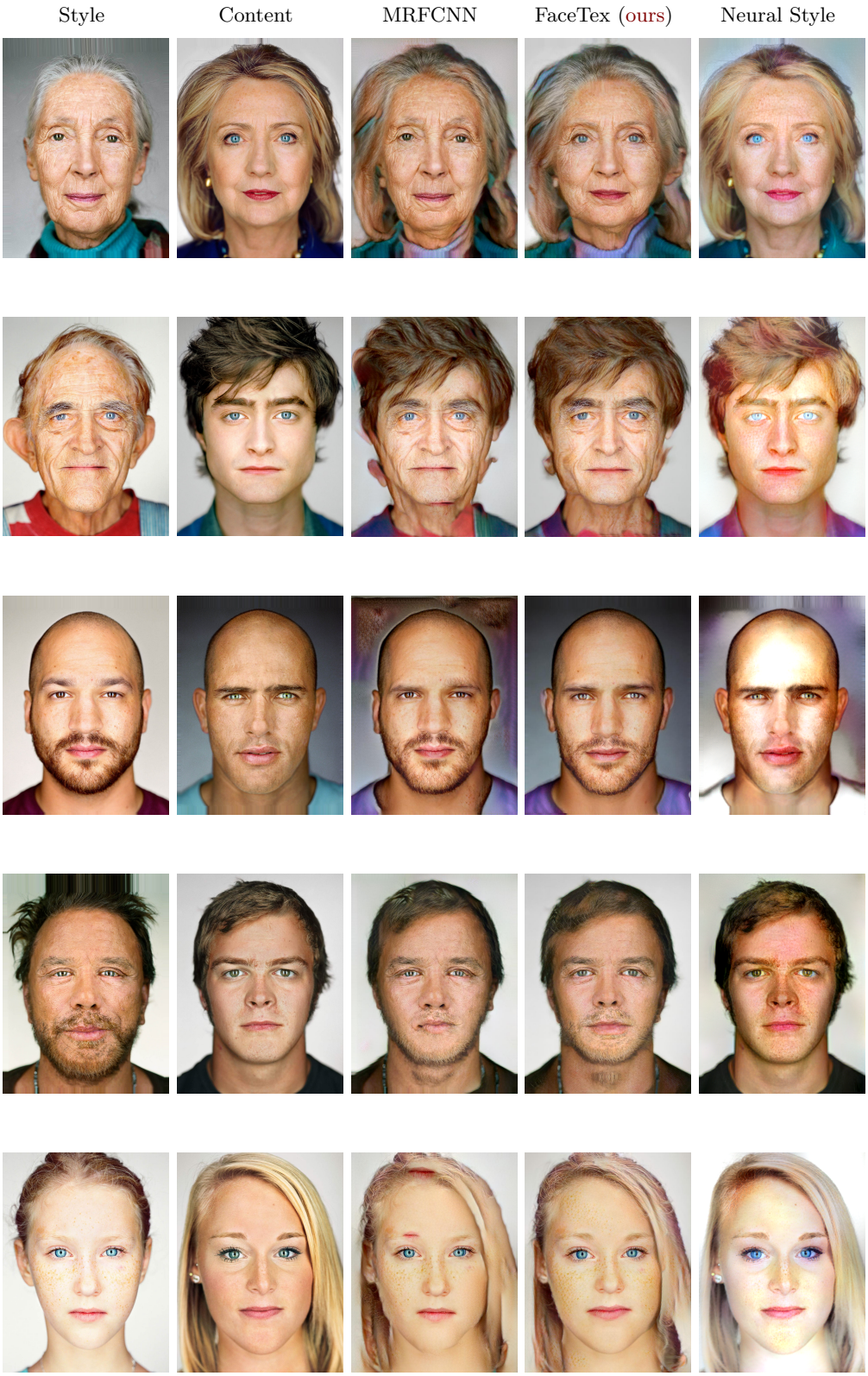


Figure 6.3: Our facial texture transfer on different content-style pairs. FaceTex (our approach, 4th column) preserves the identity of the content image and also transfers the textural details from the style image. Neural style (last column) preserves the identity but does not transfer the textural details. MRFCNN (3rd column) transfers the textural details but does not preserve the content image identity as well as FaceTex (compare 3rd and 4th column to content image in 2nd column). Content/style photos: Martin Scheoller/Art+Commerce.

Implementation Details. We follow the work of MRF-CNN using layers relu3_1 and relu4_1 of VGG-19 [169] for texture loss. Layer relu4_2 of a pre-trained VGG-Face [178] is used for facial semantic structure loss. The facial prior mask is generated by connecting the landmark points using 40 pixel thickness line and applying a Gaussian blurring with the kernel size of 65 and standard divination of 30. In addition, the background mask provided in the dataset is also used. We incorporate facial prior regularization to block the changes of facial prior and background regions. We resize the content and style images to $1,000$ pixels along the long edge. The output image is initialized with the content image and the optimization is performed using the L-BFGS solver. We follow Li and Wand [171] using a multi-resolution process during the generation, the content and style images are scaled accordingly. We start with $\frac{1}{4}$ resolution and scale up by a factor of 2, and perform 200 iterations at each resolution. We use the same resolution for both baselines and our approach in this experiment.

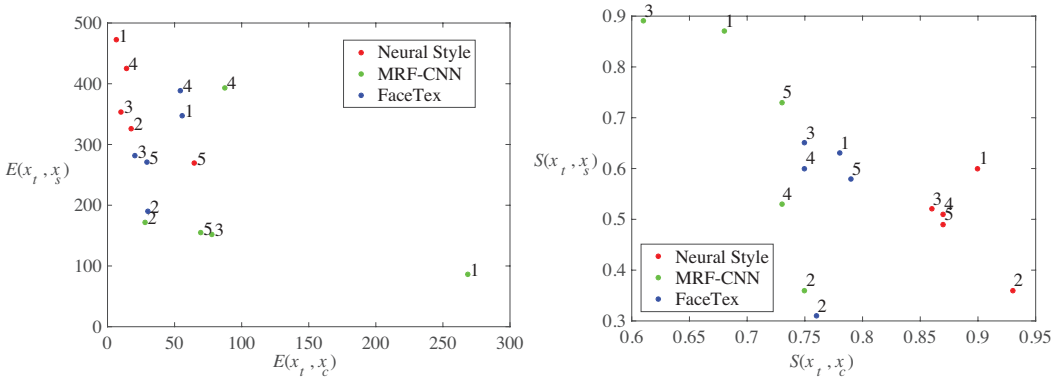
Metrics for Quantitative Evaluation. We identify two metrics to quantitatively measure the facial structural inconsistency and texture similarity of the output image x_t with the content image x_c and the style image x_s .

Landmark Error: Using the methods described in section 6.3.3, we obtain $L = 66$ landmarks for each facial image. The output image has same facial structure if its landmark points remain the same as content image. The mean square error of the landmarks between the two images accounts for the facial structural inconsistency between them. Lower error indicates identity is preserved. The landmark error between two facial images is given by the L^2 distance of the pixel coordinates for the landmark points.

Texture Correlation: To measure the similarity between the output image and the input images, we can extract skin patches from the images and use the *normalized correlation coefficient*. Higher value of correlation coefficient indicates a better match of facial textures. Texture similarity of two patches p and q is given by:

$$S(p,q) = \frac{\sum_{ij}(p_{ij} - \bar{p})(q_{ij} - \bar{q})}{\sqrt{\sum_{i,j}(p_{ij} - \bar{p})^2 \sum_{i,j}(q_{i,j} - \bar{q})^2}}, \tag{6.8}$$

where p_{ij} and q_{ij} are the image values at pixel coordinates (i, j) of the patches, $\bar{p}$ and



(a) (b)

Figure 6.4: Quantitative evauation for each content-style pair. (a) Landmark Error. For FaceTex, $E(x_t, x_c)$ is small and much closer to Neural Style than MRF indicating that it preserves identity as in Neural Style approach. (b) Texture similarity. For FaceTex, $S(x_t, x_s)$ is high and closer to MRF-CNN, transferring tecture from style image.

$\bar{q}$ are the average pixel values of patches p and q.

6.4.2 Qualitative Comparison

We use the head portrait dataset provided by Shih *et al.* [182] for evaluation. Figure 6.3 shows the comparison of the output image generated using FaceTex with Neural Style and MRF-CNN. We provide additional comparison in the supplementary material. We observe that Neural Style preserves the facial structure and shape well but fails to transfer the texture, which demonstrates that the Gram Matrix transfers global styles well but fails to preserve the local finer texture and also makes the image unrealistic. MRF-CNN transfers local texture very well but it does not preserve the meso-structures which leads to more significant change the observed facial identity. Our proposed FaceTex approach generates photo-realistic images and outperforms all the baseline approaches in transferring facial texture as well as preserving the facial identity.

The quantitative comparison matches the conclusion of qualitative observation, and the results of landmark and texture metrics are listed in Table 6.1. The average values of different metrics are reported for the content-style pairs in Figure 6.3. $E(x_t, x_c)$ and

$E(x_t, x_s)$ are the landmark errors of output image with content and style images, respectively. $E(x_t, x_c)$ is very low for Neural Style (22.61) but high for MRF-CNN (106.17) indicating that identity is preserved by the Neural Style approach. For FaceTex, $E(x_t, x_c)$ is much lower (37.93) than MRF-CNN and preserves the identity. In contrast, higher value of $E(x_t, x_s)$ indicates that facial structural similarity is not maintained with the style image as expected. $S(x_t, x_c)$ and $S(x_t, x_s)$ are the texture similarities of output image with content and style images, respectively. For each content-style pair, we extract three patches and report their average normalized correlation coefficient as the texture similarity of the image. These patches are 100×100 and in forehead and both cheek regions, localized by face structure landmarks. For texture transfer, a large value of $S(x_t, x_s)$ indicates that texture is successfully transferred to output image from the style image. $S(x_t, x_s)$ is highest for MRF-CNN (0.68) and lowest for Neural Style (0.47), whereas for FaceTex similarity lies between MRF-CNN and Neural Style (0.55).

Figure 6.4(a) and (b) shows the landmark errors and texture similarity for each of the five content-style pairs in Figure 6.3. Both the error and the similarity measures for FaceTex (blue dots) lie between Neural Style (red dots) and MRF-CNN (green dots), and generally much closer to MRF-CNN.

Ablation Experiments. Figure 6.5 exemplifies the necessity of augmenting the existing methods with multiple regularizations. If only the facial prior regularization is used, the generated output face still looses identity and has artifacts. Adding the facial semantic structure further preserves the identity and suppresses some artifacts.

Limitations. Our method achieves superior performance in identity preserving facial texture transfer and generates photo-realistic images, but still has its limitations. First, our approach is an optimization-based approach, which takes several minutes generating a new image, which limits the applications in real-time. This could be potentially addressed in the future work by combining a feed-forward network and a face alignment network that run in real-time. Second, the texture modeling using MRF-CNN requires high semantic similarity between two input images, which may lead to some unappealing artifacts for mismatches.

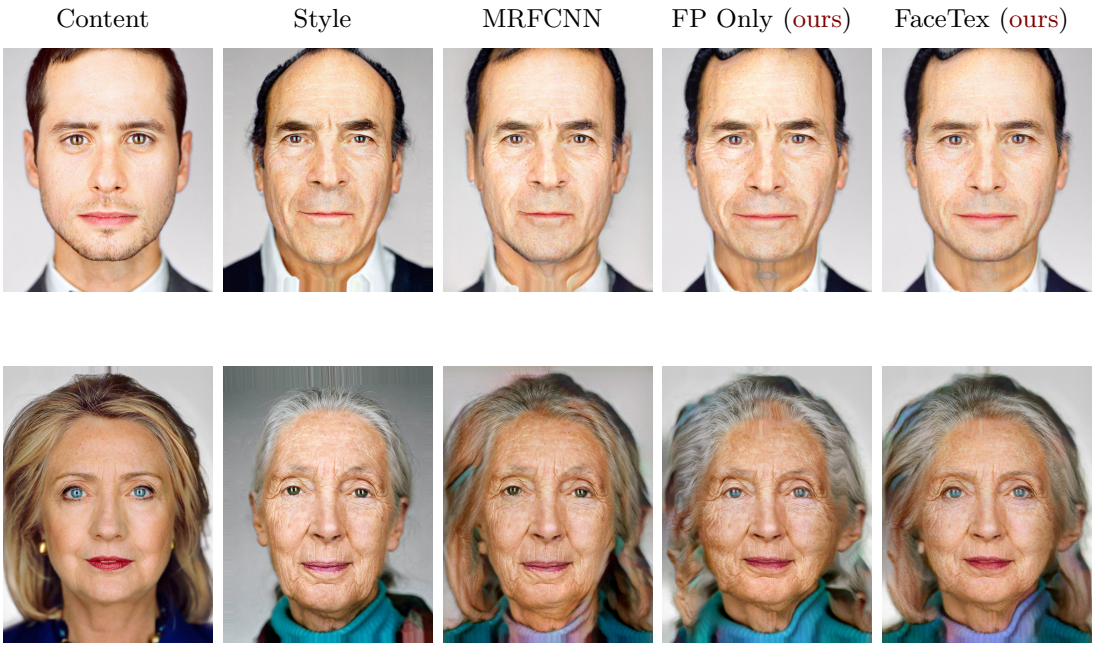


Figure 6.5: The effects of facial prior (FP) regularization and facial semantic structure loss. Using FP regularization (column 4) preserves better meso-structure of the faces comparing to MRF-CNN (column 3). Facial semantic loss effectively preserve the facial structure for identity preserving as shown in the last column. Content/style photos: Martin Scheoller/Art+Commerce.

6.5 Conclusion

We have presented the method FaceTex for photo-realistic facial style transfer. By augmenting prior work of MRF-CNN with a novel regularization consisting of a facial prior regularization and the facial semantic structure loss, we can transfer texture realistically while retaining semantic structure so that the identity of the individual remains recognizable. Our results show substantial improvement over the state-of-the-art both in the quality of the texture transfer and the preservation of the original face structure. Quantitative metrics of texture transfer and face structure are also improved using this approach.

	Neural Style	**MRF-CNN**	**FaceTex (ours)**
$E(x_t, x_c)$	22.61	106.17	37.93
$E(x_t, x_s)$	369.39	191.47	295.93
$S(x_t, x_c)$	0.89	0.70	0.76
$S(x_t, x_s)$	0.47	0.68	0.55

Table 6.1: Metrics for quantitative evaluation. The average metric values of the pairs in Figure 6.3 are reported here. For FaceTex, landmark error between output and content $E(x_t, x_c)$ is much lower than MRF-CNN indicating it is better at preserving identity. Texture similarity between output and style $S(x_t, x_s)$ is higher in FaceTex than Neural Style which shows that it is better in transferring texture.

Chapter 7

Conclusions

7.1 Summary of Contributions

This thesis is dedicated to developing compact and robust material and texture representations for visual recognition, retrieval and synthesis. This thesis explores angular and spatial variation of reflectance which give rise to reflectance models for recognition and texture models for both recognition and synthesis. We have focused on: 1) compact reflectance measurement and encoding using angular gradients for material recognition and tactile properties prediction, 2) deep texture networks to capture spatial variation in texture for robust recognition, 3) scalable generative networks for end-to-end synthesis of novel texture.

We have demonstrated the effectiveness and the efficiency of the techniques theoretically and practically. Specifically, we have developed the following methods:

Reflectance Hashing. We have introduced a novel framework for measurement and recognition of material classes. The approach encodes high dimensional reflectance with a compact binary code. We have compared several existing binary code methods to choose the most appropriate for this material recognition approach. The coding supports discrimination among the classes and can be realized in embedded vision implementations. Our method of reflectance hashing is compared to two popular baseline texton-based methods, boosting and texton histograms for material recognition.

Deep Reflectance Code. We have presented a framework of reflectance codes to encode sampled reflectance functions. There are three main contributions: 1) first of its kind friction-from-reflectance method, 2) reflectance codes for material recognition and

3) a new combination of FV-CNN with binary embedding that can be applied to image-based material recognition as well as reflectance disks. The database of reflectance disks and corresponding friction measurements will be made publicly available for future research.

Deep TEN. We have developed an Encoding Layer and built the network Deep-TEN and demonstrated the effectiveness on various material and texture recognition datasets. we developed an Encoding Layer which bridges the gap between classic computer vision approaches and the CNN architecture. This layer has two main advantages: (1) the resulting deep learning framework is more flexible by allowing arbitrary input image size, and (2) the learned convolutional features are easier to transfer since the Encoding Layer is likely to carry domain-specific information. The Encoding Layer shows superior performance in transferring pre-trained CNN features. Deep-TEN outperforms traditional off-the-shelf methods and achieves state-of-the-art results on MINC-2500, KTH and two recent material datasets: GTOS and 4D-Lightfield.

MSG-Net. The problem of multi-style transfer in real-time using a single network has been addressed in this thesis. We tackle the technical difficulties of existing approaches by introducing a novel Inspiration Layer, which explicitly matches the target styles at run time. We have demonstrated that the Inspiration Layer embedded in our proposed Multi-style Generative Network enables 20 styles transfer without losing quality. The proposed MSG-Net puts the computational burden of matching feature statistics in the training process, which enables real-time transfer.

Face-Tex We have presented the method FaceTex for photo-realistic facial style transfer. By augmenting prior work of MRF-CNN with a novel regularization consisting of a facial prior regularization and the facial semantic structure loss, we can transfer texture realistically while retaining semantic structure so that the identity of the individual remains recognizable. Our results show substantial improvement over the state-of-the-art both in the quality of the texture transfer and the preservation of the original face structure.

7.2 Future Work

There are several challenging topics remaining in the material and texture modeling field. We discuss interesting and promising topics that we will pursue in the future work in the following:

Reflectance Encoding. Reflectance naturally encodes rich information about the material surface, including what it is made of and how it is shaped. In this thesis, we have explored how can we leverage reflectance for material recognition and tactile property prediction, in particular, friction coefficient prediction. Can we go beyond the problem of recognition and prediction to leverage the reflectance modeling for material rendering? It is interesting to explore surface material manipulation using a given or captured reflectance type, and even combine with our proposed compact reflectance measurement for real-time synthesis.

End-to-end Texture Model. The Deep TEN architecture has achieved state-of-the-art material and texture recognition results in some gold-standard public benchmarks, but material and texture modeling is still a challenging and less studied field in computer vision. There are many remaining difficulties to be addressed in the future: 1) how to leverage the global semantic information to improve the classification using the local feature statistics, 2) how can material recognition be beneficial to other vision tasks, such as scene understanding, object recognition and detection. The proposed Encoding Layer captures the local feature statistics from the convolutional featuremaps, which is powerful for material and texture representations but simultaneously removes the global information. However, the global semantic information can be a valuable clue for the material types, for example we may not expect an indoor material such as carpet to appear in an outdoor scene e.g. near the beach. Therefore, an approach combining local feature statistics with global semantics is desired.

Texture Synthesis and Style Transfer. This thesis has achieved multi-style transfer in real-time using a single compact feed-forward network, which leverage the

fast speed of feed-forward network with the flexibility of the transitional optimization-based approaches. However, this end-to-end solution is not compatible with some standard features from successful classic approaches, such as color matching and brush-size control. Real-time performance, output quality and framework flexibility is the Holy Grail of neural stylization at the moment. Even though a few prior works have achieved real-time performance, but they still require a powerful GPU. Further boosting the performance for mobile application is interesting for future work. In addition, extending the flexibility of the current framework is a promising and challenging topic to study, such as style mixing, color preservation, scale/brush-size control which hasn't been addressed in existing feed-forward approaches.

Appendix A

A.1 Encoding Layer Implementations

This appendix section provides the explicit expression for the gradients of the loss ℓ with respect to ($w.r.t$) the layer input and the parameters for implementing Encoding Layer as introduced in Chapter 4. The $L2$-normalization as a standard component is used outside the encoding layer.

Gradients w.r.t Input X

The encoder $E = \{e_1, ...e_K\}$ can be viewed as k independent sub-encoders. Therefore the gradients of the loss function ℓ $w.r.t$ input descriptor x_i can be accumulated $\frac{d_\ell}{d_{x_i}} = \sum_{k=1}^{K} \frac{d_\ell}{d_{e_k}} \cdot \frac{d_{e_k}}{d_{x_i}}$. According to the chain rule, the gradients of the encoder $w.r.t$ the input is given by

$$\frac{d_{e_k}}{d_{x_i}} = r_{ik}^T \frac{d_{a_{ik}}}{d_{x_i}} + a_{ik} \frac{d_{r_{ik}}}{d_{x_i}}, \tag{A.1}$$

where a_{ik} and r_{ik} are defined in Sec 4.2, $\frac{d_{r_{ik}}}{d_{x_i}} = 1$. Let $f_{ik} = e^{-s_k \|r_{ik}\|^2}$ and $h_i = \sum_{m=1}^{K} f_{im}$, we can write $a_{ik} = \frac{f_{ik}}{h_i}$. The derivatives of the assigning weight $w.r.t$ the input descriptor is

$$\frac{d_{a_{ik}}}{d_{x_i}} = \frac{1}{h_i} \cdot \frac{d_{f_{ik}}}{d_{x_i}} - \frac{f_{ik}}{(h_i)^2} \cdot \sum_{m=1}^{K} \frac{d_{f_{im}}}{d_{x_i}}, \tag{A.2}$$

where $\frac{d_{f_{ik}}}{d_{x_i}} = -2 s_k f_{ik} \cdot r_{ik}$.

Gradients w.r.t Codewords C

The sub-encoder e_k only depends on the codeword c_k. Therefore, the gradient of loss function $w.r.t$ the codeword is given by $\frac{d_\ell}{d_{c_k}} = \frac{d_\ell}{d_{e_k}} \cdot \frac{d_{e_k}}{d_{c_k}}$.

$$\frac{d_{e_k}}{d_{c_k}} = \sum_{i=1}^{N} (r_{ik}^T \frac{d_{a_{ik}}}{d_{c_k}} + a_{ik} \frac{d_{r_{ik}}}{d_{c_k}}), \tag{A.3}$$

where $\frac{d_{r_{ik}}}{d_{c_k}} = -1$. Let $g_{ik} = \sum_{m \neq k} f_{im}$. According to the chain rule, the derivatives of assigning $w.r.t$ the codewords can be written as

$$\frac{d_{a_{ik}}}{d_{c_k}} = \frac{d_{a_{ik}}}{d_{f_{ik}}} \cdot \frac{d_{f_{ik}}}{d_{c_k}} = \frac{2 s_k f_{ik} g_{ik}}{(h_i)^2} \cdot r_{ik}. \tag{A.4}$$

Gradients w.r.t Smoothing Factors

Similar to the codewords, the sub-encoder e_k only depends on the k-th smoothing factor s_k. Then, the gradient of the loss function $w.r.t$ the smoothing weight is given by $\frac{d_\ell}{d_{s_k}} = \frac{d_\ell}{d_{e_k}} \cdot \frac{d_{e_k}}{d_{s_k}}$.

$$\frac{d_{e_k}}{d_{s_k}} = -\frac{f_{ik} g_{ik} \|r_{ik}\|^2}{(h_i)^2} \tag{A.5}$$

Note

In practice, we multiply the numerator and denominator of the assigning weight with e^{ϕ_i} to avoid overflow:

$$a_{ik} = \frac{\exp(-s_k \|r_{ik}\|^2 + \phi_i)}{\sum_{j=1}^{K} \exp(-s_j \|r_{ij}\|^2 + \phi_i)}, \tag{A.6}$$

where $\phi_i = \min_k \{s_k \|r_{ik}\|^2\}$. Then $\frac{d_{\bar{f}_{ik}}}{d_{x_i}} = e^{\phi_i} \frac{f_{ik}}{d_{x_i}}$.

A Torch [131] implementation is provided in supplementary material and available at https://github.com/zhanghang1989/Deep-Encoding.

A.2 Multi-size Training-from-Scratch

We also tried to train Deep-TEN from-scratch on MINC-2500 , the result is omitted in the main paper due to having inferior recognition performance comparing with employing pre-trained ResNet-50. As shown in Figure A.1, the converging speed is significantly improved using multi-size training, which proves our hypothesis that multi-size training helps the optimization of the network. The validation error is less improved than the training error, since we adopt single-size test for simplicity.

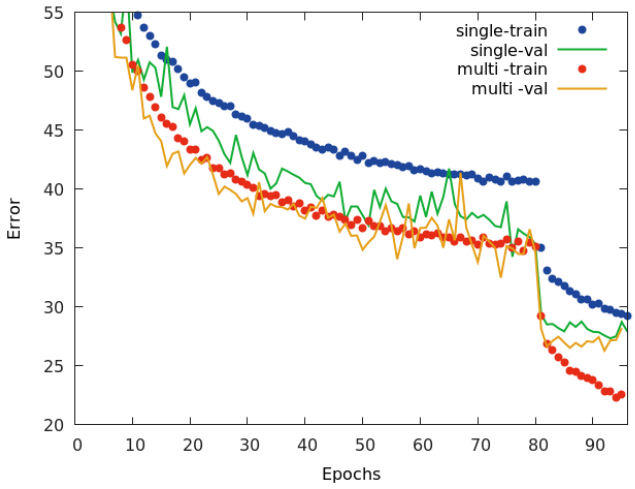


Figure A.1: Training Deep-TEN-50 from-scratch on MINC-2500 (first 100 epochs). Multi-size training helps the optimization of Deep-TEN and improve the performance.

A.3 Additional Results for MSG-Net

We make comparison with a concurrent work Huang *et al.* [3] using the same examples of the main thesis. Figure A.3 shows that both approaches achieve good perceptive quality, but MSG-Net handles the target styles better than the other work. Especially for transferring the textures, MSG-Net preserves both the colors and textures in the style images, while the other work scatters mosaic-like textures to each image. However, this comparison is not inferring that our approach is better, but showing the differences of the applications and motivations of these two works. Huang *et al.* is aiming at arbitrary styles, while our proposed MSG-Net is targeting at a set of specific styles and transferring them without losing quality compared with existing approaches.

Additionally, we include the example results for another 11 styles in Figure A.4 that are omitted form Figure 5.6 of the main thesis due to space constraints (20 styles in total).

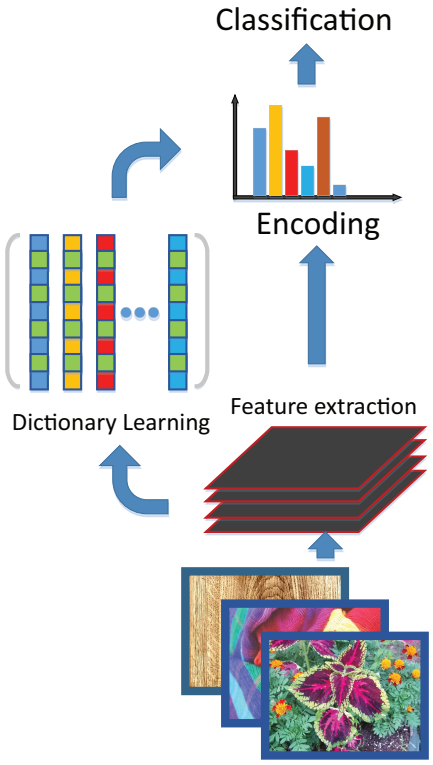


Figure A.2: Pipelines of classic computer vision approaches. Given in put images, the local visual appearance is extracted using hang-engineered features (SIFT or filter bank responses). A dictionary is then learned off-line using unsupervised grouping such as K-means. An encoder (such as BoWs or Fisher Vector) is built on top which describes the distribution of the features and output a fixed-length representations for classification.

(a) Inputs (b) MSG-Net(ours) (c) Huang [3]

Figure A.3: Comparison with Huang *et al.* [3].

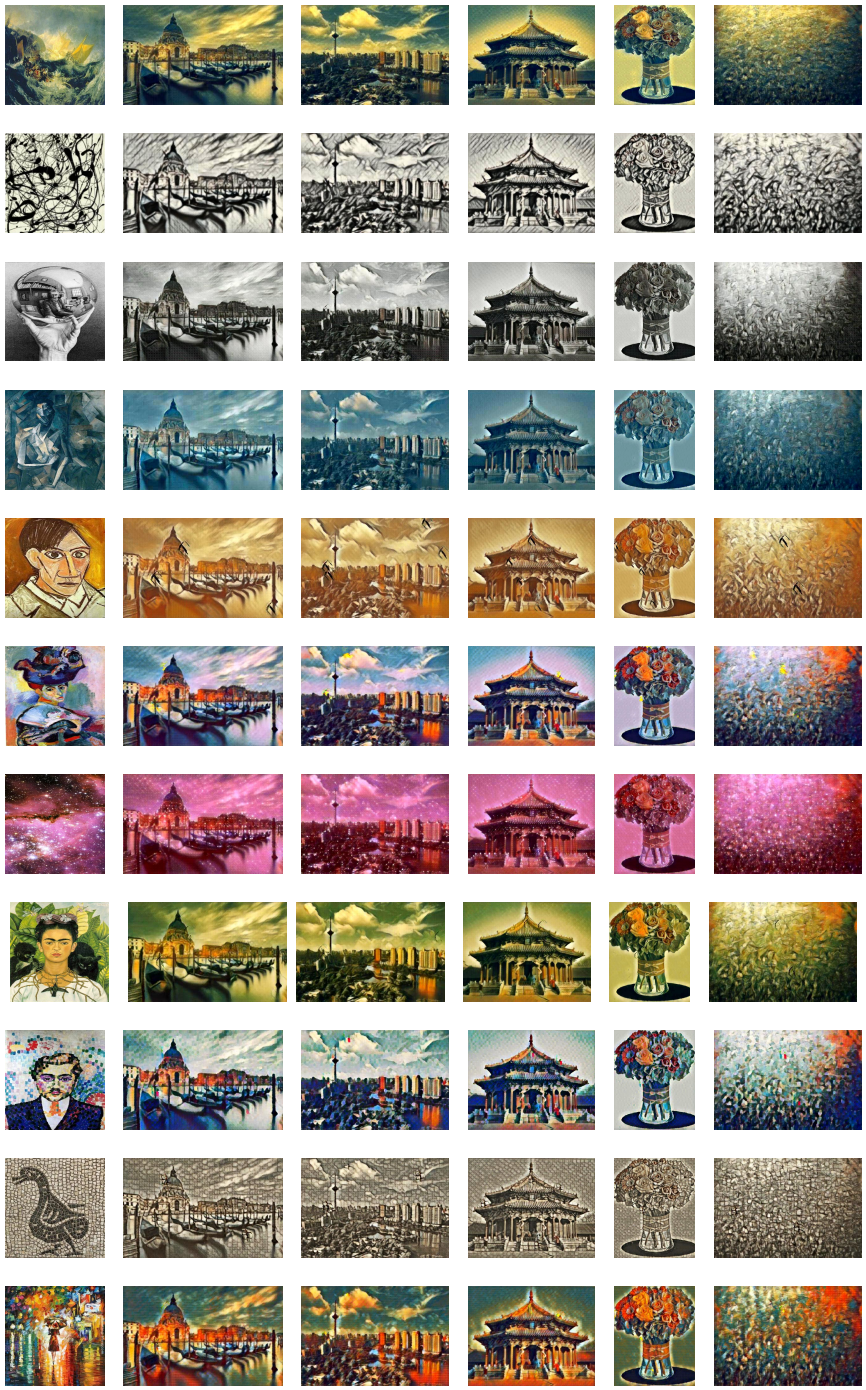


Figure A.4: Additional examples for figure 5.6. Diverse of images that are generated using a single MSG-Net. First row shows the input content images and the other rows are generated images with different style targets (first column).

References

[1] M. Cimpoi, S. Maji, and A. Vedaldi, "Deep filter banks for texture recognition and segmentation," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 3828–3836, 2015.

[2] Y. Gong, L. Wang, R. Guo, and S. Lazebnik, "Multi-scale orderless pooling of deep convolutional activation features," in *Computer Vision–ECCV 2014*, pp. 392–407, Springer, 2014.

[3] X. Huang and S. Belongie, "Arbitrary style transfer in real-time with adaptive instance normalization," *arXiv preprint arXiv:1703.06868*, 2017.

[4] H. Jégou, M. Douze, C. Schmid, and P. Pérez, "Aggregating local descriptors into a compact image representation," in *Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference on*, pp. 3304–3311, IEEE, 2010.

[5] F. Perronnin, J. Sánchez, and T. Mensink, "Improving the fisher kernel for large-scale image classification," in *Computer Vision–ECCV 2010*, pp. 143–156, Springer, 2010.

[6] D. G. Lowe, "Distinctive image features from scale-invariant keypoints," *International journal of computer vision*, vol. 60, no. 2, pp. 91–110, 2004.

[7] O. G. Cula and K. J. Dana, "Compact representation of bidirectional texture functions," *IEEE Conference on Computer Vision and Pattern Recognition*, vol. 1, pp. 1041–1067, December 2001.

[8] O. G. Cula and K. J. Dana, "Recognition methods for 3d textured surfaces," *Proceedings of SPIE Conference on Human Vision and Electronic Imaging VI*, vol. 4299, pp. 209–220, January 2001.

[9] T. Leung and J. Malik, "Representing and recognizing the visual appearance of materials using three-dimensional textons," *International journal of computer vision*, vol. 43, no. 1, pp. 29–44, 2001.

[10] M. Varma and A. Zisserman, "Classifying images of materials: Achieving viewpoint and illumination independence," in *European Conference on Computer Vision*, pp. 255–271, Springer, 2002.

[11] T. Joachims, "Text categorization with support vector machines: Learning with many relevant features," in *European conference on machine learning*, pp. 137–142, Springer, 1998.

[12] G. Csurka, C. Dance, L. Fan, J. Willamowski, and C. Bray, "Visual categorization with bags of keypoints," in *Workshop on statistical learning in computer vision, ECCV*, vol. 1, pp. 1–2, Prague, 2004.

[13] J. Sivic, B. C. Russell, A. A. Efros, A. Zisserman, and W. T. Freeman, "Discovering objects and their location in images," in *Tenth IEEE International Conference on Computer Vision (ICCV'05) Volume 1*, vol. 1, pp. 370–377, IEEE, 2005.

[14] L. Fei-Fei and P. Perona, "A bayesian hierarchical model for learning natural scene categories," in *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05)*, vol. 2, pp. 524–531, IEEE, 2005.

[15] L. A. Gatys, A. S. Ecker, and M. Bethge, "Texture synthesis using convolutional neural networks," in *Proceedings of the 28th International Conference on Neural Information Processing Systems*, NIPS'15, (Cambridge, MA, USA), pp. 262–270, MIT Press, 2015.

[16] D. Ulyanov, V. Lebedev, A. Vedaldi, and V. Lempitsky, "Texture networks: Feed-forward synthesis of textures and stylized images," in *Int. Conf. on Machine Learning (ICML)*, 2016.

[17] J. Johnson, A. Alahi, and L. Fei-Fei, "Perceptual losses for real-time style transfer and super-resolution," in *European Conference on Computer Vision*, 2016.

[18] C. Li and M. Wand, "Combining markov random fields and convolutional neural networks for image synthesis," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 2479–2486, 2016.

[19] H. Zhang, K. Dana, and K. Nishino, "Reflectance hashing for material recognition," *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 3071–3080, 2015.

[20] H. Zhang, K. Dana, and K. Nishino, "Friction from reflectance: Deep reflectance codes for predicting physical surface properties from one-shot in-field reflectance," in *European Conference on Computer Vision*, pp. 808–824, Springer, 2016.

[21] H. Zhang, J. Xue, and K. Dana, "Deep ten: Texture encoding network," in *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, July 2017.

[22] H. Zhang and K. Dana, "Multi-style generative network for real-time transfer," *arXiv preprint arXiv:1703.06953*, 2017.

[23] P. Kaur, H. Zhang, and K. Dana, "Photo-realistic facial texture transfer," *arXiv preprint arXiv:1706.04306*, 2017.

[24] J. Xue, H. Zhang, K. Dana, and K. Nishino, "Differential angular imaging for material recognition," in *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, July 2017.

[25] K. J. Dana, "Brdf/btf measurement device," *International Conference on Computer Vision*, vol. 2, pp. 460–6, July 2001.

[26] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "ImageNet: A Large-Scale Hierarchical Image Database," in *CVPR09*, 2009.

[27] S. Lombardi and K. Nishino, "Reflectance and Natural Illumination from A Single Image," in *European Conference on Computer Vision*, vol. VI, pp. 582–595, 2012.

[28] O. G. Cula and K. J. Dana, "3D texture recognition using bidirectional feature histograms," *International Journal of Computer Vision*, vol. 59, pp. 33–60, August 2004.

[29] T. Leung and J. Malik, "Representing and recognizing the visual appearance of materials using three-dimensional textons," *International Journal of Computer Vision*, vol. 43, no. 1, pp. 29–44, 2001.

[30] R. Kumar, M. Jones, and T. Marks, "Morphable reflectance fields for enhancing face recognition," in *Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference on*, pp. 2606–2613, 2010.

[31] J. Gu and C. Liu, "Discriminative illumination: Per-pixel classification of raw materials based on optimal projections of spectral brdf," in *Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference on*, pp. 797–804, IEEE, 2012.

[32] L. Sharan, C. Liu, R. Rosenholtz, and E. Adelson, "Recognizing materials using perceptually inspired features," *International Journal of Computer Vision*, vol. 103, no. 3, pp. 348–371, 2013.

[33] C. Liu, L. Sharan, E. Adelson, and R. Rosenholtz, "Exploring features in a bayesian framework for material recognition," in *Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference on*, pp. 239–246, 2010.

[34] W. Li and M. Fritz, "Recognizing materials from virtual examples," in *Computer Vision–ECCV 2012*, pp. 345–358, Springer, 2012.

[35] E. H. Adelson, "On seeing stuff: the perception of materials by humans and machines," in *Photonics West 2001-Electronic Imaging*, pp. 1–12, International Society for Optics and Photonics, 2001.

[36] S. Lombardi and K. Nishino, "Single Image Multimaterial Estimation," in *IEEE Conference on Computer Vision and Pattern Recognition*, pp. 238–245, 2012.

[37] G. Oxholm and K. Nishino, "Multiview shape and reflectance from natural illumination," *IEEE Conference on Computer Vision and Pattern Recognition*, 2014.

[38] G. Oxholm and K. Nishino, "Shape and reflectance from natural illumination," *European Conference on Computer Vision*, pp. 528–541, 2012.

[39] C. Liu and J. Gu, "Discriminative illumination: Per-pixel classification of raw materials based on optimal projections of spectral brdf," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 36, no. 1, pp. 86–98, 2014.

[40] S.-C. Foo, "A gonioreflectometer for measuring the bidirectional reflectance of material for use in illumination computation," Master's thesis, Cornell University, 1996.

[41] G. Ward, "Measuring and modeling anisotropic reflection," in *ACM SIGGRAPH 92*, pp. 265–272, 1992.

[42] M. Levoy and P. Hanrahan, "Light field rendering," in *Proceedings of the 23rd Annual Conference on Computer Graphics and Interactive Techniques*, SIGGRAPH '96, (New York, NY, USA), pp. 31–42, ACM, 1996.

[43] K. J. Dana, B. van Ginneken, S. K. Nayar, and J. J. Koenderink, "Reflectance and texture of real world surfaces," *ACM Trans. Graphics*, vol. 18, pp. 1–34, January 1999.

[44] W. Matusik, H. Pfister, M. Brand, and L. McMillan, "A Data-Driven Reflectance Model," *ACM Trans. on Graphics*, vol. 22, no. 3, pp. 759–769, 2003.

[45] W. Matusik, H. Pfister, M. Brand, and L. McMillan, "Efficient Isotropic BRDF Measurement," in *Proc. of Eurographics Symposium on Rendering*, 2003.

[46] O. Wang, P. Gunawardane, S. Scher, and J. Davis, "Material classification using brdf slices," in *IEEE Conf. on Computer Vision and Pattern Recognition*, pp. 2805–2811, 2009.

[47] M. Johnson and E. Adelson, "Retrographic sensing for the measurement of surface texture and shape," in *Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on*, pp. 1070–1077, 2009.

[48] K. Dana and J. Wang, "Device for convenient measurement of spatially varying bidirectional reflectance," *Journal of the Optical Society of America A*, vol. 21, pp. pp. 1–12, January 2004.

[49] J. Y. Han and K. Perlin, "Measuring bidirectional texture reflectance with a kaleidoscope," *ACM Trans. Graph.*, vol. 22, pp. 741–748, July 2003.

[50] M. Varma and A. Zisserman, "A statistical approach to material classification using image patch exemplars," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 31, no. 11, pp. 2032–2047, 2009.

[51] J. Wang and K. J. Dana, "Hybrid textons: Modeling surfaces with reflectance and geometry," *IEEE CVPR Conference on Computer Vision and Pattern Recognition*, pp. 372–378, 2004.

[52] J. Shotton, J. Winn, C. Rother, and A. Criminisi, "Textonboost: Joint appearance, shape and context modeling for multi-class object recognition and segmentation," in *Proceedings of the 9th European Conference on Computer Vision - Volume Part I*, ECCV'06, (Berlin, Heidelberg), pp. 1–15, Springer-Verlag, 2006.

[53] J. Shotton, J. Winn, C. Rother, and A. Criminisi, "Textonboost for image understanding: Multi-class object recognition and segmentation by jointly modeling texture, layout, and context," *Int. J. Comput. Vision*, vol. 81, pp. 2–23, Jan. 2009.

[54] L. Ladicky, J. Shi, and M. Pollefeys, "Pulling things out of perspective," in *Computer Vision and Pattern Recognition (CVPR), 2014 IEEE Conference on*, pp. 89–96, June 2014.

[55] A. Torralba, K. P. Murphy, and W. T. Freeman, "Sharing visual features for multiclass and multiview object detection," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 29, no. 5, pp. 854–869, 2007.

[56] B. Kulis and T. Darrell, "Learning to hash with binary reconstructive embeddings," in *Advances in Neural Information Processing Systems (NIPS)*, pp. 1042–1050, 2009.

[57] A. Bergamo, L. Torresani, and A. Fitzgibbon, "Picodes: Learning a compact code for novel-category recognition," *Advances in Neural Information Processing Systems 24*, pp. 2088–2096, 2011.

[58] J. Wang and S. fu Chang, "Sequential projection learning for hashing with compact codes," 2010.

[59] W. Liu, J. Wang, and S. fu Chang, "Hashing with graphs," in *IInternational Conference on Machine Learning (ICML)*, 2011.

[60] W. Liu, J. Wang, R. Ji, Y.-G. Jiang, and S.-F. Chang, "Supervised hashing with kernels," in *Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference on*, pp. 2074–2081, 2012.

[61] Y. Weiss, A. Torralba, and R. Fergus, "Spectral hashing," *Advances in Neural Information Processing Systems (NIPS)*, pp. 1753–1760, 2008.

[62] A. Torralba, R. Fergus, and Y. Weiss, "Small codes and large image databases for recognition," in *Computer Vision and Pattern Recognition, 2008. CVPR 2008. IEEE Conference on*, pp. 1–8, 2008.

[63] M. Raginsky and S. Lazebnik, "Locality-sensitive binary codes from shift-invariant kernels," *Advances in Neural Information Processing Systems (NIPS)*, pp. pp. 1509–1517, 2010.

[64] J. Wang and K. J. Dana, "A novel approach for texture shape recovery," *ICCV International Conference on Computer Vision*, pp. 1374–1380, October 2003.

[65] J. Wang and K. J. Dana, "Relief texture from specularities," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 28, no. 3, pp. 446–457, 2006.

[66] J. C. van Gemert, J.-M. Geusebroek, C. J. Veenman, and A. W. M. Smeulders, "Kernel codebooks for scene categorization," in *ECCV 2008, PART III. LNCS*, pp. 696–709, Springer, 2008.

[67] F. X. Yu, S. Kumar, Y. Gong, and S.-F. Chang, "Circulant binary embedding," in *International Conference on Machine Learning*, 2014.

[68] Y. Gong, S. Kumar, H. A. Rowley, and S. Lazebnik, "Learning binary codes for high-dimensional data using bilinear projections," *IEEE Conference on Computer Vision and Pattern Recognition*, pp. 484–491, 2013.

[69] Y. Gong, S. Lazebnik, A. Gordo, and F. Perronnin, "Iterative quantization: A procrustean approach to learning binary codes for large-scale image retrieval," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 35, no. 12, pp. 2916–2929, 2013.

[70] Y. Gong, S. Kumar, V. Verma, and S. Lazebnik, "Angular quantization-based binary codes for fast similarity search," in *Advances in Neural Information Processing Systems*, pp. 1196–1204, 2012.

[71] Y. Weiss, A. Torralba, and R. Fergus, "Spectral hashing," in *Advances in Neural Information Processing Systems 21* (D. Koller, D. Schuurmans, Y. Bengio, and L. Bottou, eds.), pp. 1753–1760, Curran Associates, Inc., 2009.

[72] M. S. Charikar, "Similarity estimation techniques from rounding algorithms," in *Proceedings of the thiry-fourth annual ACM symposium on Theory of computing*, pp. 380–388, ACM, 2002.

[73] M. Raginsky and S. Lazebnik, "Locality-sensitive binary codes from shift-invariant kernels," in *Advances in neural information processing systems*, pp. 1509–1517, 2009.

[74] C. Liu, G. Yang, and J. Gu, "Learning discriminative illumination and filters for raw material classification with optimal projections of bidirectional texture functions," in *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2013.

[75] S. Su, F. Heide, R. Swanson, J. Klein, C. Callenberg, M. Hullin, and W. Heidrich, "Material classification using raw time-of-flight measurements," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 3503–3511, 2016.

[76] S. Bell, P. Upchurch, N. Snavely, and K. Bala, "Material recognition in the wild with the materials in context database," *Computer Vision and Pattern Recognition (CVPR)*, 2015.

[77] M. Cimpoi, S. Maji, I. Kokkinos, S. Mohamed, and A. Vedaldi, "Describing textures in the wild," in *Computer Vision and Pattern Recognition (CVPR), 2014 IEEE Conference on*, pp. 3606–3613, IEEE, 2014.

[78] L. Van der Maaten and G. Hinton, "Visualizing data using t-sne," *Journal of Machine Learning Research*, vol. 9, no. 2579-2605, p. 85, 2008.

[79] G. Erdogan, L. Alexander, and R. Rajamani, "A novel wireless piezoelectric tire sensor for the estimation of slip angle," *Measurement Science and Technology*, vol. 21, no. 1, p. 015201, 2010.

[80] F. Gustafsson, "Slip-based tire-road friction estimation," *Automatica*, vol. 33, no. 6, pp. 1087–1099, 1997.

[81] R. Matsuzaki and A. Todoroki, "Wireless strain monitoring of tires using electrical capacitance changes with an oscillating circuit," *Sensors and Actuators A: Physical*, vol. 119, no. 2, pp. 323–331, 2005.

[82] S. Ramkumar, D. Wood, K. Fox, and S. Harlock, "Developing a polymeric human finger sensor to study the frictional properties of textiles part i: artificial finger development," *Textile research journal*, vol. 73, no. 6, pp. 469–473, 2003.

[83] E. Bertaux, M. Lewandowski, and S. Derler, "Relationship between friction and tactile properties for woven and knitted fabrics," *Textile Research Journal*, vol. 77, no. 6, pp. 387–396, 2007.

[84] C. Nyahumwa and J. Jeswiet, "A friction sensor for sheet-metal rolling," *CIRP Annals-Manufacturing Technology*, vol. 40, no. 1, pp. 231–233, 1991.

[85] M. Varma and A. Zisserman, "Classifying images of materials: Achieving viewpoint and illumination independence," in *Computer VisionECCV 2002*, pp. 255–271, Springer, 2002.

[86] F. Perronnin, Y. Liu, J. Sánchez, and H. Poirier, "Large-scale image retrieval with compressed fisher vectors," in *Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference on*, pp. 3384–3391, IEEE, 2010.

[87] R. Salakhutdinov and G. Hinton, "Semantic hashing," *International Journal of Approximate Reasoning*, vol. 50, no. 7, pp. 969–978, 2009.

[88] A. Torralba, R. Fergus, and Y. Weiss, "Small codes and large image databases for recognition," in *Computer Vision and Pattern Recognition, 2008. CVPR 2008. IEEE Conference on*, pp. 1–8, IEEE, 2008.

[89] V. Erin Liong, J. Lu, G. Wang, P. Moulin, and J. Zhou, "Deep hashing for compact binary codes learning," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 2475–2483, 2015.

[90] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, A. C. Berg, and L. Fei-Fei, "ImageNet Large Scale Visual Recognition Challenge," *International Journal of Computer Vision (IJCV)*, pp. 1–42, April 2015.

[91] J. Donahue, Y. Jia, O. Vinyals, J. Hoffman, N. Zhang, E. Tzeng, and T. Darrell, "Decaf: A deep convolutional activation feature for generic visual recognition," in *Proceedings of The 31st International Conference on Machine Learning*, pp. 647–655, 2014.

[92] A. S. Razavian, H. Azizpour, J. Sullivan, and S. Carlsson, "Cnn features off-the-shelf: an astounding baseline for recognition," in *Computer Vision and Pattern Recognition Workshops (CVPRW), 2014 IEEE Conference on*, pp. 512–519, IEEE, 2014.

[93] B. Zhou, A. Lapedriza, J. Xiao, A. Torralba, and A. Oliva, "Learning deep features for scene recognition using places database," in *Advances in Neural Information Processing Systems*, pp. 487–495, 2014.

[94] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Advances in neural information processing systems*, pp. 1097–1105, 2012.

[95] Y. Jia, E. Shelhamer, J. Donahue, S. Karayev, J. Long, R. Girshick, S. Guadarrama, and T. Darrell, "Caffe: Convolutional architecture for fast feature embedding," in *Proceedings of the ACM International Conference on Multimedia*, pp. 675–678, ACM, 2014.

[96] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *Nature*, pp. 436–444, May 2015.

[97] R. Arriaga, S. Vempala, *et al.*, "An algorithmic theory of learning: Robust concepts and random projection," in *Foundations of Computer Science, 1999. 40th Annual Symposium on*, pp. 616–623, IEEE, 1999.

[98] Q. Shi, C. Shen, R. Hill, and A. Hengel, "Is margin preserved after random projection?," in *Proceedings of the 29th International Conference on Machine Learning (ICML-12)*, pp. 591–598, 2012.

[99] B. Caputo, E. Hayman, and P. Mallikarjuna, "Class-specific material categorisation," in *Computer Vision, 2005. ICCV 2005. Tenth IEEE International Conference on*, vol. 2, pp. 1597–1604, IEEE, 2005.

[100] X. Zhang, F. X. Yu, R. Guo, S. Kumar, S. Wang, and S.-F. Chang, "Fast orthogonal projection based on kronecker product," *International Conference on Computer Vision*, 2015.

[101] K. Chatfield, K. Simonyan, A. Vedaldi, and A. Zisserman, "Return of the devil in the details: Delving deep into convolutional nets," in *British Machine Vision Conference*, 2014.

[102] A. Vedaldi and K. Lenc, "Matconvnet – convolutional neural networks for matlab," 2015.

[103] A. Vedaldi and B. Fulkerson, "VLFeat: An open and portable library of computer vision algorithms." http://www.vlfeat.org/, 2008.

[104] V. Sydorov, M. Sakurada, and C. H. Lampert, "Deep fisher kernels-end to end learning of the fisher kernel gmm parameters," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1402–1409, 2014.

[105] T.-Y. Lin, A. RoyChowdhury, and S. Maji, "Bilinear cnn models for fine-grained visual recognition," in *Proceedings of the IEEE International Conference on Computer Vision*, pp. 1449–1457, 2015.

[106] K. He, X. Zhang, S. Ren, and J. Sun, "Spatial pyramid pooling in deep convolutional networks for visual recognition," in *European Conference on Computer Vision*, pp. 346–361, Springer, 2014.

[107] R. Arandjelović, P. Gronat, A. Torii, T. Pajdla, and J. Sivic, "Netvlad: Cnn architecture for weakly supervised place recognition," *arXiv preprint arXiv:1511.07247*, 2015.

[108] S. Lloyd, "Least squares quantization in pcm," *IEEE transactions on information theory*, vol. 28, no. 2, pp. 129–137, 1982.

[109] B. S. Everitt, *Finite mixture distributions*. Wiley Online Library, 1981.

[110] J. Shotton, J. Winn, C. Rother, and A. Criminisi, "Textonboost: Joint appearance, shape and context modeling for multi-class object recognition and segmentation," in *European conference on computer vision*, pp. 1–15, Springer, 2006.

[111] L. Liu, L. Wang, and X. Liu, "In defense of soft-assignment coding," in *2011 International Conference on Computer Vision*, pp. 2486–2493, IEEE, 2011.

[112] A. Krizhevsky and G. Hinton, "Learning multiple layers of features from tiny images," *University of Toronto, Technical Report*, 2009.

[113] A. Coates, H. Lee, and A. Y. Ng, "An analysis of single-layer networks in unsupervised feature learning," *Ann Arbor*, vol. 1001, no. 48109, p. 2, 2010.

[114] J. Xiao, J. Hays, K. A. Ehinger, A. Oliva, and A. Torralba, "Sun database: Large-scale scene recognition from abbey to zoo," in *Computer vision and pattern recognition (CVPR), 2010 IEEE conference on*, pp. 3485–3492, IEEE, 2010.

[115] F. Yu, A. Seff, Y. Zhang, S. Song, T. Funkhouser, and J. Xiao, "Lsun: Construction of a large-scale image dataset using deep learning with humans in the loop," *arXiv preprint arXiv:1506.03365*, 2015.

[116] M. Everingham, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman, "The PASCAL Visual Object Classes Challenge 2007 (VOC2007) Results." http://www.pascal-network.org/challenges/VOC/voc2007/workshop/index.html.

[117] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick, "Microsoft coco: Common objects in context," in *European Conference on Computer Vision*, pp. 740–755, Springer, 2014.

[118] S. Bell, P. Upchurch, N. Snavely, and K. Bala, "Material recognition in the wild with the materials in context database," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 3479–3487, 2015.

[119] S. Ren, K. He, R. Girshick, and J. Sun, "Faster r-cnn: Towards real-time object detection with region proposal networks," in *Advances in neural information processing systems*, pp. 91–99, 2015.

[120] K. Simonyan and A. Zisserman, "Two-stream convolutional networks for action recognition in videos," in *Advances in Neural Information Processing Systems*, pp. 568–576, 2014.

[121] R. Collobert and J. Weston, "A unified architecture for natural language processing: Deep neural networks with multitask learning," in *Proceedings of the 25th international conference on Machine learning*, pp. 160–167, ACM, 2008.

[122] M. Cimpoi, S. Maji, I. Kokkinos, and A. Vedaldi, "Deep filter banks for texture recognition, description, and segmentation," *International Journal of Computer Vision*, vol. 118, no. 1, pp. 65–94, 2016.

[123] T.-C. Wang, J.-Y. Zhu, E. Hiroaki, M. Chandraker, A. Efros, and R. Ramamoorthi, "A 4D light-field dataset and CNN architectures for material recognition," in *Proceedings of European Conference on Computer Vision (ECCV)*, 2016.

[124] A. Quattoni and A. Torralba, "Recognizing indoor scenes," in *Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on*, pp. 413–420, IEEE, 2009.

[125] L. Fei-Fei, R. Fergus, and P. Perona, "Learning generative visual models from few training examples: An incremental bayesian approach tested on 101 object categories," *Computer Vision and Image Understanding*, vol. 106, no. 1, pp. 59–70, 2007.

[126] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," *arXiv preprint arXiv:1409.1556*, 2014.

[127] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," *arXiv preprint arXiv:1512.03385*, 2015.

[128] J. Zhao, M. Mathieu, R. Goroshin, and Y. Lecun, "Stacked what-where auto-encoders," *arXiv preprint arXiv:1506.02351*, 2015.

[129] K. He, X. Zhang, S. Ren, and J. Sun, "Identity mappings in deep residual networks," *arXiv preprint arXiv:1603.05027*, 2016.

[130] A. Dosovitskiy, J. T. Springenberg, M. Riedmiller, and T. Brox, "Discriminative unsupervised feature learning with convolutional neural networks," in *Advances in Neural Information Processing Systems*, pp. 766–774, 2014.

[131] R. Collobert, K. Kavukcuoglu, and C. Farabet, "Torch7: A matlab-like environment for machine learning," in *BigLearn, NIPS Workshop*, no. EPFL-CONF-192376, 2011.

[132] C. Li and M. Wand, "Combining markov random fields and convolutional neural networks for image synthesis," in *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2016.

[133] A. A. Efros and T. K. Leung, "Texture synthesis by non-parametric sampling," in *Computer Vision, 1999. The Proceedings of the Seventh IEEE International Conference on*, vol. 2, pp. 1033–1038, IEEE, 1999.

[134] A. A. Efros and W. T. Freeman, "Image quilting for texture synthesis and transfer," in *Proceedings of the 28th annual conference on Computer graphics and interactive techniques*, pp. 341–346, ACM, 2001.

[135] L.-Y. Wei and M. Levoy, "Fast texture synthesis using tree-structured vector quantization," in *Proceedings of the 27th annual conference on Computer graphics and interactive techniques*, pp. 479–488, ACM Press/Addison-Wesley Publishing Co., 2000.

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

[137] J. S. De Bonet, "Multiresolution sampling procedure for analysis and synthesis of texture images," in *Proceedings of the 24th annual conference on Computer graphics and interactive techniques*, pp. 361–368, ACM Press/Addison-Wesley Publishing Co., 1997.

[138] D. J. Heeger and J. R. Bergen, "Pyramid-based texture analysis/synthesis," in *Proceedings of the 22nd annual conference on Computer graphics and interactive techniques*, pp. 229–238, ACM, 1995.

[139] J. Portilla and E. P. Simoncelli, "A parametric texture model based on joint statistics of complex wavelet coefficients," *International journal of computer vision*, vol. 40, no. 1, pp. 49–70, 2000.

[140] E. P. Simoncelli and W. T. Freeman, "The steerable pyramid: A flexible architecture for multi-scale derivative computation," in *Image Processing, 1995. Proceedings., International Conference on*, vol. 3, pp. 444–447, IEEE, 1995.

[141] P. Burt and E. Adelson, "The laplacian pyramid as a compact image code," *IEEE Transactions on communications*, vol. 31, no. 4, pp. 532–540, 1983.

[142] L. Gatys, A. S. Ecker, and M. Bethge, "Texture synthesis using convolutional neural networks," in *Advances in Neural Information Processing Systems*, pp. 262–270, 2015.

[143] L. A. Gatys, A. S. Ecker, and M. Bethge, "Image style transfer using convolutional neural networks," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 2414–2423, 2016.

[144] D. Chen, L. Yuan, J. Liao, N. Yu, and G. Hua, "Stylebank: An explicit representation for neural image style transfer," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017.

[145] C. Li and M. Wand, "Precomputed real-time texture synthesis with markovian generative adversarial networks," in *Proceedings of the European Conference on Computer Vision (ECCV)*, 2016.

[146] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 770–778, 2016.

[147] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, *et al.*, "Imagenet large scale visual recognition challenge," *International Journal of Computer Vision*, vol. 115, no. 3, pp. 211–252, 2015.

[148] J. Long, E. Shelhamer, and T. Darrell, "Fully convolutional networks for semantic segmentation," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 3431–3440, 2015.

[149] A. Radford, L. Metz, and S. Chintala, "Unsupervised representation learning with deep convolutional generative adversarial networks," *arXiv preprint arXiv:1511.06434*, 2015.

[150] D. Ulyanov, A. Vedaldi, and V. Lempitsky, "Improved texture networks: Maximizing quality and diversity in feed-forward stylization and texture synthesis," *arXiv preprint arXiv:1701.02096*, 2017.

[151] A. Mahendran and A. Vedaldi, "Understanding deep image representations by inverting them," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 5188–5196, 2015.

[152] H. Zhang, J. Yang, Y. Zhang, and T. S. Huang, "Non-local kernel regression for image and video restoration," in *European Conference on Computer Vision*, pp. 566–579, Springer, 2010.

[153] P. Isola, J.-Y. Zhu, T. Zhou, and A. A. Efros, "Image-to-image translation with conditional adversarial networks," *arXiv preprint arXiv:1611.07004*, 2016.

[154] H. Zhang, T. Xu, H. Li, S. Zhang, X. Huang, X. Wang, and D. Metaxas, "Stackgan: Text to photo-realistic image synthesis with stacked generative adversarial networks," *arXiv preprint arXiv:1612.03242*, 2016.

[155] C. Feichtenhofer, A. Pinz, and A. Zisserman, "Convolutional two-stream network fusion for video action recognition," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1933–1941, 2016.

[156] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative adversarial nets," in *Advances in neural information processing systems*, pp. 2672–2680, 2014.

[157] T. Che, Y. Li, A. P. Jacob, Y. Bengio, and W. Li, "Mode regularized generative adversarial networks," *arXiv preprint arXiv:1612.02136*, 2016.

[158] X. Huang, Y. Li, O. Poursaeed, J. Hopcroft, and S. Belongie, "Stacked generative adversarial networks," *arXiv*, 2016.

[159] T. Q. Chen and M. Schmidt, "Fast patch-based style transfer of arbitrary style," *arXiv preprint arXiv:1612.04337*, 2016.

[160] V. Dumoulin, J. Shlens, and M. Kudlur, "A learned representation for artistic style," 2016.

[161] J. Johnson, "neural-style." https://github.com/jcjohnson/neural-style, 2015.

[162] D. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.

[163] R. Collobert, K. Kavukcuoglu, and C. Farabet, "Torch7: A matlab-like environment for machine learning," in *BigLearn, NIPS Workshop*, no. EPFL-CONF-192376, 2011.

[164] M. Cimpoi, S. Maji, I. Kokkinos, S. Mohamed, , and A. Vedaldi, "Describing textures in the wild," in *Proceedings of the IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2014.

[165] L. A. Gatys, A. S. Ecker, and M. Bethge, "Image style transfer using convolutional neural networks," in *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2016.

[166] P. Sinha, B. Balas, Y. Ostrovsky, and R. Russell, "Face recognition by humans: Nineteen results all computer vision researchers should know about," *Proceedings of the IEEE*, vol. 94, no. 11, pp. 1948–1962, 2006.

[167] E. McKone, N. Kanwisher, and B. C. Duchaine, "Can generic expertise explain special processing for faces?," *Trends in cognitive sciences*, vol. 11, no. 1, pp. 8–15, 2007.

[168] L. A. Gatys, A. S. Ecker, and M. Bethge, "A neural algorithm of artistic style," *arXiv preprint arXiv:1508.06576*, 2015.

[169] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," *arXiv preprint arXiv:1409.1556*, 2014.

[170] J. Johnson, A. Alahi, and L. Fei-Fei, "Perceptual losses for real-time style transfer and super-resolution," in *European Conference on Computer Vision*, 2016.

[171] C. Li and M. Wand, "Precomputed real-time texture synthesis with markovian generative adversarial networks," in *European Conference on Computer Vision*, pp. 702–716, Springer, 2016.

[172] D. Ulyanov, A. Vedaldi, and V. S. Lempitsky, "Instance normalization: The missing ingredient for fast stylization," *CoRR*, vol. abs/1607.08022, 2016.

[173] V. Dumoulin, J. Shlens, and M. Kudlur, "A learned representation for artistic style," *CoRR*, vol. abs/1610.07629, 2016.

[174] F. Luan, S. Paris, E. Shechtman, and K. Bala, "Deep photo style transfer," *arXiv preprint arXiv:1703.07511*, 2017.

[175] S. C. Zhu, Y. N. Wu, and D. Mumford, "Filters, random field and maximum entropy: Towards a unified theory for texture modeling," *International Journal of Computer Vision*, vol. 27, pp. 1–20, March/April 1998.

[176] N. Srivastava, G. E. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: a simple way to prevent neural networks from overfitting.," *Journal of Machine Learning Research*, vol. 15, no. 1, pp. 1929–1958, 2014.

[177] J. M. Saragih, S. Lucey, and J. F. Cohn, "Face alignment through subspace constrained mean-shifts," in *Computer Vision, 2009 IEEE 12th International Conference on*, pp. 1034–1041, Ieee, 2009.

[178] O. M. Parkhi, A. Vedaldi, and A. Zisserman, "Deep face recognition.," in *BMVC*, vol. 1, p. 6, 2015.

[179] T. Beier and S. Neely, "Feature-based image metamorphosis," in *ACM SIGGRAPH Computer Graphics*, vol. 26, pp. 35–42, ACM, 1992.

[180] C. Liu, J. Yuen, and A. Torralba, "Sift flow: Dense correspondence across scenes and its applications," *IEEE transactions on pattern analysis and machine intelligence*, vol. 33, no. 5, pp. 978–994, 2011.

[181] J. Johnson, "neural-style." https://github.com/jcjohnson/neural-style, 2015.

[182] Y. Shih, S. Paris, C. Barnes, W. T. Freeman, and F. Durand, "Style transfer for headshot portraits," 2014.