Image Based PTM Synthesis For Realistic Rendering of Low Resolution 3D models

Thesis submitted in partial fulfillment
of the requirements for the degree of

Master of Science (by Research)
in
Computer Science

by

N.Pradeep Rajiv
200402028
pradeepr@research.iiit.ac.in

CVIT
International Institute of Information Technology
Hyderabad - 500 032, INDIA
June 2011

International Institute of Information Technology
Hyderabad, India

CERTIFICATE

It is certified that the work contained in this thesis, titled “Image based PTM synthesis for Realistic Rendering of Low Resolution 3D models ” by N.Pradeep Rajiv, has been carried out under my supervision and is not submitted elsewhere for a degree.

Date

Adviser: Anoop M Namboodiri
Assistant Professor
IIIT,Hyderabad

To CVIT, the place that taught me life and helped realize my dream of becoming a scientist.

To My Family&Friends, who were in it with me always and lent support,without whome this would have been incomplete.

To My Advisor, who taught me the essence of research and inculcated the values of scientific thought in me.

Acknowledgments

Three years it's been since I started working on this interesting idea born out of my professor's curiosity and this thesis is the culmination of all the efforts I and my professor have put in. All this time, so many people have lent support to me going out of their ways and what better place to acknowledge them than here. First and foremost, I would like to thank my advisor Anoop M Namboodiri who taught me the essence of research, inculcating in me the scientific values of research and instilled in me the confidence to tackle the toughest problems of real world. The freedom of thought he gave me is one thing that stands out the most.

The research department I belong to, Centre For Visual Information Technology (**CVIT**), made me what I am today and all the people part of it need a special mention. Prof P.J.Narayanan, who advocated research as an essence of development, heading **CVIT**, put in place an efficient system which catered to various needs of the students, leaving no stone unturned to put the best research environment in place. I am happy to be a product of research lab and proud to be its prodigy. I would also like to mention the efforts put in by Prof C.V.Jawahar, **CVIT** who laid a strong research foundation by teaching us some of the important stream courses in an effective manner.

I also take this opportunity to acknowledge the efforts of my family which was in it with me all the time, instilling confidence and encouraging me. The friends in my department and institute, who all guided me through tough times and made my stay here happier and excite need a special mention. Last but not the least, CVIT, which supported me all through my research financially, academically and in every way expected, paved the pay for this work and I would like to acknowledge its assistance.

Abstract

Capturing the shape and texture of large structures such as monuments and statues at very high resolution is extremely expensive, both in terms of time as well as storage space. In many cases the inner details are generated by surface properties of the material, and the appearance is statistically uniform. In this paper, we present an approach to add surface details to a coarse **3D** model of an object based on two additional information: a set of images of the object and a high resolution model of the material that the object is made of. The material model that we employ is the Polynomial Texture Map (**PTM**), which captures the appearance of a surface under various illumination conditions. We use the observed images of the object as constraints to synthesize texture samples for each triangle of the object under any given illumination.

The primary challenge is to synthesize a polynomial model of the texture, where the constraints arise in the image domain. We use the knowledge of object illumination to map the texture models into image space and compute the opti- mal patch. The texture transfer then happens as a complete **3D** texturemodel. We also consider the problems of pose, scale, reflectance and smoothness of surface while carrying out the texture transfer. We synthesize the texture of an object at a per-triangle basis while carrying out operations such as normalization and blending to take care of discontinuities at the edges.

Contents

Chapter Page

1 Introduction . . . 1
1.1 Problem . . . 3
1.2 Motivation . . . 3
1.3 Challenges . . . 3
1.4 Our Approach . . . 5

2 Texture: Its Analysis and Modeling . . . 7
2.1 What is texture? . . . 7
2.1.1 Texture Mapping . . . 7
2.1.2 Texture Analysis . . . 8
2.1.3 Texture Modeling . . . 9
2.2 3D Textures . . . 11

3 Texture Synthesis . . . 14
3.1 What is Texture Synthesis ? . . . 14
3.2 Overview of Texture Models For Synthesis . . . 15
3.3 2D Texture Synthesis algorithms . . . 17
3.4 Surface Texture Synthesis . . . 25
3.4.1 Short-comings of Simple Color Textures And Need For Better Models . . . 29
3.5 Synthesis algorithms for Reflectance Textures . . . 29

4 Realistic Rendering of Real World Objects . . . 31
4.1 Image Based Modeling of 3D Objects . . . 32
4.2 Reflectance Properties of Natural Materials . . . 33
4.3 Synthesis of Reflectance Texture Maps . . . 34
4.3.1 Relevance to Realistic Rendering . . . 34
4.4 Our Work: Image Based PTM synthesis . . . 35
4.4.1 Constrained PTM synthesis . . . 35
4.4.2 Image constrained PTM synthesis for Planar Rectangular Surfaces . . . 36
4.4.2.1 Image Based Constraints . . . 37
4.4.2.2 Overlapping Constraints . . . 37
4.4.2.3 . . . 38
4.5 Conclusion . . . 39

5 Image Constrained PTM Synthesis for Real world Objects 42
5.0.0.4 Step1 43
5.0.0.5 Step2 43
5.0.0.6 Step3 44
5.0.0.7 Step4 44
5.0.1 Rendering of the PTM Model 45
5.0.2 Experimental Results 45

6 Conclusions 47

Bibliography 50

List of Figures

Figure Page

1.1 Reality vs Rendering 4
1.2 An artificially-colored model of the statue containing 8 million polygons. Notice the artifacts across the surface. 5

2.1 Natural Textures 7
2.2 Spectrum of Natural Textures 9
2.3 PTM vs Conventional Texture Map: The upper portions of the images shows the visual appearance of a **PTM** while the bottom half shows the conventional Texture map. Note how the former appears realistic while the later suffers from unrealistic lighting and shadows. 13

3.1 Synthesis of a Texture 15
3.2 Synthesis Results of paget and Long-staff's Algorithm 18
3.3 Results of De Bonet's Algorithm 19
3.4 Results of Efros and Leung's Algorithm 21
3.5 Results of Wei and Levoy's TSVQ Algorithm 22
3.6 Results of Ashikhmin's Algorithm 23
3.7 Results of Liang et al.'s Patch based sampling Algorithm 25
3.8 Results of Efros and Freeman's Image Quilting Algorithm 26
3.9 Results of Lapped Textures 27
3.10 Results of Wei and Levoy's Surface Synthesis Algorithm 28
3.11 Sponge, Popcorn Kernels and Peas textured Teapots illuminated in different light conditions 30

4.1 Reality vs Rendering 31
4.2 Images of a rough plaster surface obtained under varying light conditions. Note the change in surface appearance in each of (a), (b), (c) and (d) 33
4.3 Blocks from Input sample are checked for Image Based and Overlapping Constraints and the best ones pasted into the Output 36
4.4 Demonstration of Our 2D Synthesis Algorithm for Natural Material Surfaces:Images 1(a), 2(a), 3(a) show high resolution texture patches of rough plaster surface,medium to smooth plaster surface and a directional plaster surface.Images 1(b), 2(b) and 3(b) are low resolution images of rectangular objects made of the above mentioned materials and 1(c), 2(c) and 3(c) represent the views generated from their corresponding **PTM**s .Images 1(d), 2(d) and 3(d) are the higher resolution views of the models .The final row of images 1(e), 2(e) and 3(e) are the ones obtained by zooming in 1(b), 2(b), 3(b) respectively. 40

4.5 Constrained Synthesis vs Unconstrained Synthesis 41

5.1 Blending of Neighboring Triangles Red region of T1 blends with blue region of T2 and vice versa 44

5.2 Synthesis Results for 3D object:Images (a), (b) and (c) show arbitrary views of rugged planar surface.Image(d) shows a rugged sphere and (e),(f) show two views of the texture model generated model generated.Likewise images (g),(e) and (f) show the same for a rugged Cylinder. 46

List of Tables

Table Page

Chapter 1

Introduction

Computer Graphics is a branch of Computer Science that deals with the representation and synthesis of visual content. It focuses on the mathematical and computational foundations of image generation and processing. Although the term usually refers to **3D** computer graphics, it also encompasses two-dimensional graphics and image processing.

3D computer graphics are graphics that use a three-dimensional representation of geometric data that is stored in the computer for the purposes of performing calculations and rendering **2D** images. Such images may be stored for viewing later or displayed in real-time. A **3D** graphics system usually consists of a rendering engine that takes as input a model file, that contains a mathematical representation of any three-dimensional object, performs a series of calculations and operations on it and generates as output the **2D** views of it.

The input **3D** model usually consists of mathematical description of the surface geometry of the object and the surface color/texture information. The geometrical description consists of location of points on the surface and the color information usually consists of simple color value or the position of a corresponding pixel on a texture image. The two most common sources of **3D** models are those created on the computer by an artist or engineer using some kind of **3D** modeling tool, and those scanned into a computer from real-world objects. They may be created using multiple approaches: use of NURBS curves to generate accurate and smooth surface patches, polygonal mesh modeling (manipulation of faceted geometry), or polygonal mesh subdivision. Polygonal mesh models and Dense point models are widely used representation to model real world objects because of their ability to model complex geometry and simplicity to work with.

The Core engine performs the task of automatically converting **3D** wire frame models into **2D** images with photo realistic effects on a computer. It consumes data about polygons with vertices, edges and faces that constitute the whole model and geometry in the complete **3D** scene is lit according to the defined locations of light sources and reflectance and other surface properties and **2D** view of the model generated.

Rendering is the final process of creating the actual **2D** image or animation from the prepared scene. This can be compared to taking a photo or filming the scene after the setup is finished in real life. Several different, and often specialized, rendering methods have been developed. These range from the distinctly non-realistic wire-frame rendering through polygon-based rendering, to more advanced techniques such as: scan-line rendering, ray tracing, or radiosity. Rendering may take from fractions of a second to days for a single image/frame. In general, different methods are better suited for either photo-realistic rendering, or real-time rendering.

Rendering for interactive media, such as games and simulations, is calculated and displayed in real time, at rates of approximately 20 to 120 frames per second. In real-time rendering, the goal is to show as much information as possible as the eye can process in a fraction of a second. The primary goal is to achieve an as high as possible degree of photorealism at an acceptable minimum rendering speed (usually 24 frames per second).

3D rendering is the computer graphics process of automatically converting three dimensional wire frame models into **2D** images with photo realistic effects on a computer. Visual-realism of such models is usually enhanced by careful modeling of the surface shape and color information. Methods have been proposed that emulate the physical generative process of some of the surfaces seen in real world and assign similar color information to the vertices of a **3D** mesh model. The resulting model looks both visually realistic and aesthetic. However, this is a complex, time taking and often a laborious task, hindering the scope for real-time rendering and viewing of such models. Moreover, mimicking the generative process is not always possible and limited to a small class of surfaces.

This is where, texture-mapping, a powerful tool for adding the surface detail to an object by wrapping or projecting the color information from a digital image, comes handy. Computer rendering of objects with surface texture are more interesting and realistic than those without texture. Images are the widely used source of textures as they ably capture visual and structural information of the real world. They are also capable of capturing a high level of object properties. This led to the advent of Image based modeling (**IBMR**) techniques, that rely on a set of two-dimensional images of an object to generate its three-dimensional model and then render some novel views of the same. View synthesis methods, also belonging to the class of **IBMR**, use multiple two-dimensional images of an object in order to generate directly novel views, skipping the manual modeling stage. However, when it comes to modeling of real-world objects, the leverage in visualization offered by the **3D** modeling of the object is critical, UN-parallel and results in an enhanced view. We in this dissertation, underline the significance of **IBMR** methods in generating realistic models of real-world object and their importance in digital heritage.

1.1 Problem

Given the shape model (polygonal mesh model), a sparse set of views of a real-world object and auxillary information describing the material properties, texture the model augmenting the information from the images as well as the material properties, to generate a realistic model of the object which not only looks visually pleasing and similar to its real-world counter part, but also dynamically changes its visual appearance with the changing light conditions.

1.2 Motivation

Realistic rendering of real world objects is an important area of computer graphics. It is used in a variety of applications, the most prominent of them being movies, games and archival of historical artifacts. An efficient mechanism that can solve this problem holds the key for Digital Heritage project, whose objective is to create realistic models of the statues, artifacts etc. there by conserving the heritage.

1.3 Challenges

Real world objects are characterized by their shape/geometry as well as the surface properties. To faithfully model a real world object, both the shape and surface properties of the object have to be correctly captured and then rendered through the graphics pipeline.

However, when dealing with objects of large scale, it is difficult to capture these details at a fine level as the capture devices have limited resolution while working at large scales. Capturing the shape and texture of large structures such as monuments and statues at very high resolution is extremely expensive, both in terms of time as well as storage space. One could handle this problem using a very high resolution shape models of the parts of the object and fitting them together.

Recent improvements in laser range-finder technology, together with algorithms developed for combining multiple range and color images, allow one to reliably and accurately digitize the external shape and surface characteristics of many physical objects. Examples include machine parts, cultural artifacts, and design models for the manufacturing, movie making, and video game industries.As an application of this technology, a team of 30 faculty, staff, and students from Stanford University and the University of Washington spent the 1998-99 academic year in Italy scanning the sculptures and architecture of Michelangelo. The prominent among them being David Michelangelo's statue. They initially constructed a model of David containing 4 million polygons at a resolution of 1.0 mm. Although this model looks fairly good, their goal of building was to build a full-resolution (0.29 mm) model that would very much behave like the original. The final refined model contains about 2 billion triangles and 7000 color images. Clean up, align, merge, and processing this much geometric and color data is a huge and complicated task. Processing data of this scale and rendering in real time is a gigantic task. Such efforts

Figure 1.1 On the left is a photograph of Michelangelo's David. On the right is a computer rendering made from a geometric model. Notice the absence of shadows in the rendered scene

surpassing the limitations of digital acquisition and rendering are not possible to replicate in case of many real world objects. Moreover, such methods are highly time taking and gigantic cost incurring in nature.

The above mentioned approach unearths 3 different problems: i) A shape model that can capture the surface details would be extremely large, ii) Assembling a single model from that of a large number of parts is often labor intensive. iii) The shape and texture by itself is often unable to capture some of the surface properties of the object such as sub-surface scattering or translucency.

While the problems (i) and (ii) arise only in case of high-resolution models of large objects, (iii) arises because simple color textures do no model the surface material properties. Modeling the interaction of surface material with the light conditions that results in dynamic change in visual appearance is another important dimension in achieving visual-realism. The images that are used for texture mapping inherently contain a set of light conditions in which they were taken. As a result, these are baked into the texture model that is obtained by stitching these images on the mesh. Hence, the resultant model appears good in some light conditions that match those of the image acquisition and contrastingly poor when viewed under different conditions. Moreover, the appearance of the model resulting from such simple color models do not account for visual phenomena such as specularities, sub-surface scattering and shadows etc and exhibit stale lighting conditions.

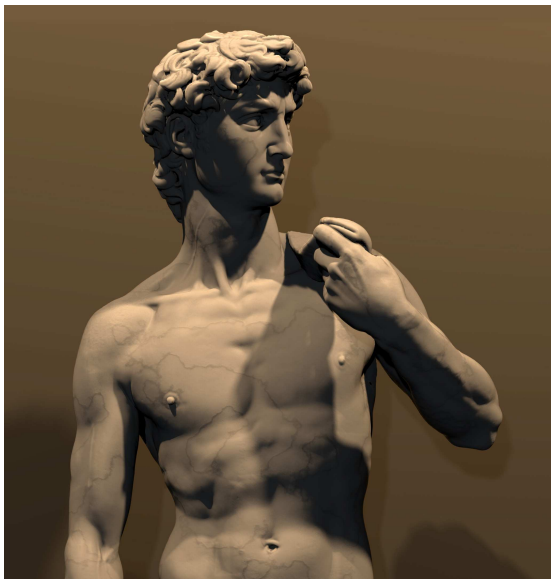


Figure 1.2 An artificially-colored model of the statue containing 8 million polygons. Notice the artifacts across the surface.

1.4 Our Approach

Existing **IBMR** techniques capture the shape information in mesh models and the finer surface details are relegated to image textures. We seek mainly two significant changes to the existing **IBMR** techniques. First and foremost, we do away with the requirement of a high resolution polygonal model (with large number of triangles) and instead seek a coarse resolution shape model. We do this mainly to avoid the physical and technical hurdles posed by the sheer task of acquiring a high resolution geometric model of a real world object. Existing techniques seek as input, high resolution geometric models with large number of polygons. This they require so that the final model is refined and exudes finer details of the surface. However, generation of high-resolution mesh model requires detailed acquisition of every small portion of the object. The limitations and challenges of geometric alignment and clean up are aplenty, thus making it a complicated task and such efforts are not possible to replicate in a generic real world scenario. On the other hand, low resolution models are easier to synthesize using images taken from long distance. This strategy, we believe, overcomes the limitation of existing **IBMR** techniques, by avoiding the difficulties faced in fine shape modeling of real world objects especially large structures and statues

Secondly, instead of stitching simple color information from the images onto the mesh model, we adopt a texture transfer mechanism to synthesize a material model that incorporates both color and material properties. A sample reflectance model of the surface material is synthesized in the laboratory conditions and sampled all across the mesh model. Simple color textures do not model the interactions of

the surface material with the light conditions and the dynamic change in visual appearance that results from it. On the contrary, our texture model infuses in the material reflectance properties as well and phenomena such as shadows, specularities, sub-surface scattering can be modelled. This enhances the visual realism of the model so rendered.

Another important observation is that finer details of such large objects are usually generated by surface material appearance and it is statistically uniform over the surface. Capturing high resolution images is often easy due to the availability of low cost and high resolution digital cameras. Images of a small surface sample facilitate generation of high resolution texture maps of the material texture and can be used to synthesize texture on the polygonal mesh models of the object using example based texture synthesis algorithms.

Guided by the above quoted observations, we present an approach to add surface details to a coarse **3D** model of an object based on two additional information: a set of images of the object and a high resolution model of the material that the object is made of. The material model that we employ is the Polynomial Texture Map (PTM), which captures the appearance of a surface under various illumination conditions. We use the observed images of the object as constraints to synthesize texture samples for each triangle of the object under any given illumination.

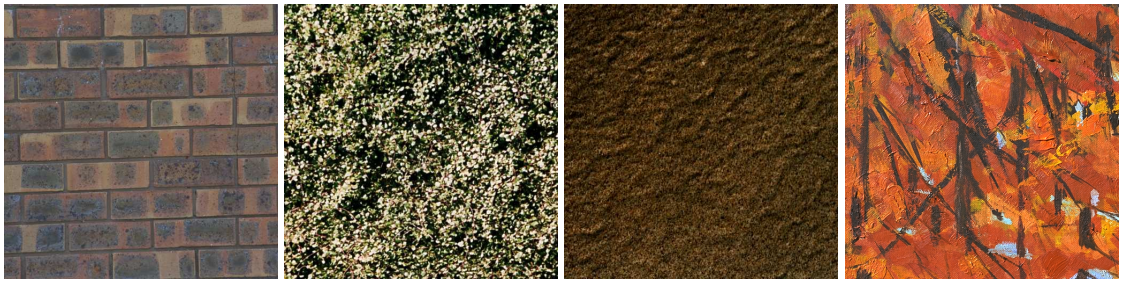
The primary challenge is to synthesize a polynomial model of the texture, where the constraints arise in the image domain. We use the knowledge of object illumination to map the texture models into image space and compute the optimal texture patch from the sample. The texture transfer then happens as a complete **3D** texture model. We also consider the problems of pose, scale, reflectance and smoothness of surface while carrying out the texture transfer. We synthesize the texture of an object at a per-triangle basis while carrying out operations such as normalization and blending to take care of discontinuities at the edges. This essentially establishes a set-up that facilitates realistic rendering of large-scale models by utilizing a coarse geometric model of the object augmented with surface details that are generated by employing example-based texture synthesis techniques. In this work, we explore the possibility of using the appearance captured in the images as well as prior knowledge of surface properties to add realistic details to a coarse **3D** mesh model of the object.

Chapter 2

Texture: Its Analysis and Modeling

2.1 What is texture?

Texture is a visual experience that describes properties of wide variety of object surfaces such as grass, animal fur, skin, water, sand, wood etc. It refers to the visual-characteristics of an image segment which human visual perception identifies as belonging to a particular class like hair, grass and sponge etc.It is an important cue in human visual perception.Texture images are spatially homogeneous and consist of repeated patterns, often subject to some randomization in their location, size, color and orientation.

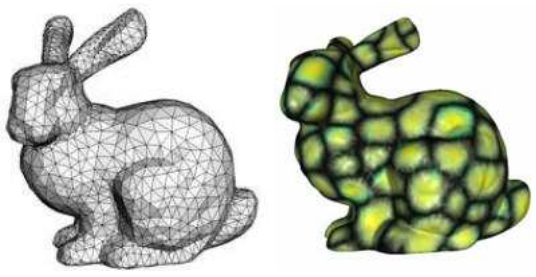


(a) Brick (b) Leaves (c) sand (d) Painting

Figure 2.1 Natural Textures

2.1.1 Texture Mapping

In computer graphics, texture usually refers to a digital image that is pasted/applied on top of a polygon or geometric object so as to obtain a realistic rendering of it.Texture mapping is a technique for adding the appearance of surface detail by wrapping or projecting a digitized texture im-



(a) Bunny Mesh-model (b) Textured Bunny

age on to the surface of an object. Computer rendering of objects with surface texture are more interesting and realistic than those with out texture. This can be observed from the pair of images shown beside. Figure 2.2(a) shows a geometric representation of the standard bunny consisting of triangles. Figure 2.2(b) shows a textured bunny. As can be observed, the textured model is more realistic and visually pleasing than a geometric mesh model. Hence texture mapping is an important criteria and extensively employed in computer graphics where reproducing the visual realism of the real world is the objective. Scanned-photographs and hand drawn pictures are the main sources of textures. Hand drawn pictures though look aesthetic, do no impart the realism that is desired of texture mapping. Digital images on the other hand are a rich source of visual information about the real world. Hence they are usually used to generate realistic models. The real world is abundant with naturally occurring textures like water, fire, clouds, vegetation etc.

Texture mapping though being a powerful tool, often suffers from two problems namely a)Seams and Tiling b) Mapping Distortion. Digitized images, being the main source of textures, are not always of desired size and shape required in the mapping process. If the image is not big enough to cover the entire object, then the texture is over-stretched and results in distorted visual appearance. Using multiple copies of the image on the other hand results in tiling and visible seams. Moreover, there is not always a natural mapping from the texture space to the topology/geometry of the object surface. This results in distortions as well.

These issues are mainly addressed by texture synthesis, a powerful technique to synthesize textures of arbitrary size and shape as and when desired. Methods have also been proposed to synthesize the texture directly on the object surface so as to avoid the need for an explicit mathematical mapping from texture space to that of topography of the object. These synthesis methods mainly employ texture analysis and modeling techniques to synthesize large samples of texture from a given input image.Hence analysis and modeling of a textures is an important aspect of understanding textures.

2.1.2 Texture Analysis

In image-processing and computer vision, texture can be defined in terms of interactions between pixels which are spatially distributed in an image.The aim of texture analysis is to capture these interactions and model them by fitting a mathematical frame-work. These texture models are the basis for texture classification/discrimination and synthesis algorithms that are extensively employed in computer vision and graphics. However texture analysis algorithms of computer vision have a different design criteria compared to those of computer graphics. Computer vision is concerned with learning accurate models of the texture to be used in texture classification and segmentation where as computer graphics is aimed at quick and efficient synthesis of textures for texture mapping without explicit need to model them.

Extensive work on texture analysis and discrimination has also been done as part of a study on human visual psycho physics research.This involved determining which aspects of a texture are hu-

mans most perceptible to and the measurements of texture variation that humans are most sensitive to, when discriminating textures. In accordance with this study, textures are described using five properties namely 1) coarseness 2) directionality 3) roughness 4) contrast and 5) line-likeness. It has been established that our human visual perception is sensitive to these aspects of a texture. These studies gave tremendous amounts of input to effectively model textures so as to develop algorithms for texture classification/discrimination and synthesis based on the human visual perception.

2.1.3 Texture Modeling

(a) Regular (b) Near Regular (c) Irregular (d) Near Stochastic (e) Stochastic

Figure 2.2 Spectrum of Natural Textures

Textures have been traditionally classified as a)Stochastic and b) Regular based on their structural appearance. Regular textures contain primitives at locations governed by a spatial placement rule. They have an order and contain noticeable structures which are placed at positions governed by a rule.Tiled floor, fishing net and checker-board patterns that are usually imprinted on clothes are some of the regular textures. Opposed to them are the stochastic structures which are random in distribution of pixels, do not contain any primitives(texels) of considerable size and obey no placement rule.They are governed by simple parameters like minimum and maximum intensities and average color. Most of them look like image noise at a coarser resolution. Textures of sand, water, bark which have no regular structure or pattern belong to this category. Most of the textures which are found in nature are partly stochastic and partly regular and only a few textures lie at the two extremes.

Figure 2.2 shows the spectrum of textures that are abundantly found in nature. Lie at the extreme are the regular and stochastic textures where as the intermediate placed in between. The properties to be modeled for regular textures are different from those of stochastic textures.Even the synthesis algorithms that work well on regular structures do not give results on stochastic textures and vice versa. This is due to the large and diversified set of texture features and not all of them being equally relevant to model any textures. As a result, the success of a synthesis algorithms depends on the features that are being

used. Hence a texture model should be generic and able to successfully capture the properties of all of kinds of textures.

Texture model is a mathematical process which can create or describe a texture. The main goal of texture modeling is to describe the texture by estimating a set of parameters and secondary goal is subsequently use those estimated parameters for texture classification. Julesz [16] conjectured that it suffices to extract k_{th} order statistics to discriminate two textures. He provided the mathematical definition of texture which gave rise to the term julesz ensemble. Given a set of statistics h extracted by human visual perception on a set of observed images of a texture pattern, a julesz ensemble is defined as the set of all the images that share the same statistics as the observed image. A julesz ensemble denoted by $\Omega(h)$, has an associated probability distribution $q(I; h)$ which is uniform over the images of the ensemble and has zero probability outside. The set of all texture images can be divided into a set of equivalence classes based on this calculated measure of statistics.

The primary approach to model textures has been to develop procedural models which emulate the physical generative process of the textures. Textures such as that of animal fur/skin, sea-shells etc have been successfully modeled using such methods. These are mainly reaction-diffusion based methods[30] which model the generative process of the textures. But they are limited in their applicability to a few textures and emulating the physical generative process is highly complicated and not always possible. But in computer vision coming up with a generic model that can accurately model a large class of textures each having their own set of unique features is really critical for texture classification, segmentation and the goal is to come up with this common frame work to model a wide variety of textures. Many models have been proposed to characterize the underlying properties of textures. Each model has its underlying assumptions and objectives. Each of them work well on a certain class of textures. These approaches mainly fall into two categories
a) Filter-based models b) Statistical models.

Filter theory emerged of detail study of the human visual perception of textures and is inspired by the multi-channel filtering mechanism discovered in neurophysics.This suggests that human vision analyzes images by decomposing them into a set of bands using a bank of linear filters and by performing some non-linear operations on top of them. Gabor filter, wavelet coefficients and image pyramid representations have all evolved as part of it. These methods are mainly employed for texture classification and segmentation.

Statistical methods try to estimate a concise model of a texture in the form of small set of parameters.But the dimensionality of the image space is vast and hence extremely difficult to model unless some assumptions are made.Locality and stationarity are the usual assumptions made.Locality of a texture asserts that the characteristics of a texture are specifics of its local spatial neighborhoods and stationarity make the statistics of a texture to depend only on relative spatial position.Probabilistic modeling techniques of this class have been greatly successful on stochastic textures which lack regular structure and primitives of comparable size. Regressive models, auto-regressive models, fractal models,

long-correlation random fields and Markov random fields are some of the statistical models.However most of them are capable of modeling higher order information.

Markov Random field methods [5] popularized by Besag(1973) [11] have largely been employed in image restoration, region segmentation and modeling textures due to their ability to capture the random nature of stochastic textures as well as the higher order information of regular textures.They consider the textures as samples drawn from a probability distribution and try to estimate the underlying distribution.

The property of **MRF** is that a variable X_s on a lattice $S = \{s = (i, j) : 0 \leq i, j \leq N\}$ can have its value x_s set to any value , but the probability of $X_s = x_s$ is conditional upon the values x_r at its neighbor sites $r \in G_s$. A local conditional probability density function (**LCPDF**) defined over its neighboring sites $r \in G_s$ determines how the variable X_s is set. The neighborhood system $G = \{G_s, s \in S\}$ and the **LCPDF** defined with respect to G and written

$$P(X_s = x_s | X_r = x_r, r \in G_s) \quad s \in S \tag{2.1}$$

defines the **MRF**.An image is modeled as an **MRF** by considering each pixel as a site on the lattice and its grey scale intensity the values of the site. The **LCPDF** that is determined from the image defines the underlying texture model.

Parametric estimation of the **MRF** are usually employed for texture classification/segmentation and non-parametric estimation used for texture synthesis.

The above methods model most of the natural textures but occasionally fail on complex textures with large structures. To model them, these methods are coupled with pyramid based image representation [1] and the texture is analyzed and modeled across multiple resolutions.The modeling at each level is usually done using the information at the already modeled lower levels and the original texture is at the highest resolution. Gaussian, Laplacian pyramids and steerable pyramids are the usual image representations used.

2.2 3D Textures

Traditional texture mapping is used to give the impression of geometric detail in a model using an image. For example, a photograph of a brick wall may be used as a texture map on a planar surface to avoid modeling the complex surface detail of the brick. However, if the lighting in the synthetic environment where the texture map is used is different from the lighting the texture map was captured under, the resulting rendering will appear incorrect and unrealistic. Worse yet when the texture is blended with the calculated lighting of a geometric surface then the resulting rendering will look very flat and smooth to the viewer. Simple color textures ignore the interaction of surface geometry with the light conditions and do not model the dynamic change in visual appearance that comes with it. Hence the modifications in appearance due to surface micro-structure are poorly approximated by attenuating the surface intensity.

Image-based re-lighting methods [8, 3, 22, 6] provide a solution to this problem. In this approach, multiple photographs of a surface, person or object are taken under varying lighting conditions and viewing directions, and a reflectance model characterizing the surface appearance is constructed. Using this model very realistic renderings of the original can be produced under arbitrary lighting and viewing conditions. These methods can be directly leveraged for the purposes of synthesizing light dependent textures also called reflectance textures. In addition to the simple color value, the reflectance textures also contain functional coefficients that control the luminance of a texel in accordance with the light position and view settings. As they model an additional dimension of spatial variation in surface luminance as a function of viewing and illumination conditions they are usually referred to as **3D** textures. **BTF**(Bi-Directional Texture Functions) and **UTF**(Uni-Directional Texture Functions) are the two reflectance texture functions usually used to model the surface reflectance properties of natural materials.

The reflectance properties of a textured opaque material can be exhaustively specified by its Bidirectional Texture Function (**BTF**) introduced in [6]. The **BTF** measures the ratio of radiance L exiting a surface at direction (ϕ_e, θ_e), to the incidence Ir-radiance I striking the surface in a differential solid angle from direction (ϕ_i, θ_i).

$$BTF_{r,g,b}(\phi_i, \theta_i, \phi_e, \theta_e, u, v) = \frac{dL(\phi_e, \theta_e, u, v)}{dL(\phi_i, \theta_i, u, v)} \tag{2.2}$$

Uni-Directional Texture Functions(**UTF**) are less exhaustive but tractable representation of reflectance properties and confine themselves to the modeling of visual appearance in relation to the lighting conditions. Unlike the **BTF**, they do not consider the view point in surface intensity calculation.

$$UTF_{r,g,b}(\phi_i, \theta_i, u, v) \tag{2.3}$$

Hence they cannot model view point phenomenon such as specularities. But they are easy to capture and do not require any camera calibration. They require only a movable light source and a stationary camera. **UTF** implicitly models surface normal information. Hence surface normals can be retrieved and then used to artificially introduce view point phenomena at the time of rendering.

Polynomial Texture Maps(**PTM**) [22] belonging to the class of **UTF** are a compact representation of reflectance textures. They model the surface luminance variations as a Bi-quadratic polynomial function at each pixel of the texture.

In order to synthesize a **PTM**, a set of images $\{I_k\}$ of the object surface are obtained under different light conditions $\{(lu_k, lv_k)\}$. These images amply capture the variations in the visual appearance of the surface and are used to build the reflectance model of the surface. The behavior at each texel is modelled independently with a reflectance function that encodes its behavior with respect to the changing light conditions. Hence the **PTM** is parameterized on spatial location (u, v) and lighting position (ϕ, θ), with the number of degrees of freedom being **4**.

Figure 2.3 PTM vs Conventional Texture Map: The upper portions of the images shows the visual appearance of a **PTM** while the bottom half shows the conventional Texture map. Note how the former appears realistic while the later suffers from unrealistic lighting and shadows.

The chromaticity of a pixel fairly remains constant under varying light conditions and it is only the luminance that varies. Hence, only the luminance $L(u,v)$ is modelled using the reflectance model. The variations in luminance L at each pixel (u,v) is approximated using a Bi-quadratic polynomial given by

$$\begin{aligned} L(u,v;l_u,l_v) &= a_o(u,v)l_u^2 + a_1(u,v)l_v^2 + a_2(u,v)l_u l_v + \\ & \quad a_3(u,v)l_u + a_4(u,v)l_v + a_5(u,v) \end{aligned} \tag{2.4}$$

where L is the luminance at pixel (u,v) and (l_u,l_v) the unit vector corresponding to the projection of light on the texture co-ordinate system. The luminance $L(u,v)$ so obtained is modulated with the normalized color value $(R_n(u,v),G_n(u,v),B_n(u,v))$ of the pixel to get the actual color.

$$\begin{aligned} R(u,v) &= L(u,v)R_n(u,v) \\ G(u,v) &= L(u,v)G_n(u,v) \\ B(u,v) &= L(u,v)B_n(u,v) \end{aligned} \tag{2.5}$$

The above representation is called **LRGB PTM** and it takes advantage of the redundancy in surface color. At each texel (u,v) of the texture map, the coefficients $(ao,...,a5)(u,v)$ of the corresponding bi-quadratic polynomial along with the normalized color value $(R_n(u,v),G_n(u,v),B_n(u,v))$ are stored. The luminance coefficients $(ao,...,a5)(u,v)$ of each texel are calculated using **SVD** method so as to fit the corresponding pixel data in the images.

Polynomial Texture Maps consisting of these surface luminance coefficients approximately model the visual behavior of the surface under different lighting conditions. Using the **PTM** model of an object, its visual appearance under arbitrary lighting conditions can be estimated and novel views generated.

Chapter 3

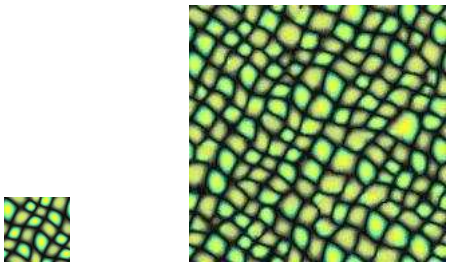
Texture Synthesis

Texture mapping is an important tool in Computer graphics that is used to add realism to the computer rendered images and objects. Textures are important for a wide variety of applications in Computer graphics and Image processing. Digitized photos are the main sources of textures due to their ability to efficiently capture the real world information. Texture mapping though is basically a simple procedure, getting the textures to be used for texture mapping is a difficult task. This is because more often that not, digitized photos are small to entirely cover a large object and this leads to visible seams and repetitions. Moreover, real world models which are **3D** in nature cannot be texture mapped directly as the surface parameterization of the object varies from the planar parameterization of the texture images. These are the two main short comings of texture mapping. Hence researchers in vision and graphics proposed texture synthesis to address these limitations of texture mapping and it has gradually evolved as an alternative for texture mapping. Using texture synthesis algorithms, textures of arbitrary size and shape can be synthesized. They can also be employed to synthesis textures directly over a **3D** surface or generate solid textures which are **3D** grid of color values. This avoids the usual problems of texture distortion that are resulted in texturing **3D** objects with planar textures. Potential applications of texture synthesis include foreground removal, image de-noising, occlusion fill-in and realistic rendering in graphics and texture compression.

3.1 What is Texture Synthesis ?

Texture synthesis refers to the procedure of algorithmically generating an image of large size from another image such that the characteristics of the synthesized image match that of the input sample. Formally, the goal of texture synthesis is to define a mathematical function F, that analyzes an input texture sample I_{in} and generates an output image I_{out} of user defined size and shape such that it appears to have been generated by the same underlying stochastic process, yet sufficiently different from the input sample in a visual manner.

Figure 3.1 shows a sample synthesis in which a small sample was taken as input and a large texture generated from it. As can be observed, the two images appear to belong to the same texture i.e they



(a) Input Image

(b) Synthesis Result

Figure 3.1 Synthesis of a Texture

have the same textural properties, yet visually they are not alike i.e the synthesis result doesn't appear to a be obtained by copying the input texture multiple times. In order for a texture synthesis algorithm to be successful, maintaining both these properties is very important.

The requirement of synthesis techniques to generate an image with similar textural properties again brings back the question of understanding what attributes and properties of a texture humans are most sensitive to. This brings the areas of texture analysis, modeling and synthesis more closer and makes them strongly connected. Hence texture synthesis along with analysis and modeling has evolved as an important area of research in the last three decades in computer science. This has also led to the advent of various texture synthesis algorithms. While the objective of all of them is essentially the same, they differ in their underlying texture models used and the assumptions made.

3.2 Overview of Texture Models For Synthesis

The initial approach to synthesize textures has been to develop methods which emulate the physical generative process of the textures they are trying to mimic. Sea shells, animal skin and fur are some of the textures successfully modeled using these procedural methods. Reaction-diffusion and cellular texturing are usually employed to simulate the biological and chemical formation of such textures. Some weathering and mineral phenomenon can also be reproduced using simulations. But such procedural methods are limited in their applicability to a set of few textures as mentioned in the earlier chapter. Moreover, these methods are governed by parameters that are complex in their functioning and have to be repeatedly tuned to get a desired result. Controlling these parameters to affect the synthesis procedure in a desirable manner is a complicated and un-intuitive procedure. Hence a need for a generic texture model that is controlled by few parameters in a logical manner and capable of describing a wide variety of textures emerged.

The first generic texture model was proposed by Julesz [16] in the 1960's in the form of a conjecture that the characteristics unique to a texture can be obtained by extracting the k^{th} order statistics. This led to the concept of Julesz ensemble. Later on two main texture modeling techniques emerged. One is

based on Filter theory adopted from research in neuro-physiology and the other based on probabilistic modeling. This classification is solely depending on the underlying texture model used and the resultant synthesis procedure.

Filter theory based synthesis techniques evolved out of research in psycho-physics to understand the perception of textures and the features of a texture that humans are most sensitive to. Research in neuro-physics has established that human eye in order to understand a visual image decomposes it into a set of linear filters and performs some non-linear operations on top of them. This has led to the modeling of textures as a set of features extracted by applying a bank of filters and filter theory based techniques for texture synthesis emerged. These techniques, given an input sample, decompose it into a set of features by applying a bank of filters and collect a set of statistics about them. This essentially gives a distribution of feature statistics in a global space. Then a random noise image of user specified size is modified in a series of steps by coercing it to have the same the set of feature statistics as the input sample. The matching of these feature statistics is continued till a convergence is obtained. In this process of matching the features of input texture with those of the output, the output gets gradually modified to look texturally similar to the input sample.

These feature based techniques give good results for stochastic structures but are not so effective on the more regular textures. Computationally these algorithms are efficient than probabilistic techniques, but usually suffer from lack of features to model higher order texture information. Moreover, the set of features to be modeled is predefined and no particular set of features works well on a wide variety of textures. Hence the feature set being not generic, these techniques look only for a specific set of features and they cannot be applied to model a texture that has a different set of features.

Probabilistic texture synthesis techniques are another class of algorithms that model the textures as samples drawn from probabilistic distributions. Given an example texture, these techniques model the interaction between individual pixels in a neighborhood, determine the local conditional distribution of the original image and synthesize a new texture by sampling from it. As opposed to feature based techniques from filter theory, they do not model any set of features that is specific of a particular class of textures. Hence this modeling of textures is generic and applicable to a wide variety of textures. However the quality of synthesis solely depends on the ability of the underlying mathematical framework to model the pixel-pixel interactions of desired order. Initially some regressive , auto-regressive and correlation based methods of this class have been proposed that were mostly linear and bi-linear in nature. Hence their application was limited to a small set of stochastic textures as they couldn't model the higher order information usually found in regular textures. Later on Markov Random Fields (**MRF**) proposed by Besag [11] have emerged as the popular model for textures because of their ability to capture the properties of a wide spectrum of textures. In (**MRF**) based methods, the input image sample is analyzed and its local conditional Probability density function (**LCPDF**) estimated. Then the output , which is initially a random noise image, is iteratively updated pixel by pixel with respect to the **LCPDF**. This iterative method of synthesizing an image from an **MRF** is known as stochastic relaxation (**SR**). Gibbs

sampler [11] and Iterative conditional modes (**ICM**) are the usual relaxation algorithms employed for sampling.

Natural textures which are abundantly found in nature are partly stochastic and partly regular in nature. **MRF** ably models these natural textures. Hence **MRF** based sampling techniques are highly effective in synthesizing a wide variety of textures. However, as these models require a detailed modeling of pixel wise interactions and due to the overwhelmingly large dimensionality of image space, estimating the underlying distribution of a texture is computationally demanding and very hard to infer unless some assumptions about the textural properties are made. Locality and stationarity are the characteristics of a texture that are usually assumed and density model usually restricted to gaussian. The success of these synthesis techniques depends on the structure of the density estimator employed and the size of the neighborhood . While stochastic textures can be readily synthesized with small order neighborhoods, large neighborhoods need to be modelled as the textures get more and more structured. Irrespective of this issues, **MRF**s are widely employed for synthesis because of this ability to model a large number of textures. The main challenges to these methods are estimating the underlying stochastic process and efficiently sampling from the estimated model. The only drawback of these techniques is that they are computationally expensive and with the size of the neighborhood the synthesis time increases in an exponential manner.

Since the Julesz conjecture , a flurry of synthesis algorithms have been developed that synthesize a texture based on an input sample. However, none of them work equally well on all types of textures. Simple probabilistic techniques and feature based techniques work well on stochastic textures and perform poorly on highly regular textures. On the other hand, **MRF** based techniques perform reasonably on regular textures as well and patch based techniques where the unit of synthesis is a patch are highly successful with regular textures. Based on these notions, texture synthesis techniques can be loosely categorized as either structural or stochastic depending on the kind of textures they work best on. Structural techniques tend to work well on regular textures with large structures which are distributed across the texture space according to placement rule. Stochastic techniques work efficiently on textures with only local variations spread across few pixels. In the next section, we will discuss a few important **2D** texture synthesis algorithms that have changed the coarse of the texture synthesis literature and research and follow it up with another section that extends them to synthesizing textures on arbitrary surfaces.

3.3 2D Texture Synthesis algorithms

While the texture synthesis literature as noted earlier has a flurry of methods built on various models of texture synthesis , we will give an overview only of a few important algorithms that laid foundation to our research work. The algorithms that we discuss are example based synthesis algorithms which synthesis a larger texture based on an example input.

Popat and picard proposed a probabilistic synthesis technique [24] where in the distribution of an input sample is summarized using a clustering mechanism. The Probability Mass Function (**PMF**)

governing the texture is obtained by populating the training data consisting of all the causal conditional neighborhoods in the input sample and fitting a set of gaussian clusters to it. The **PMF** so obtained is used to synthesize the pixels of the output in a scan-line order. Each pixel of the output texture is given a value in accordance with the **PMF**. For better quality of synthesis, the output texture can be synthesized in a hierarchical fashion. This method really works well on stochastic textures, but performs poorly on regular textures, where the structure is larger compared to that of the neighborhood. This is expected as its underlying model is an approximation and as the order of modeling gets higher, the error that comes with the approximation also increases. Moreover, the synthesis, being causal, can lose its direction if a few initial pixels generated were too far from those in the input sample.

Heeger and Bergen [13], motivated by research on human visual perception, proposed a technique for stochastic textures. This method from filter theory captures the texture properties by decomposing an image into a chosen set of linear filter responses. It starts with an input image and a noise image of desired size, constructs their Laplacian/steerable pyramids and performs histogram equalization across various pyramid levels resulting in similar pyramids. The output pyramid is then collapsed to generate the result texture. The build and collapse operations on pyramids are performed multiple times to obtain a convergence. However, this model is restricted to second order statistics and works well only on stochastic textures, doesn't capture all of the perceptual structures of natural textures and performs poorly on inhomogeneous textures, quasi textures and random mosaic textures.

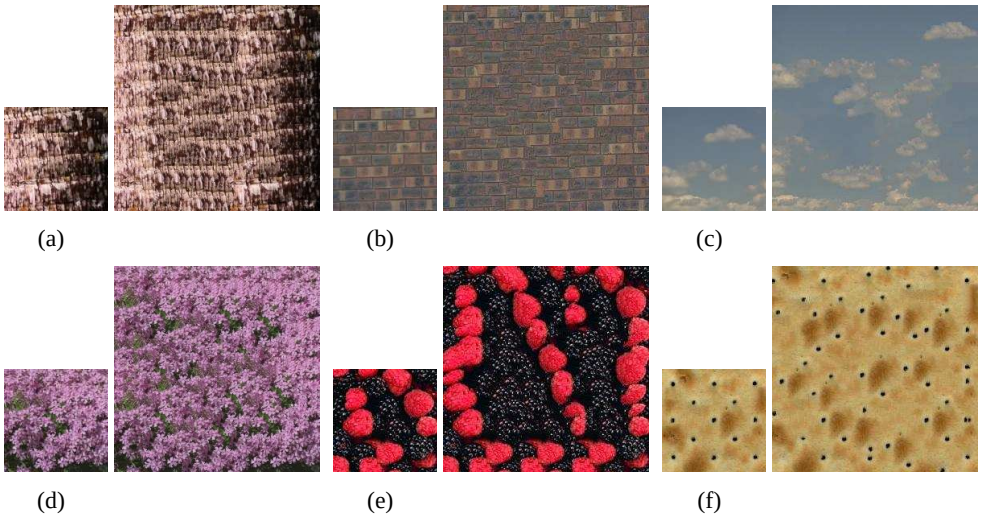


Figure 3.2 Synthesis Results of paget and Long-staff's Algorithm

paget and Long-staff,proposed a synthesis technique based on non-parametric **MRF** modeling of textures [23] in which,they addressed the limitations of clustering based approach proposed by Popat and picard. They came up with a multi-scale top down approach, where the frequency components of a texture are gradually introduced into a synthetic texture from lower to higher frequencies. Synthesis

is performed in a Multi Scale fashion, where stochastic relaxation (**SR**) is employed at a low level and the information used to constrain the synthesis at higher levels of resolution. They introduced the novel concept of pixel temperature function which serves the purpose of local annealing, avoids growing of garbage and helps in achieving global characteristics in less number of iterations. Unlike the earlier methods, this non-parametric multi-scale synthesis algorithm can successfully model and synthesize all natural textures ranging from stochastic to structured. However, this elaborate modeling incurs high computational load and the synthesis is slow.

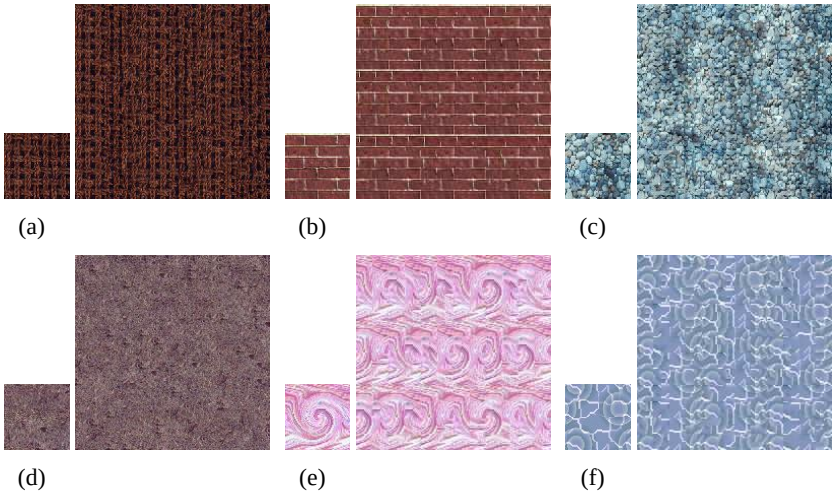


Figure 3.3 Results of De Bonet's Algorithm

De Bonet et al. [7] proposed a multi resolution sampling procedure that is a variant of the pyramid based approach of Heeger and Bergen[] discussed earlier. It improves upon the Heeger and Bergen's method by adopting a top-down synthesis approach which is akin to the one used by Paget and Longstaff [23] and also uses better texture discrimination features,which are a filter bank of first and second gaussian derivatives and Laplacian, across multiple resolutions. This improvement over the Heeger and Bergen's model [13] is capable of capturing the characteristics of a plethora of textures compared to the former method. Moreover, texture structure is also better handled than in Heeger and Bergen's method by further restricting the sampling procedure to pixels that fall within a threshold determined by texture features. Although De Bonet's method performed better than Heeger and Bergen's method for a wider variety of textures, the tuning of the threshold parameters is not intuitive and the constraints being local, the technique cannot model complex visual textures and higher order statistics. Simple addition of more complex features only over-fits the model and results in tiling.

Zhu et al. [36] combined filter theory , maximum entropy principle and **MRF** based modeling to produce a new system called **FRAME** model. It draws powerful features from filter theory , uses maximum entropy principle to find the probability distribution of the texture and uses Gibbs sampling to synthesize a new texture. Given a set of filters, at each stage , each of them is applied separately on both

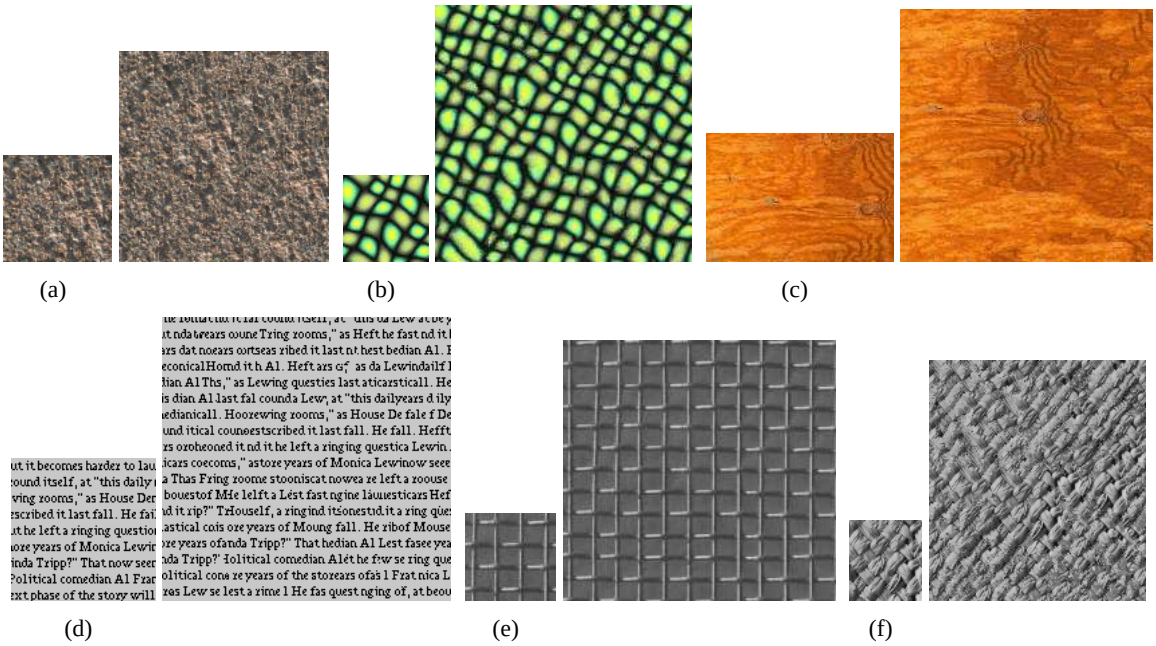
the input and to be modified output image and the histograms of the filter responses compared. The filter responses are used as the marginal estimates of the underlying distribution and the filter whose response varies the largest from input to the output is selected and used to re-estimate the probability distribution of the texture. The resultant distribution is used to modify the output using Gibbs sampler. This process is continued till a set of selected from the bank of filters models the input texture sufficiently. Unlike the previous statistical techniques, this model has the ability to capture interactions between pixels that are farther and has the theoretical framework to work well on a wide variety of textures. However, the selection of filters is a computationally intensive process , the synthesis very slow and the choice of filters cannot always be properly defined. Later, Wu et al. [34] proposed a slightly faster algorithm that avoids explicit estimation of parameters and synthesis textures directly from the filter responses using a Markov Chain Monte Carlo (**MCMC**) algorithm. This sampling algorithm is an extension of single site Gibbs sampler, converges fast and produces better results.

Portilla and Simoncelli [25] proposed a universal statistical model for texture synthesis that uses joint statistics of coefficients in a multi-scale complex wavelet representation. A similar technique to that of Heeger and Bergen [13], but where Heeger and Bergen updated the complete filter response using histogram equalization, Simoncelli and Portilla updated each point in the pyramid of filter responses with respect to the correlations using a method similar to projection onto convex sets (**POCS**). They did this by finding an orthogonal projection from the filter response of the synthetic texture to that of the original. After the projection of all filter responses, the wavelet pyramid is collapsed, further projection performed, and then the pyramid reconstructed. This iteration continues until a convergence is obtained. It is an improvement over Heeger and Bergen's method in that it always does synthesis by finding an orthogonal projection from output texture to the input and Heeger and Bergen's method is limited to a statistical modeling of order two while the number of constraints here are many and complicated. Hence this gives the present model a capability to model large number of natural textures. The heuristic strategy to select the desired features is also akin to the greedy approach observed in the maximum entropy approach by Zhu et al. [36]. The drawbacks of this method are that the choice of parameters governing the texture model cannot be guaranteed to be unique and it performs poorly on textures containing large structures.

The above mentioned works more or less established the theoretical platform for texture modeling and synthesis. The synthesis techniques that followed later dwelled more on speed, real-time rendering, efficiency and robustness with the research taking a shift in orientation from computer vision to graphics. However these new generation techniques all have their foundations in the age-proven filter theory and **MRF** based formulations.

Efros and Leung [10] proposed a non-parametric sampling scheme that avoided explicit parameterization of the texture model and instead synthesized the input texture by employing a nearest neighbor search mechanism which is an effective approximation. It models the texture as an **MRF** and grows output texture, pixel by pixel, outwards from an initial seed. Like in Popat's work [24], each pixel is synthesized conditional upon the distribution of pixels in the neighborhood, avoids the problem of es-

timation of parameters and instead substitutes it with a nearest neighbor approach. At each step, the neighborhood around a pixel is taken, queried for similar neighborhoods in the input image and the center pixel of the most similar neighborhood is copied into the current pixel. This algorithm can also be used for constrained synthesis where in the existing structure is used to fill the unknown portion of the texture. However, the process is computationally intensive due to the exhaustive search that is employed to synthesize each pixel and has the tendency to occasionally slip into wrong part of the search space and start growing garbage or get locked on to some place in the input image and start growing multiple copies of the same.



(a) (b) (c)

(d) (e) (f)

Figure 3.4 Results of Efros and Leung's Algorithm

Later, Wei and levoy [32] improved upon Efros and Leung's algorithm by accelerating the search procedure by adopting a Tree structure vector Quantization (**TSVQ**) based data structure from data compression to quicken the neighborhood search. This search technique is similar to the clustering mechanism adopted by Popat et al. [24] They also stuck to the raster scan order of synthesis of Popat as it results in better search results. As you can see in figure 3.5 this algorithm performs as well as Efros and Leung's algorithm and importantly runs two orders of magnitude faster than any of the previous techniques. The two advantages of this method are the quality of the texture generated and the speed of the synthesis procedure. This multi level synthesis algorithm is capable of synthesizing a wide variety of textures ,but suffers from the garbage growing problem also seen in most sequential synthesis algorithms.

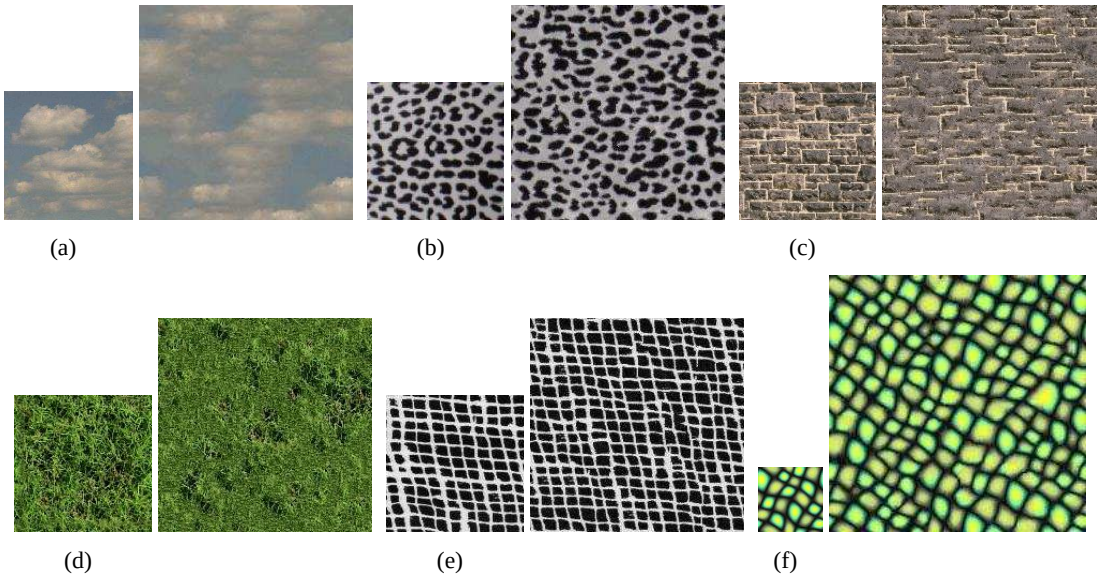


Figure 3.5 Results of Wei and Levoy's TSVQ Algorithm

The sequential non parametric algorithm of Wei and Levoy [32] and that of Efros and Leung [10] use L_2 norm for neighborhood comparison which is not a suitable measure of texture similarity as few erratic boundary pixels in a neighborhood can impair the similarity score. This results is growing of large regions and undesirable smoothing. These deficiencies were addressed by Harrison et al. [12] who proposed a non-hierarchical procedure for synthesis that adopts a non-sequential prioritized order of synthesis by using an entropy measure suggestive of interaction between neighboring pixels. They avoided L_2 norm and instead used Manhattan distance as a measure of neighborhood similarity. This improved the results and preserved the structural information in the synthesis.

While all the above algorithms are generic in nature and have varied applicability across texture spectrum, Ashikhmin et al. [2] proposed a technique that is well suited for a specific class of textures called natural textures. Flower fields, pebble, grass patches , bark etc are some of the textures on which this method works really well. This is inspired from the WL algorithm [32] by Wei and Levoy for fast synthesis. The WL algorithm, though works well on a wide variety of textures, is not suited to the class of natural textures which contain arrangements of small objects with irregular structure but familiar shapes and sizes. Moreover the L_2 norm used in the WL algorithm tends to blur out the textures and also results in uncontrolled region growth. Ashikhmin et al. addressed these issues by modifying the search procedure and limiting the search space at any step to a small set of candidates that are appropriately forward-shifted with respect to the pixels of the input already used in synthesis. Unlike WL, it neither employs **TSVQ** nor does exhaustive search in the input image, the neighborhood size required to capture the texture characteristics is also less and requires no multi-level synthesis. As a result it avoids the complexity of search that is inherent in WL algorithm. These are the two main

improvements over WL algorithm, though its applicability is limited to a small class of textures. The utility of the algorithm can also be enhanced by intuitively controlling the output by providing a target image that outlines the general characteristics of the result. The algorithm is reasonably fast, efficient and generates good results for the class of natural textures as evident in figure 3.6.

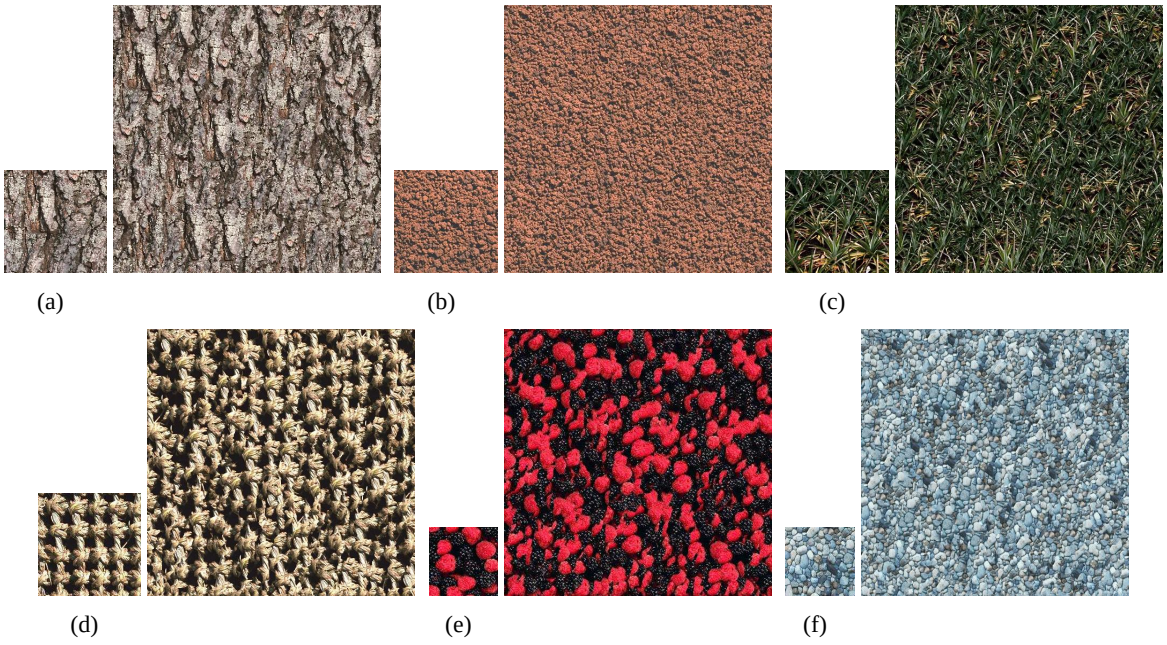


Figure 3.6 Results of Ashikhmin's Algorithm

Hertzmann et al. [15] modelled texture synthesis problem as an application of image analogies framework. Their method combines Wei and Levoy's **TSVQ** based algorithm [32] and Ashikhmin's algorithm [2] to address the issues inherent in both the methods. Wei and Levoy's algorithm suffers from smoothing and growth of unnecessary large areas due to the L_2 norm employed for estimating patch similarity, where as Ashikhmin's method suffers from discontinuities across locally grown small regions. Hertzmann et al's image analogies driven texture synthesis algorithm employs both the exhaustive search of Wei and Levoy's algorithm and coherent neighborhood search of Ashikhmin's method and synthesize a pixel based on the error measure produced by either of the methods. Hence by controlling the choice carefully at each step, it avoids the artifacts produced by the two parent methods. The results produced are much better and synthesis can be performed across multiple scales for better results. But it is computationally intensive and uses only low-level features not harnessing the potential of the image analogies framework.

The texture synthesis algorithms that have been discussed so far synthesize the output one pixel at a time. Hence they are called pixel-based synthesis techniques. These techniques are well suited for stochastic textures where the order of the structural information is local and confined to a few pixels. However, when the input texture is complex and consists of arrangement of bigger primitives, pixel-

based synthesis results in artifacts such as smoothing, unnecessary growing of regions with deformed primitives. Moreover, for most complex textures very few pixels actually have a choice of values that can be assigned to them and are determined by what has been synthesized so far. This means that a lot of time is wasted on pixels which are already determined by the higher order structural constraints. These are the two main limitations of pixel-based techniques. The key to solve these problems is identifying the relevant unit of synthesis called texel, which is a textural element and synthesizing the output texel by texel rather than pixel by pixel. Texels are usually small patches of a texture that capture enough local information and minimal global information i.e the arrangement of these texels. Hence the techniques which adopt this strategy of synthesizing a texture in a series of patches are called patch based synthesis techniques. Determining what are the patches and how they are put together are the two main challenges to this class of techniques.

Xu et al. [35] proposed a patch-based synthesis technique called Chaos mosaic, where in the input texture is tiled and random blocks chosen from the resulting texture and re-distributed across to produce a visually realistic synthetic texture. Chaos mosaic not only preserves local features of the input texture which is essential for visual similarity, but also provides a visually stochastic and even global distribution of the features. Unlike the conventional statistical based methods, it doesn't analyze and model the input sample, and no constraint is imposed on the result texture to adhere to a probability distribution. It is fast and memory efficient and the resulting texture can be succinctly represented in a compact representation known as visual texture. This gives the ability to render the texture in a procedural manner.The main advantages of this algorithm are its ability to synthesize large samples of a texture quickly in a memory efficient manner and tailor-madeness to be rendered in a procedural fashion. It has advantage over traditional statistical methods in that it avoids system latency when the texture is too big to fit in the system memory or when there isn't enough storage space for it. Chaos Mosaic works well for all stochastic textures and artifacts appear in cases of structured textures. The problem of artifacts and seams are addressed to an extent by employing cross-edge filtering along the boundaries of the displaced random blocks. However, the mismatch of patches along the boundaries is still a problem when it comes to synthesizing structured textures.

Later Liang et al. [19] proposed a patch-based sampling technique, that addressed the problems of seams and artifacts in Chaos mosaic [35]. Liang et al. improved upon chaos mosaic by incorporating the notions of **MRF** based texture model and employing non-parametric estimation of local conditional **MRF** density to sample the patches of the output texture. Hence in every step, a new patch which best agrees with the output synthesized is introduced. This avoids the mismatch of features across patch boundaries that is inherent in chaos mosaic. The crux of this algorithm involves searching for meaningful input texture patches to be pasted in the output. Kd-tree, Quad-Tree pyramid and PCA analysis are usually employed to speed the search process. Alpha-blending is applied across neighboring patches and this avoids seams and artifacts. As a result, this method works well on all varieties of textures ranging from stochastic to structured. It also works well on natural textures where in Efros and Leung's [10] and Wei and Levoy's [32] methods fail. This work also draws comparisons with the

work on Texture quilting by Efros and Freeman [9]. While Liang et al. consider the issues of speed and constrained texture synthesis, texture quilting dwells upon the concept of texture transfer.

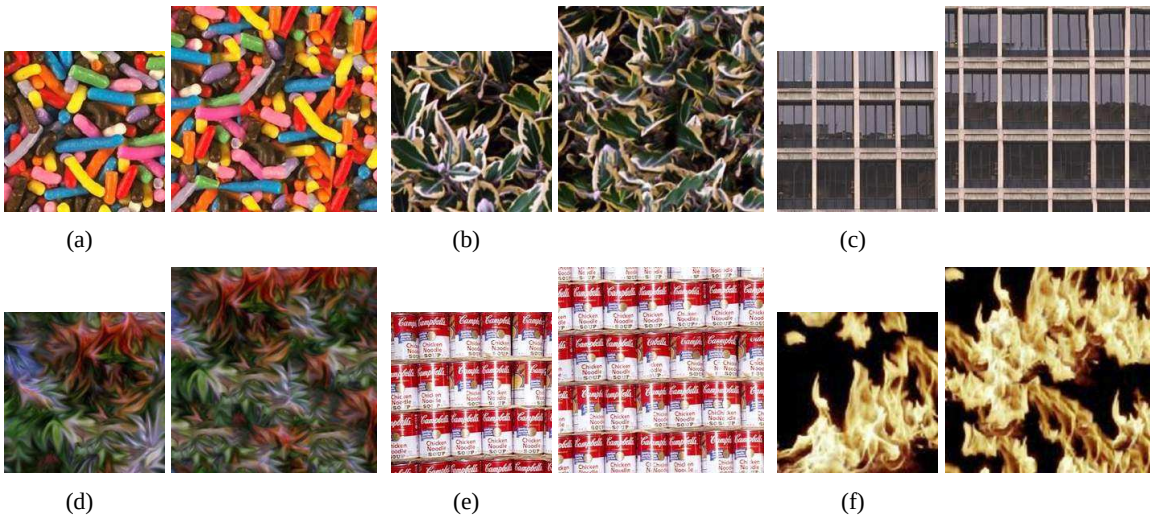


Figure 3.7 Results of Liang et al.'s Patch based sampling Algorithm

Efros and Freeman developed image quiting [9], a novel patch based synthesis technique, which is a concurrent work to Liang et al's approach [19] and very much similar to it. Like Liang et al's method, the synthesis proceeds in a series of steps in each of which a patch is selected from the input and pasted in the output overlapping with the already pasted neighboring patches. The two approaches however differ in the way they handle overlap regions. Where Liang et al's method employs feathering i.e weighed alpha-blending, image quilting uses minimal error boundary cut to determine a boundary along which the difference in pixels is minimum. This algorithm can also be extended so as to perform texture transfer by imposing additional constraints to make the quilting patches agree with a correspondence target image. This algorithm performs as well as Liang et al's method. However it suffers from excessive repetitions and distorted boundaries.

3.4 Surface Texture Synthesis

Computer Graphics applications often use surface textures to give an illusion of fine detail with out detailed geometric modeling. Algorithms exist for synthesizing a wide variety of textures on the **2D** plane from example texture. However these methods cannot be extended to texture arbitrary topological surfaces due to lack of continuous surface parameterization. One solution is to paste such synthesized planar textures on to the $3D$ objects. But this results in distortions or discontinuities. An effective approach to tackle this problem is to synthesize the texture directly over the surface. This works really well as many natural and man-made surface patterns are created by interactions between texture ele-

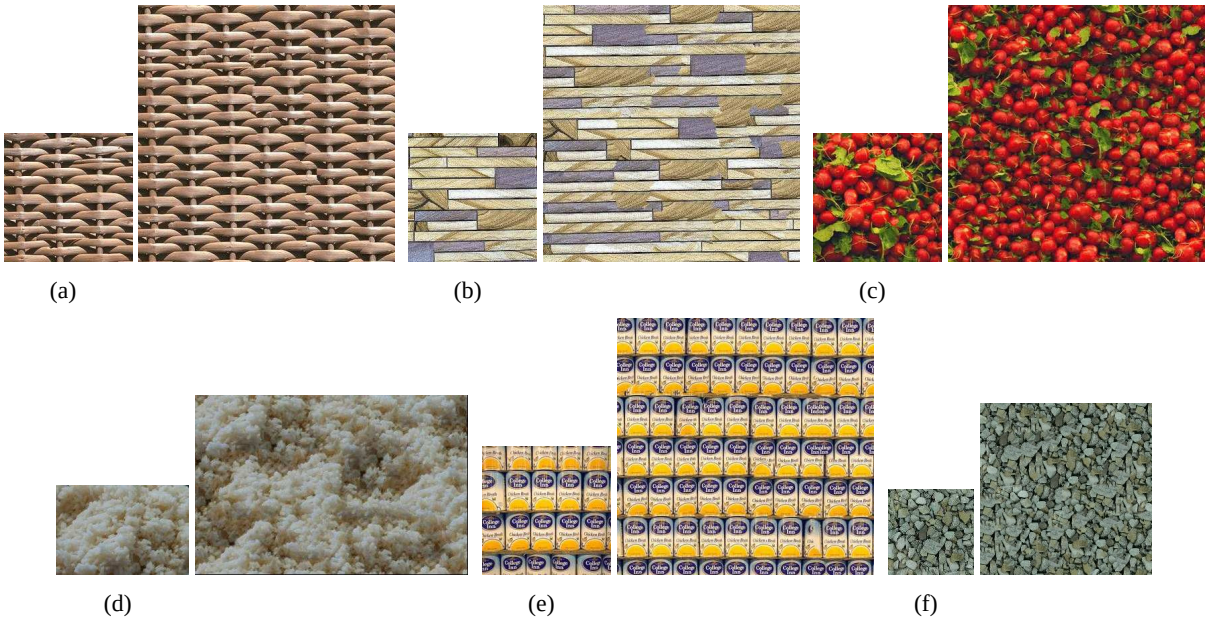


Figure 3.8 Results of Efros and Freeman's Image Quilting Algorithm

ments and surface geometry. Such algorithms which synthesize the texture directly over a polygonal mesh placed over the surface are called surface synthesis algorithms.

Praun et al. [26]proposed a technique in which the candidate texture patches in the input sample image are identified and repeatedly pasted on the mesh model of a surface until it is full covered. The collection of these overlapped texture patches is called lapped textures. This method has been inspired from chaos-mosaic [35] proposed for quick synthesis of a planar texture from example. Praun et al's algorithm identifies small portions on the mesh called surface patches each of which can be locally mapped on to the **2D** plane easily and repeatedly pastes texture patches across all of them till the mesh is completely covered. To prevent seams and texture distortion, orientation and scale of the texture patches are aligned with those of the surface patches and alpha-blending applied across overlapping patches. The scale and orientation of each of the surface patches is derived from surface tangential vector field, which is partially obtained through user intervention and then interpolated to find the same across all the vertices on the mesh during the pre-processing stage. In each paste operation, an un-textured point on the mesh is identified and a surface patch homeomorphic to a disc grown around it. This surface patch is parameterized into the texture space by aligning the axis of the texture patch with the tangential vector field of the surface patch. This parametric-optimization is solved using a sparse linear system and the resulting patch-mapping saved. Once the patch placements are computed, the texture model can be rendered in real-time using compositing operations, either into a texture atlas during pre-process or directly rendering the surface patches during run time. This method of synthesizing texture on **3D** meshes is very practical allowing real-time texturing of **3D** objects. On the flip side, visible seams are

produced when the input sample contains low frequency component and also when viewed up-close in the case of structured textures. Some visual artifacts also appear due to poor field sampling of the tangential vectors.

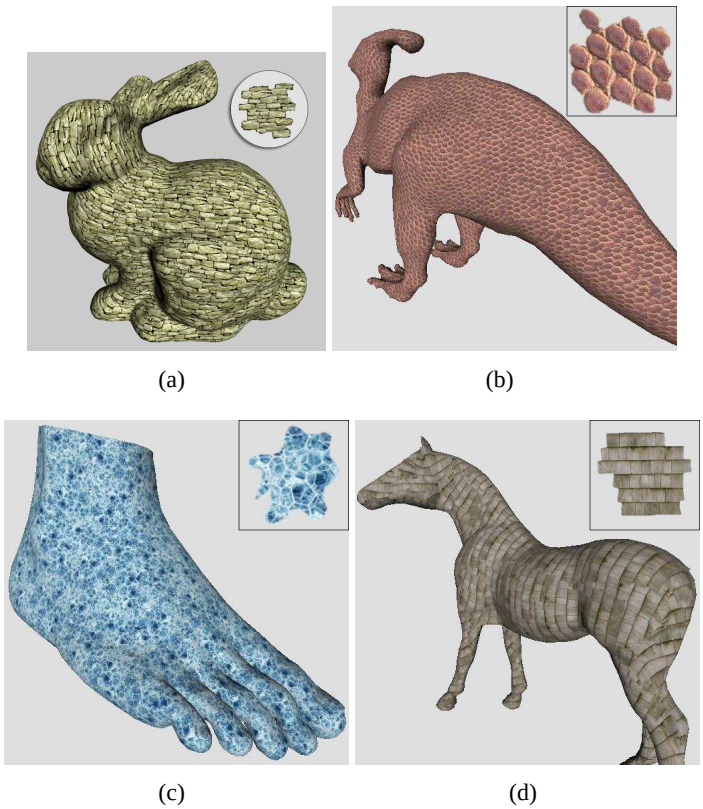


Figure 3.9 Results of Lapped Textures

Turk et al. [31] borrowed ideas from the sample based synthesis techniques employed on rectangular pixel lattices and proposed a point based sampling technique to synthesize texture on arbitrary polygonal surfaces. It employs hierarchal mesh-model representation which is similar to the gaussian image pyramid representation performs synthesis from lower to higher resolution levels of it. Like lapped textures [26], it also requires a user-specified surface tangential vector field at a sparse set of points given which it calculates the values at other points on the mesh. It performs synthesis employing four operations namely 1)interpolation 2) Low-pass filtering 3)Up-sampling 4) Down-sampling on the mesh model and these are very much similar to their corresponding versions in image-pyramids in **2D**. The points on the mesh are order in accordance with the flow of the vector field in such a way that visiting them in this order will sweep across the surface from one end to the other. Each point is colored by examining the colors values of the neighborhood points that have already been textured and identifying a similar neighborhood in the input sample.This assignment of color is done in a hierarchal fashion from sparser to dense mesh-levels. This point by point texturing scheme is inspired from the work of Wei and

levoy [32] on non-parametric sampling technique for **2D** texture synthesis. The synthesized texture is transferred on to a texture atlas using a technique proposed by Soucy et al [28]. This technique performs better than lapped textures [26] over a wide variety of textures over surfaces of arbitrary topologies.

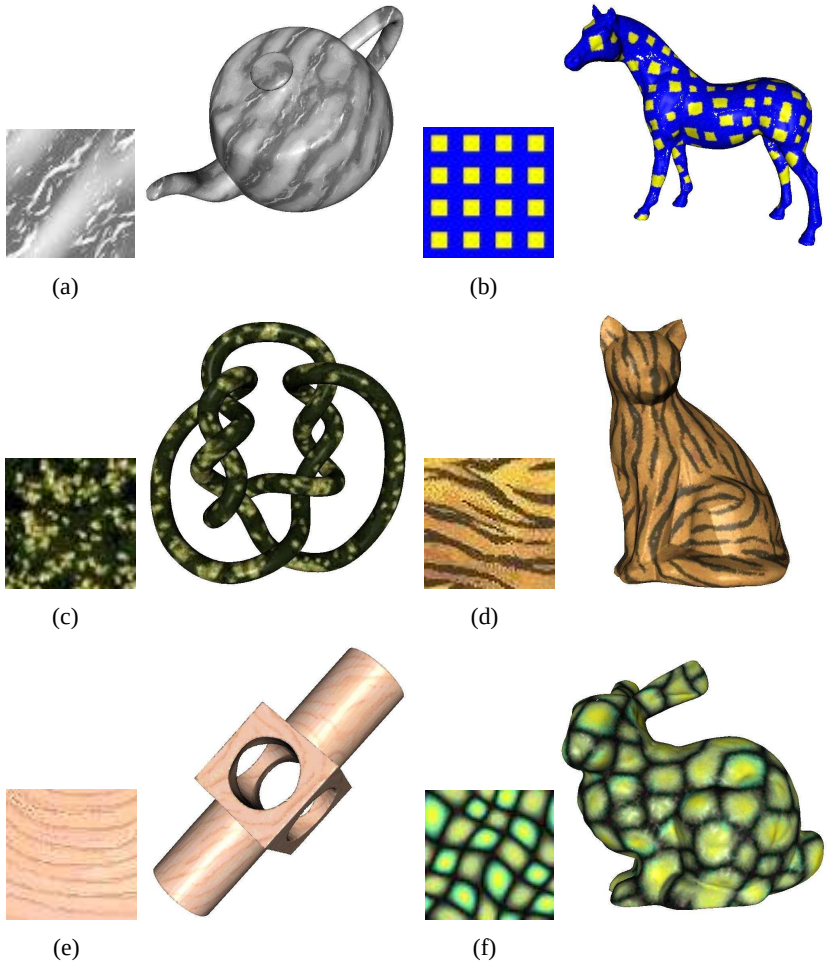


Figure 3.10 Results of Wei and Levoy's Surface Synthesis Algorithm

Concurrently with Turk et al., Wei and Levoy proposed an extension [33] to their **TSVQ** based **2D** synthesis technique [32] to address the challenges of synthesizing textures on arbitrary manifold surfaces. They introduced two modifications to their existing **2D** algorithm. First and foremost, the vertices are processed in a random order instead of the scan-line fashion employed for rectangular domains. Second, the rectangular parameterization of the output domain is replaced with a surface tangential vector field, coupled with a scale factor derived from mesh vertex density. Like the earlier techniques, the surface tangential vector field is obtained with user help or gradually evaluated using a relaxation procedure. At each vertex, a local parameterization is obtained by flattening the neighboring triangles and then sampling a rectangular neighborhood from it. This neighborhood is searched for

matches in the input sample and the color of the vertex obtained. For effective synthesis, the algorithm builds a mesh hierarchy and carries a two-pass synthesis at each level from lower to higher resolutions. Though this algorithm was developed in concurrence with Turk et al's [31] technique and the results of these techniques being similar, there are differences in the methodologies. Turk's algorithm creates a smooth vector field where as Wei and Levoy's technique employs symmetric and random vector fields. Turk's algorithm visits the vertices of the mesh in an sweeping order creating a parent-child relationship, where as Wei and Levoy's technique visits the vertices randomly. There is also a difference in the ways neighborhoods are created in both the methods. Turk's algorithm uses surface matching where as the current algorithm employs flattening and resampling.

3.4.1 Short-comings of Simple Color Textures And Need For Better Models

All the above discussed example based synthesis algorithms synthesize simple color information on the surface of an object. These values are static and do not imbibe material information of the object. Hence they do not model the interactions between material surface and light conditions that results in sub-surface scattering phenomena such as shadows, inter-reflections, self-occlusions and specularities, which affect the visual appearance of an object. Moreover, the images that are texture mapped imbibe the lighting conditions under which they have been captured. Hence the texture model so obtained looks good in lighting conditions that match those of the input image, but very poor when viewed under different lighting conditions. Due to these limitations, simple color texturing and synthesis falls short of accomplishing the task of realistic modeling and rendering real world objects.

3.5 Synthesis algorithms for Reflectance Textures

The synthesis of reflectance textures from examples is conceptually different from the **2D** texture synthesis. A collection of images of a particular surface acquired under various lighting conditions cannot be treated as an independent collection of **2D** textures. There are strong correlations between the sampled images, as all of them are instances of a unique underlying physical surface. These correlations have to be maintained while synthesizing a novel reflectance texture.

Liu et al. [20] presented a novel approach to synthetically generate bidirectional texture functions (BTFs) of real-world surfaces in which they used a texture's height-field along with an albedo map as an intermediate representation for **BTF**. This representation is reconstructed from the texture examples, using shape-from-shading techniques. Then, a synthesis scheme is applied directly to the height-field, using non-parametric sampling , resulting in a representation of a novel texture from which a new **BTF** is derived.

Leung and Malik [17] suggest using the **3D** texton map as a basis for generating a novel **3D** texture. This approach is similar in spirit to [20] where a texton map is used as an intermediate compact

representation. The texture's **BTF** can be derived from this representation similarly to the height-field map.

Buoyed by the above works, Tong et al. [29] presented a method that uses the texton map representation as a basis for **BTF** synthesis directly on a **3D** object. This algorithm takes as input a **BTF** sample and a polygonal mesh and synthesizes **BTF** on the mesh such that the resultant model is perceptually similar to the input sample and exhibits a consistent meso structure across the view and light spaces. The input **BTF** sample is first analyzed and a vocabulary of **3D** textons generated. This vocabulary is later used to generate a texton map t_{in} of the input sample and then a texton space s generated. The texton map t_{in} is treated as a texture sample and a surface texton map t_{out} generated by incrementally assigning each mesh vertex v a texton label t and a texture co-ordinate (a, b). The color of the surface textons is evaluated at run time and rendered. This work is very much similar to Turk et al. [31] and Wei-Levoy's [33] and can be considered as their extension for reflectance texture maps.

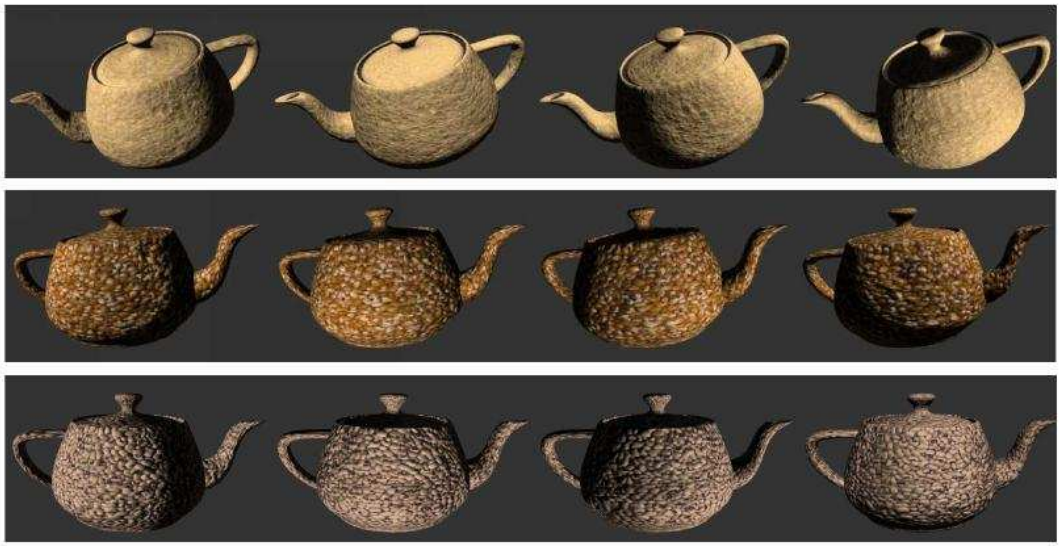


Figure 3.11 Sponge, Popcorn Kernels and Peas textured Teapots illuminated in different light conditions

Later, Hel-Or et al. [14] proposed an approach for synthesizing Polynomial Texture maps (**PTM**) on arbitrary manifold surfaces. This is an extension of the block-based texture synthesis methods from working on images containing color values, to images of reflectance functions. They viewed the **PTM** as a texture of functions rather than a texture of values and regarded it as a realization of a Markovian process in the spatial domain. The stochastic process of synthesis is performed over functions rather than over values. This same approach allows any texture synthesis method that compares pixel colors to be extended in the analogous manner to support the synthesis of reflectance function textures.

Chapter 4

Realistic Rendering of Real World Objects

Realistic rendering of real world objects is an important area of computer graphics. It is used in a variety of applications, the most prominent of them being movies, games and archival of historical artifacts. Real world objects are characterized by their shape/geometry as well as the surface properties. To faithfully model a real world object, both the shape and surface properties of the object have to be correctly captured and then rendered through the graphics pipeline. In this chapter, we will outline our approach for realistic rendering of large scale objects.

Figure 4.1 On the left is a photograph of Michelangelo's David. On the right is a computer rendering made from a geometric model

When dealing with objects of large scale, it is difficult to capture these details at a fine level as the capture devices have limited resolution while working at large scales. One could handle this problem using a very high resolution shape models of the parts of the object and fitting them together [18]. However, this approach has a number of shortcomings: i) A shape model that can capture the surface details would be extremely large, ii) Assembling a single model from that of a large number of parts is often labor intensive.

On the other hand, capturing high resolution images is often easy due to the availability of low cost and high resolution digital cameras. Hence Image based modeling techniques, which facilitate rendering of large scale models of an object by augmenting the surface texture with retrieved shape information, evolved as an effective manner to accomplish this otherwise complex task.

4.1 Image Based Modeling of 3D Objects

Images are the most abundant source of visual and structural information of the real world. They are capable of capturing high level object properties effectively. Hence, image based modeling techniques [28, 27, 4] have emerged as an effective approach for realistic rendering of **3D** objects, where multi-view geometry is utilized in directly synthesizing an unseen view of an object from nearby views without explicit surface reconstruction. Multi view modeling methods on the other hand use a set of images of the object, register them and recover the 3D locations of points. A standard mesh model is derived from the point cloud, which is then texture mapped using the images that were used to derive the shape. Both approaches combine the pictorial details obtained from the individual photographs captured, to the shape information of the object inferred from the collection. While the first approach often leads to realistic rendering of unseen views, it lacks the flexibility of **3D** model based visualization.

We notice that the traditional object models capture the shape information in the polygonal mesh representation, while the reflectance and surface properties are relegated to the texture. Hence the method of pasting surface texture information on the coarse mesh model of an object is an effective procedure to accomplish this task of representation and rendering of real world objects. Many texture synthesis algorithms [24, 13, 7, 25, 10, 32, 2, 19, 9] have been developed to generate large samples of texture from scanned photographs. These methods are effective and make the texture mapping process more efficient and robust by facilitating the generation of textures of any required size. These have been later extended to synthesize texture directly over **3D** models and arbitrary manifold surfaces [26, 31, 33]. However, unlike traditional texture synthesis, where the goal is to generate a new texture patch that retains simple color distribution of the original, the objective here is to capture the surface properties far more faithfully, including the effects of small scale height variations on the surface and generate a new texture patch that retains the characteristics of the surface material.

4.2 Reflectance Properties of Natural Materials

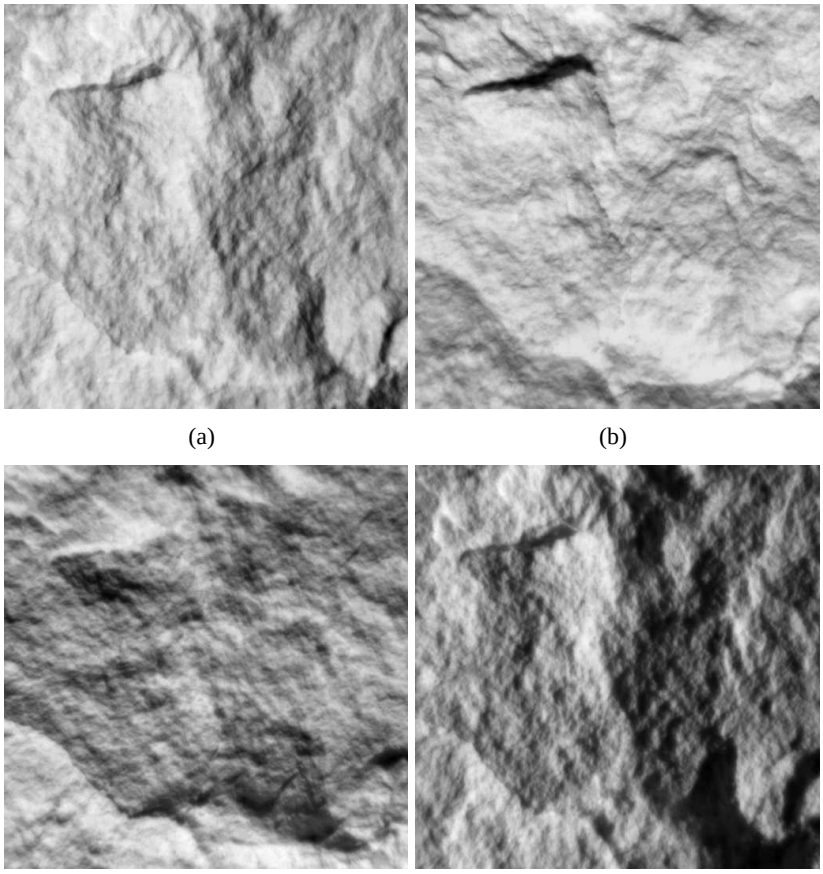


Figure 4.2 Images of a rough plaster surface obtained under varying light conditions. Note the change in surface appearance in each of (a), (b), (c) and (d)

The visual characteristics of natural surfaces arise from the variation of two properties across its surface: i) the variation in normals, and ii) reflectance. These cause effects such as shadows, self occlusions, inter-reflections, and specularity, which affect the visual appearance of the surface. As a result, a surface looks considerably different under different lighting and viewing conditions. These effects are observed in all natural surface reliefs that are abundant in real world.

Simple color texture models ignore these two properties of the natural textures. Hence they cannot model these variations in visual appearance caused under varying illumination/viewing conditions. Moreover, the images that are texture mapped on to the mesh models inherently contain the lighting conditions under which they have been captured. Hence the texture model so obtained looks good in lighting conditions similar to that of the available images, but very poor when viewed under different

lighting conditions. Hence simple color texture mapping of mesh models is insufficient and falls short of accomplishing the task of realistically modeling and rendering real world objects.

The characterization of surface reflectance properties is essential to achieve realistic rendering. The reflectance properties of a surface affect its appearance under the influence of changing light and viewing conditions. This led to the study of relation between surface appearance and illumination/viewing conditions of natural material surfaces. The concept of illumination dependent texture, anologous to the concept of **3D** texture, was introduced. This led to further investigation into the problem of representation, recognition, synthesis of natural materials and their rendering under arbitrary viewing/lighting conditions [17]. Image based re-lighting techniques [22, 6, 3] have been used to model the surface reflectance properties of natural materials. In these techniques, multiple images of the object/surface are captured under different lighting/view point conditions and then the variations in visual appearance modelled as Reflectance Texture Maps as discussed in chapter 3.

4.3 Synthesis of Reflectance Texture Maps

Techniques to synthesize **2D** textures on arbitrary shaped objects [26, 31, 21, 33] have also been extended to synthesis reflectance texture maps on the same [29]. In these techniques, the synthesis starts from an arbitrary patch and then it grows on till all the mesh-model is covered. The only constraint imposed on the synthesis process is that a patch to be synthesized agrees with the already synthesized neighboring patches. This constraint makes sure that no visible seams appear on the textured model. Using this approache, **PTM**s have been efficiently synthesized over **3D** models and rendered [14].

4.3.1 Relevance to Realistic Rendering

The above mentioned **3D** texturing algorithms when coupled with image based modeling techniques provide an effective platform for realistic modeling of real world objects. Pictorial information of an object can be obtained in a small set of images and later used to constrain the procedure of texture synthesis, which is otherwise unconstrained (except for inter-patch consistency), on its mesh model. This hybrid approach provides an effective way to synthesize the texture model of an object based on its real world appearance. We use the above notions of coupling image based modeling methods and texture synthesis techniques to pose the task of realistic modeling and rendering of **3D** objects as image constrained texture synthesis problem.

4.4 Our Work: Image Based PTM synthesis

We want to address this problem of constrained **3D** texturing of mesh-models to make them more realistic and near to their real world counter-parts. Our goal is to capture a small set of images of the object under known lighting conditions, and use these to systematically synthesize a reflectance model of the object from a sample **PTM** of the object's material. These sparse set of images decide the sub-samples of the input sample PTM_{in} that are stitched across the mesh model so that the resultant model would behave more similar to its real world counterpart.

This work is inspired from two works, one being that of Efros et al. [9] for effective Texture transfer and the other that of Yacov Hel-Or et al. [14] for synthesizing **PTM** models of **3D** objects. We present a method to effectively synthesize the reflectance model of a real world object from a sample **PTM** of its material, using a small set of images captured of the object under different known lighting conditions as constraints, so as to make the texture model appear realistic and behave similar to the original. The **PTM** model so generated can be efficiently rendered under arbitrary lighting conditions to generate novel views of the object.

4.4.1 Constrained PTM synthesis

Given a **PTM** sample and a triangular mesh model of an object, small patches extracted from the sample can be seamlessly stitched across the mesh model and a **PTM** model of the object synthesized. The **PTM** model so synthesized behaves like a real world object in terms of its visual appearance under varying light conditions. These patch based texture synthesis algorithms, when coupled with image based modelling techniques, provide an effective approach to synthesize the **3D** texture models of real world objects. A set of images of the object captured under varying light and viewpoints decide the set of texture patches that are stitched across the mesh model. Hence the texture model so obtained not only looks realistic but also similar to its real world counter part.

Our work essentially builds on the work by Efros et al. [9] for Texture Transfer and the work by Yacov Hel-or et al. [14] for **PTM** modelling of **3D** objects. We suggest an approach to generate the reflectance texture model of a real world object from a **PTM** sample of the object material and a set of images of the object. We extended the patch based **PTM** synthesis algorithm to also include the image based information in influencing the selection of the texture patches so as to make the resultant texture model more similar to the object. The **PTM** model so obtained can be used to generate novel views of an object.

In the next section, we detail our method for **PTM** modeling of planar rectangular surfaces and discuss the algorithm for **3D** seperately in the next chapter as it differs considerably from the algorithm for planar objects and also due to the increasing number of challenges posed by the task.

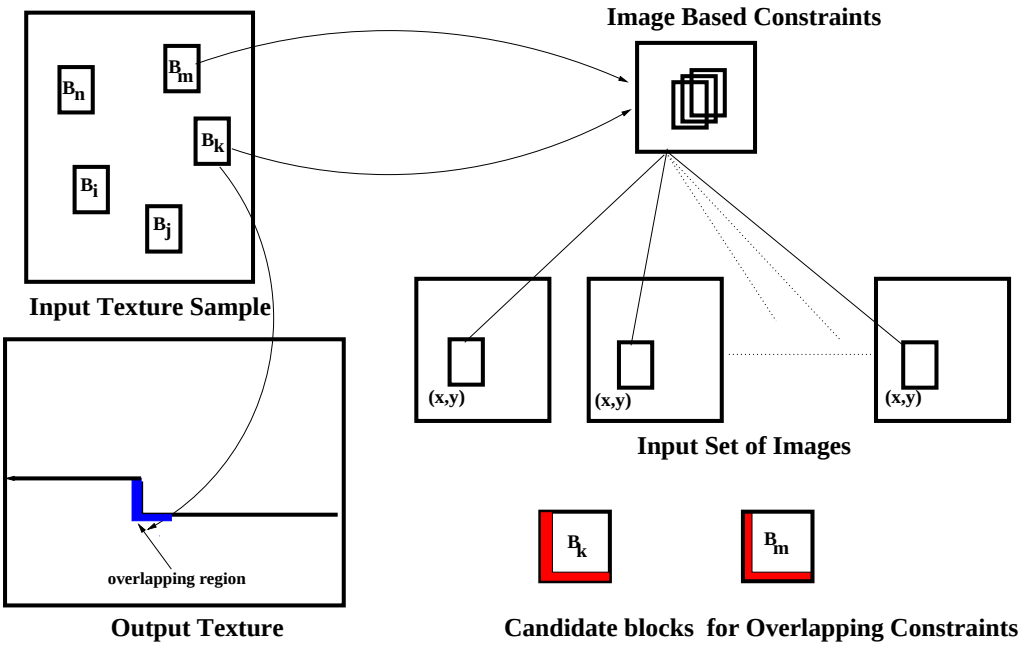


Figure 4.3 Blocks from Input sample are checked for Image Based and Overlapping Constraints and the best ones pasted into the Output

4.4.2 Image constrained PTM synthesis for Planar Rectangular Surfaces

In this section, we explain in detail our hybrid approach, that couples the Patch Based **2D** texture synthesis algorithm [19, 9] and the **PTM** synthesis algorithm by Yacov Hel-or et.al [14], to synthesize the texture model of a planar rectangular surface from a sample **PTM** of the surface material and a set of images of the surface taken under various known lighting conditions.

The synthesis algorithm takes as input, a sample texture PTM_{in}, a sparse set of images $\{I_n\}$ of the object as constraints and generates the reflectance model PTM_{out} of the same. It uses patches taken from the input sample PTM_{in} as the building blocks to synthesize the output texture PTM_{out}. At each step k, a candidate block B_k is taken from PTM_{in} and stitched into PTM_{out} with an overlap W_e between neighboring blocks and then blended in the overlapping region. The texture map so obtained can be used to generate novel views of the object under arbitrary lighting conditions.

The selection strategy of candidate block B_k, that is stitched in to the output texture PTM_{out} at every step k, is the core of our algorithm. The output texture PTM_{out} is traversed in a raster scan fashion from left to right starting at the lower left corner and moving upwards. At each step k, a candidate block B_k is selected from PTM_{in} and pasted at the next position (x, y). The selection of the patch B_k is governed by two constraints namely

1 Image based constraints

2 Overlapping constraints

4.4.2.1 Image Based Constraints

The set of images $\{I_n\}$ which are captured under light positions (l_{un}, l_{vn}) decide the candidate patches which together make up the output texture. At each step k, the texture block B_k from PTM_{in}, which is selected to go into the next position (x, y) in PTM_{out}, should agree with the set of image blocks $\{b(I_n, x, y)\}$ that are located at position (x, y) in image set $\{I_n\}$. Let the **PTM** evaluation function be denoted by $f(P, (l_u, l_v))$, where P is a **PTM** patch and (l_u, l_v) the projection of unit light vector onto the texture co-ordinate system. This function returns as output the image obtained by evaluating the input sample with the given light vector. The patch B_k when evaluated with the light vector (l_{un}, l_{vn}) corresponding to the image I_n, should result in an image patch $f(B_k, (l_{un}, l_{vn}))$ that matches the block $b(I_n, x, y)$. Hence each image I_n of the set $\{I_n\}$ imposes constraint on the selection of the texture patches $\{B_k\}$ that together build the output PTM_{out}. These together constitute the Image based constraints involved in the synthesis. At each step k, the texture blocks $\{B\}$ from the input sample PTM_{in} are ranked according to a scoring measure S which is given as follows:

$$S(B) \quad = \quad \sum_{n=1}^{N} \|f(B, (l_{un}, l_{vn})) - b(I_n, x, y)\|_2 \tag{4.1}$$

The blocks from the input sample PTM_{in} are ranked according to the scoring mechanism S and a top few of them selected as candidate blocks for the next stage of selection.

4.4.2.2 Overlapping Constraints

At every step k of the synthesis procedure, the patch B_k that is selected to go into the output texture PTM_{out} should also agree with the patches $\{B_0, B_1, ..., B_{k-1}\}$ that have so far been pasted in the previous steps. The candidate block B_k which is currently being pasted should agree with its neighboring patches in the overlapping region. This constitutes the overlapping constraint and is a must for seamless stitching of input blocks.

The set of candidates which are selected based on image based constraints in step(1) are again ranked based on their overlapping measure. $L2$ norm is calculated over the difference of luminance coefficients in the overlapping region between PTM_{out} that has been synthesized so far and each B of the candidate blocks picked by step(1). The norm is calculated with the coefficients of both PTM_{out} and block B in the overlapping region transformed to a orthogonal space so that the distance between functions is same as that between function coefficients. The block B_k with minimal error measure is introduced into the output PTM_{out}.

4.4.2.3

The complete algorithm is outlined as follows:

Algorithm 1: The Constrained PTM Synthesis Algorithm

1. Traverse PTM_{out} in a raster-scan order block by block starting at lower-left corner.;
2. At every new position (x, y), select a small set s of candidate blocks from PTM_{in} using the image based constraints.;
3. Pick the best block B_k among the set s which best fits the overlapping constraint.;
4. Paste the block B_k at the location (x, y) in the output texture PTM_{out} and blend it in the overlapping region.;

Alpha-blending is usually employed to blend the texture coefficients in the overlapping regions. The texture coefficients are transformed to an orthogonal space before blending is employed.After blending the coefficients are transformed back by applying an inverse transformation.

The above approach generates a **PTM** model of the given surface that behaves not only realistic but similar to the planar object being modeled and it can be used to generate novel views of the object.

In Figure 4.4, we showed the results of our **2D** Synthesis algorithm on three different surfaces. Sample **PTM**s of variants of plaster surface were created from their high resolution images. Images 1(a),2(a) and 3(a) show the high resolution images used in the creation of their corresponding sample **PTM**s. For each object, a sparse set of its low resolution images and sample **PTM** were used to synthesize its texture model and the model shown at the coarser resolution at which the object was captured, as well as at a higher resolution. The material information present at high resolution affirms our synthesis procedure.Images 1(b), 2(b) and 3(b) are the objects' views captured at low resolution and 1(c), 2(c) and 3(c) represent the corresponding views of their texture models generated using our method. Notice the visual likeness between the object views and the generated views of the texture models. Images 1(d),2(d) ad 3(d) are the higher resolution views of 1(c), 2(c) and 3(c) respectively and 1(e), 2(e) and 3(e) were obtained by scaling images 1(b), 2(b) and 3(b) respectively. The higher resolution views (1(d),2(d) ad 3(d))of the texture models exude more material information, less blur and loss in detail compared to the scaled versions (1(e), 2(e) and 3(e)) of the objects' views.

In figure 4.5, we demonstrate the superiority of our method over the unconstrained synthesis technique by comparing the views generated by both with the original views of the object. It can be observed that our results in the middle row bear more resemblance to the object views than those in the last row generated by relighting the model that is obtained by employing unconstrained synthesis technique.

4.5 Conclusion

The above results demonstrate the potential of our hybrid approach which couples image-based modeling and texture synthesis techniques to synthesize high resolution reflectance models of planar surfaces which behave not only realistic but more similar to their real-world counterparts. Our Method considerably differs from the modeling techniques which solely map color information onto a mesh model of the object using image registration in two aspects namely 1) Our method in addition to modeling the color information , also incorporates the dynamic change in visual appearance which is caused by interaction between the surface material and light conditions where as image-based methods suffer from unrealistic shadows and color changes.

2) Our method generates a high resolution reflectance map where as the image-based techniques are limited by the resolution of images captured of the object.

In the next chapter, we discuss our algorithm for **PTM** modeling of **3D** objects and rendering efficiently.

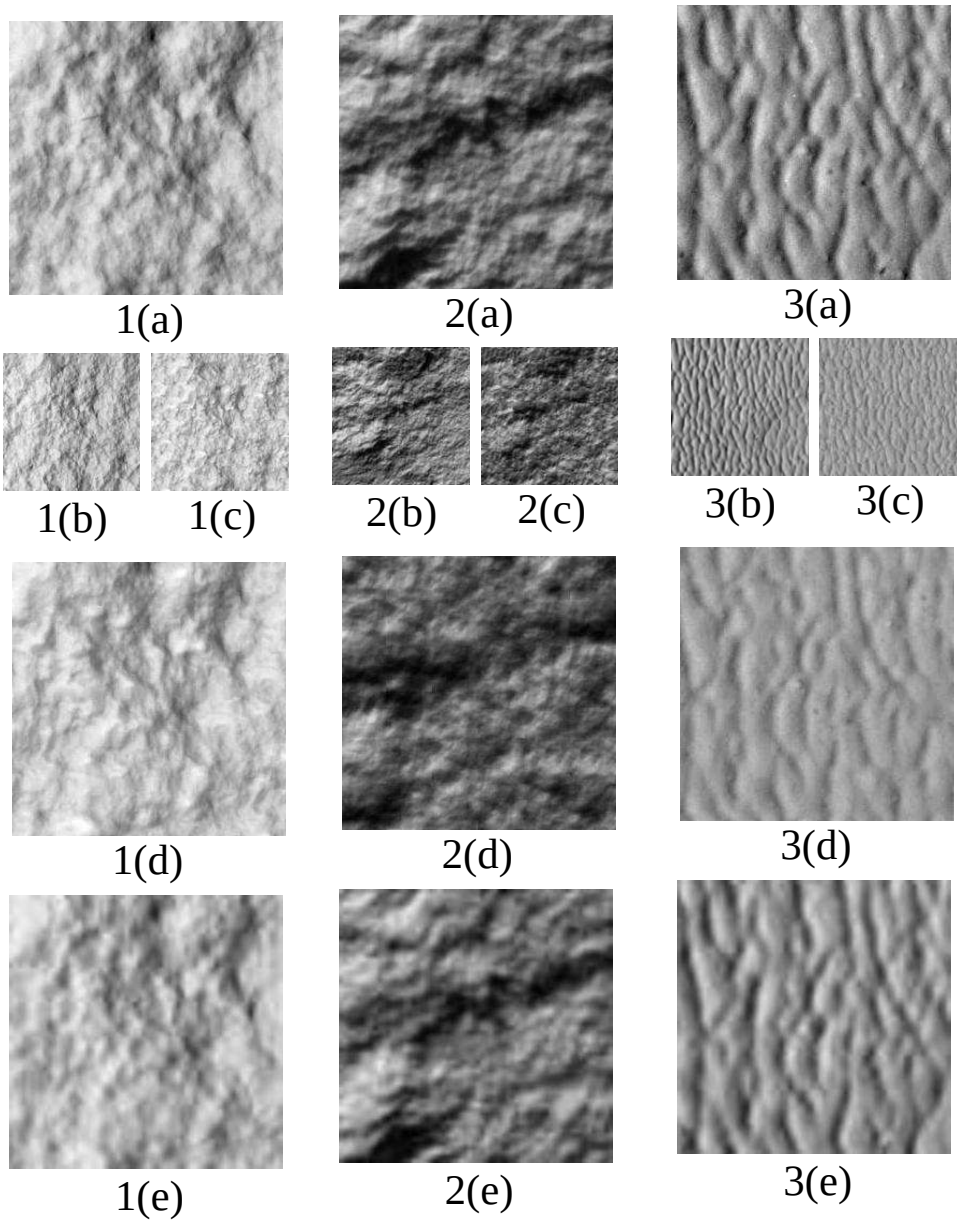
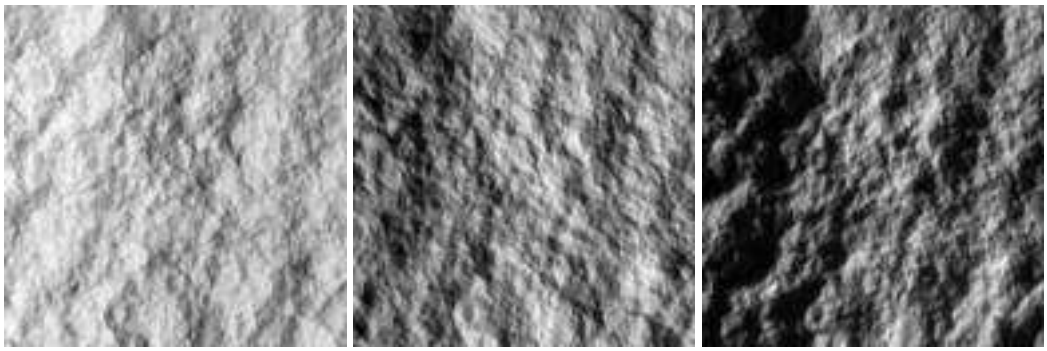
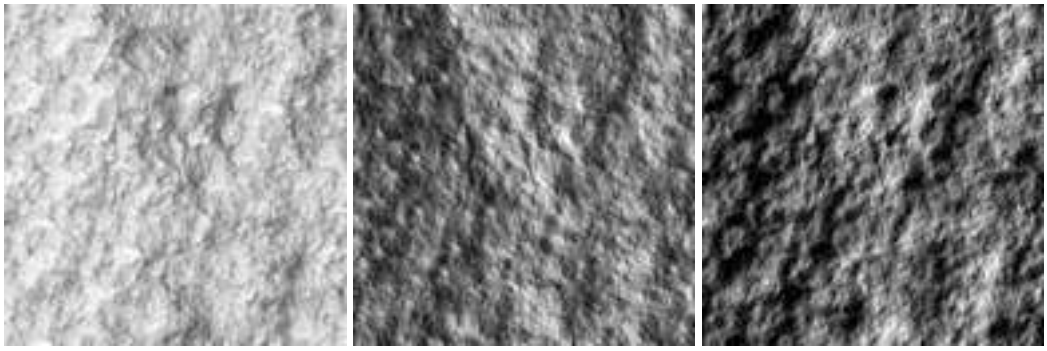


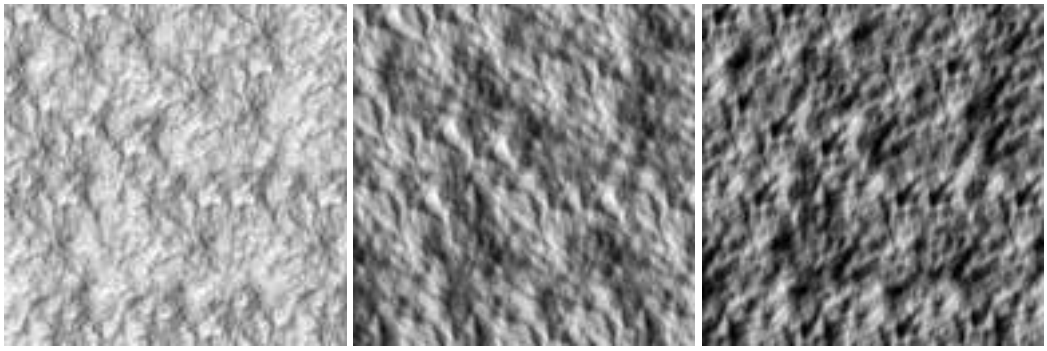
Figure 4.4 Demonstration of Our 2D Synthesis Algorithm for Natural Material Surfaces:Images 1(a), 2(a), 3(a) show high resolution texture patches of rough plaster surface,medium to smooth plaster surface and a directional plaster surface.Images 1(b), 2(b) and 3(b) are low resolution images of rectangular objects made of the above mentioned materials and 1(c), 2(c) and 3(c) represent the views generated from their corresponding **PTM**s .Images 1(d), 2(d) and 3(d) are the higher resolution views of the models .The final row of images 1(e), 2(e) and 3(e) are the ones obtained by zooming in 1(b), 2(b), 3(b) respectively.



(a) Object Views



(d) Our Results



(g) Unconstrained Synthesis Results

Figure 4.5 Constrained Synthesis vs Unconstrained Synthesis

The top row shows view of a planar surface, the middle row shows the corresponding views generated by our model and the last row shows the same generated by relighting the model obtained from unconstrained synthesis

Chapter 5

Image Constrained PTM Synthesis for Real world Objects

Patch-based **2D** Texture Synthesis algorithms [9, 19] use square patches as the quilting blocks because of their simplicity to work with. The same cannot be said about synthesis for real world objects which are **3D** in nature. The **3D** objects are usually represented with standard triangular mesh models.

Triangle is the basic primitive for rendering **3D** models. Hence its much more apt to consider triangle as the quilting block and texture map triangles rather than the usual square patches. But the triangles of the mesh model are of different sizes and shapes unlike the square patches used in the previous section which are all of uniform size. It becomes only difficult that all the triangles having various texture orientations.

Considering all the above mentioned issues, we devised an approach to synthesize image based **PTM** models of real world objects.Given a set of images $\{I_n\}$ of the object captured under known light and camera positions $\{(l_{un}, l_{vn}), C_n\}$ and a texture sample PTM_{in} of the object material, we synthesize a texture model of the object by pasting triangular subsamples taken from PTM_{in} all across the triangular mesh model of the object.Like the earlier approach for planar surfaces, this approach also considers the image-based constraints and overlapping constraints in selection of triangular patches for the texture model.

We outline the basic steps of our synthesis algorithm followed by a detailed description of each of them.

Algorithm 2: PTM Synthesis of Real world Objects

1. Assign each triangle T of the mesh model, an image $I_k \in \{I_n\}$ in which it is best visible and calculate its mapping t in I_k.;
2. Generate the normal view t_n from t, find its best matching triangular texture patch p in PTM_{in} and extract a rectangular patch B containing p.;
3. Perform Alpha-blending across every edge of the mesh model, update the texture patches $\{B_i\}$ with blended values.;
4. Extract the minimal bounding box b_i contained inside B_i of each triangular texture patch p_i, and pack all such b_i into a number of texture atlases.;

5.0.0.4 Step1

The object is imaged multiples times from different known light and view-point conditions $\{(l_{un}, l_{vn}), C_n\}$ to obtain a set of images $\{I_n\}$. Each triangle T of the mesh model is then mapped to an image $I_k \in \{I_n\}$ in which it is best visible. The images in which the triangle T is completely visible are picked and then an image I_k among them in which it is best visible is taken. The criteria for visibility is the angle made by the normal n of the triangle T with the directional vector of the camera C from its center. We rely on the assumption that each triangle T of the mesh model is completely visible in at least one image.The camera matrix M_k corresponding to the image I_k is calculated and then used to obtain the mapping t of triangle T in the image I_k.

5.0.0.5 Step2

After step 1, each T is mapped to its best-view t in an Image I_k. Now based on the angle θ between the normal n of T and the direction of the camera center C, the lengths of sides of the triangle in the normal view t_n are obtained using the following formula

$$l_{in} = l_i / \cos\theta, \tag{5.1}$$

where l_i is the length of ith side of t in the image I_k.

The geometry of t_n is is determined by these sides $\{l_{in}\}$ and angles A, B, C of the original triangle T. For simplicity, the side connecting the first 2 vertices of t_n is made parallel to X-axis. The color information from t to t_n is transfered using a re-sampling algorithm.

Calculate the local light vector l_T with respect to a co-ordinate system placed at the centroid of T. The X-axis of this co-ordinate system aligns with the side connecting the first two vertices of T, Z-axis along the normal of T and Y-axis decided by the former two. Evaluate the input sample PTM_{in} using

l_T and search the resultant image for a set of patches $\{t\prime\}$ which best agree with t_n. This constitutes the image based constraints.Each $t\prime$ corresponds to a triangular texture patch $p\prime$ in the input texture sample PTM_{in}.

Now pick the best texture patch $p \in \{p\prime\}$ which best agrees with the texture patches of already processed neighboring triangles $\{T_j\}$ of T. This constitutes the overlapping constraint imposed on the synthesis. In order to impose overlapping constraints, at least one of the three neighbors of the current triangle T should have been already processed. Hence random processing of triangles of the mesh model might result in occasional weakening of the selection strategy and the quality of texture model so obtained.

To prevent this, the triangles of the mesh model are processed in a Breadth-First-Search(**BFS**) order. By doing so, overlapping constraints are imposed in the selection of texture patch p for every triangle T of the mesh model except for the first one.

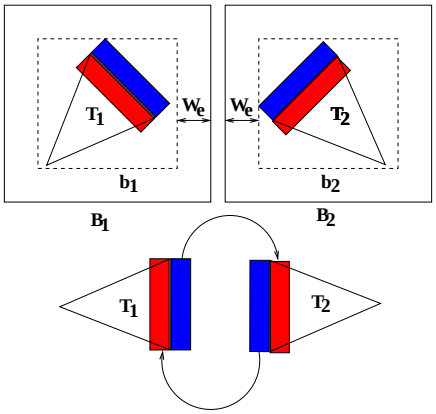


Figure 5.1 Blending of Neighboring Triangles Red region of T1 blends with blue region of T2 and vice versa

For each triangle T_i, a minimal bounding box b_i surround its triangular texture patch p_i is identified and a bigger rectangular patch B_i containing b_i surrounded by a extra texel strip(5 to 10 texels) all around is extracted from the texture sample PTM_{in}. The extra strip of texels is used for blending with texture patches of neighboring triangles.

5.0.0.6 Step3

Each of the above mentioned boxes B_i include an extra strip of W_e on all the 4 sides of the corresponding minimal bounding box b_i. This strip is essentially used for alpha-blending across edges.The extra texel padding around the actual triangular texture is blended with the border information of neighboring triangle as shown in the Figure 5.1. The alpha-blended information is written back to the set of boxes $\{B_i\}$

5.0.0.7 Step4

Minimal bounding boxes $\{b_i\}$ are extracted from $\{B_i\}$ by cutting off the extra strip of texels present around. These $\{b_i\}$ are then packed in to a number of atlas maps of desired dimensions W and H using any of the standard bin-packing algorithms. The texture mapping co-ordinates of all the triangles T_i are updated all along the procedure and the final mapping co-ordinates with respect to the PTM atlases $\{P_j\}$ are stored.

The above process of PTM synthesis for a real world object is an off-line process. Hence we limit ourselves to only the synthesis procedure and not wade in to the details of time complexity, techniques to speed it, etc.

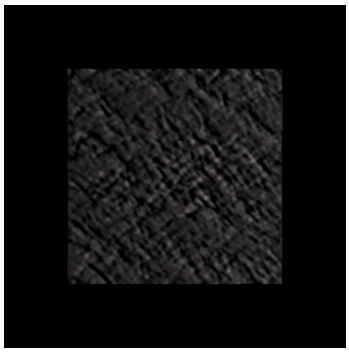
5.0.1 Rendering of the PTM Model

The **PTM** model of the object obtained can be efficiently rendered at run time to generate novel views of the real world object under different lighting conditions. In this procedure, each triangle T is considered separately and the position of light with respect to it calculated. The unit vector (lu_T, lv_T) so obtained is used to evaluate its PTM patch p to generate an image patch.Hence we get image atlases corresponding to the set of **PTM** atlases. These image atlases are used as texture objects, loaded and the textured model rendered.

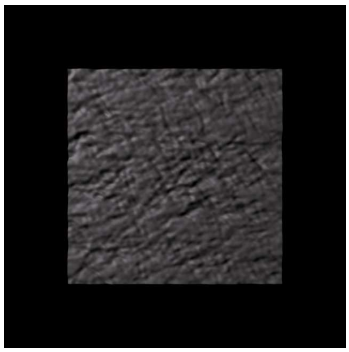
5.0.2 Experimental Results

We demonstrate our **3D** synthesis algorithm on a set of rough surface models created using displacement mapping. Synthetic **3D** textures and object models are used so that the same texture model can be used to generate surface textures for the mesh model. We generated a height map and applied it individually on the plane surface, a smooth sphere and a cylinder using displacement mapping to create rough objects. A sample **PTM** is then created using a set of images of the rough planar surface. A small set of images of the rough sphere and cylinder were taken to provide the image based constraints. These images and the sample **PTM** were used to construct the **PTM** models of the rough sphere and the cylinder. In Figure 5.2, the images (a), (b) and (c) show three views of a rough plain and these are used to construct the texture sample employed in synthesis. Images (d) and (g) show a rough sphere and a cylinder created using displacement mapping. Images (e) and (f) show two arbitrary views of the constructed **PTM** model of the rugged sphere and images (h) and (i) show the same for the rugged cylinder.

As we note, the synthesized model is able to capture the surface properties, as the lighting directions change, which would be impossible in the case of 2D textures. Moreover, as seen from synthesis of the planar object, the synthesized 3D texture generates images resemble the observed images of the original object. However, there are primarily two issues that still remain to achieve photo-realistic rendering of 3D mesh models: i) The PTM model itself does not handle shadows and specularities in the texture well as it creates an overly smooth approximation of the transition from light to shadows with change in light direction, ii) Variations in appearance with lighting direction is accentuated at the triangle boundaries. Currently we are working on developing improved models of the PTM to handle the first issue, and with synthesis techniques that directly create smooth transitions over triangle boundaries.



(a) View1 of a Rugged Planar Surface



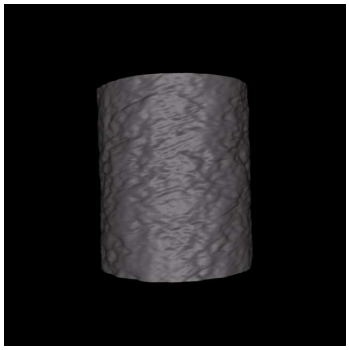
(b) View2

(c) View3

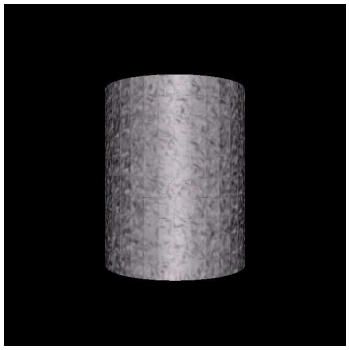
(d) Rugged Sphere

(e) View1 of Textured Spherical Mesh Model

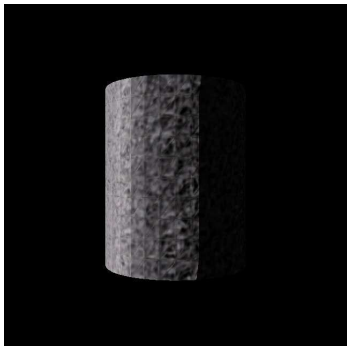
(f) View2



(g) Rugged Cylinder



(h) View1 of Textured Cylindrical Mesh Model



(i) View2

Figure 5.2 Synthesis Results for 3D object:Images (a), (b) and (c) show arbitrary views of rugged planar surface.Image(d) shows a rugged sphere and (e),(f) show two views of the texture model generated model generated.Likewise images (g),(e) and (f) show the same for a rugged Cylinder.

Chapter 6

Conclusions

We demonstrated an image based texture synthesis technique to effectively synthesize reflectance textures for material surfaces and objects. We developed the idea of transfering texture on to the mesh models of real world objects to realistically reproduce the natural visual appearance, perception and their interaction with the lighting environment. While the synthesis algorithm for planar surfaces is robust and efficient, the 3D synthesis algorithm offers challenges and scope for improvements both in synthesis and rendering aspects. The main challenges were the blending of reflectance functions across the edges of triangles which is a non-trivial task, the artifacts caused by lack of smoothness or continuity in directionality of texture across triangular patches and the inability to model the view point phenomena.

The synthesis algorithms that synthesize texture on mesh models [26, 31, 33] rely on specification or computation of surface tangential vector field across the surface of the mesh models. Traversing along the flow of the vector field ensures the orderly sweeping of points/polygons on the mesh model just like a scan-line order visit of pixels on an image ensures the same. The vector field not only determines the order of processing of triangles but also the scale and rotational parameters of the synthesized triangular patches if the material is isotropic in nature.

We have assumed the anisitropic nature of the synthesis material as our experiments were mostly confined to the surfaces like granite, concrete etc which lacked directionality. Contrary to the orderly processing of triangles suggested by the conventional texture synthesis algorithms mentioned above, we adopted a region-growing policy that starts with a randomly chosen polygon as the seed and then proceeds to its immediate neighbors and so on, growing a region outwards. This alone we believe ensures orderly processing of polygons so as to impose proper overlapping constraints on the patch selection procedure. However, this doesn't hold good in the case of iso-tropic textures which also have a directionality to their surface and surface tangential vector field cannot be ignored in such a scenario.

Inspite of texture blending that is applied across the border of triangular textures patches in the functional space, artifacts caused by dis-oriented texture patches of neighboring triangles is still a cause for

concern as the blending doesn't consider the dimensions of scale and orientation. We noticed some artifacts presicely caused by this reason even while synthesizing some anisotropic materials. Hence,we feel that introducing the step of estimation of surface tangential vector field holds the key for a significant improvement not only because it brings into consideration two more aspects of a texture, but also impacts the blending procedure by correcting for the scale and orientation when blending is applied across neighboring triangles.

Another important challenge is the task of blending reflectance function coefficients across every edge of the mesh models. In contrast to the alpha-blending that is applied in the case of simple color textures, the blending of reflectance functional coefficients is a tricky challenge. The conventional alpha-blending is applied in the color space where as our task requires an efficient blending technique that can blend multiple aspects of texture namely color,normal information, reflectance properties. Polynomial texture maps implicitly contain the surface normal information. Approximate surface normal information can be obtained at every texel from its functional coefficients. The goal is to obtain a blending mechanism that operates in the functional space so as to ensure smoothness of color, normal and reflectance across a triangle edge. We have employed alpha-blending directly on the coefficients in a transformed orthogonal space. From the results, we found that this alone won't suffice as this approach won't necessarily interpolate the normal information across the surface of the mesh model. Exploring the methodologies to smoothen the normals by operating in the functional space is a potential direction of work that we hope holds the key for enhancing the aspect of realism by reducing the scope for artifacts. Investigating the idea of manipulation of luminance coefficients in influencing the visual perception of shape information of the object surface is also an wider direction to carry out the present work.

As Polynomial Texture Maps do not model the view-point phenomena, our present scheme doesn't model the effects such as specularities. However, as mentioned above, the functional coefficients can be used to obtain approximate surface normal information. The so obtained normals can be used to artificially introduce view-point phenomena at the time of rendering. Introducing them will further enhance the realism and is definitely another potential direction of work.

One of the most critical piece of information that we have assumed all along is that the shape information of the object to be modeled is available as a triangulated mesh. In the case of large monuments and statues, obtaining this information is in itself another challenge. However, with a large set of images of the object, this process can be automated and made robust. It suffices even if these are low resolution images as the clarity of the model is enhance by maintain the textural attributes which is taken care of by our high-resolution synthesis mechanism.

Related Publications

Pradeep Rajiv and Anoop M.Namboodiri, “Image based PTM synthesis for realistic rendering of low resolution 3D models“, *in Proceedings of the Seventh Indian Conference on Computer Vision, Graphics and Image Processing*, Chennai, India, 2010

Bibliography

[1] C. H. Anderson, J. R. Bergen, P. J. Burt, and J. M. Ogden. Pyramid methods in image processing, 1984.

[2] M. Ashikhmin. Synthesizing natural textures. In *Proceedings of the 2001 symposium on Interactive 3D graphics*, I3D '01, pages 217–226, New York, NY, USA, 2001. ACM.

[3] M. Ashikhmin and P. Shirley. Steerable illumination textures. *ACM Trans. Graph.*, 21:1–19, January 2002.

[4] A. Baumberg. Blending images for texturing 3d models. In *Proc. Conf. on British Machine Vision Association*, pages 404–413, 2002.

[5] G. R. Cross and A. K. Jain. Markov random field texture models. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, PAMI-5(1):25 –39, 1983.

[6] K. J. Dana, B. van Ginneken, S. K. Nayar, and J. J. Koenderink. Reflectance and texture of real-world surfaces. *ACM Trans. Graph.*, 18:1–34, January 1999.

[7] 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*, SIGGRAPH '97, pages 361–368, New York, NY, USA, 1997. ACM Press/Addison-Wesley Publishing Co.

[8] P. Debevec, T. Hawkins, C. Tchou, H.-P. Duiker, W. Sarokin, and M. Sagar. Acquiring the reflectance field of a human face. In *Proceedings of the 27th annual conference on Computer graphics and interactive techniques*, SIGGRAPH '00, pages 145–156, New York, NY, USA, 2000. ACM Press/Addison-Wesley Publishing Co.

[9] 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*, SIGGRAPH '01, pages 341–346, New York, NY, USA, 2001. ACM.

[10] A. A. Efros and T. K. Leung. Texture synthesis by non-parametric sampling. In *Proceedings of the International Conference on Computer Vision-Volume 2 - Volume 2*, ICCV '99, Washington, DC, USA, 1999. IEEE Computer Society.

[11] S. Geman and D. Geman. *Stochastic relaxation, Gibbs distributions, and the Bayesian restoration of images*, pages 452–472. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 1990.

[12] P. Harrison. A non-hierarchical procedure for re-synthesis of complex textures. In *In WSCG ?2001 Conference proceedings*, pages 190–197, 2001.

[13] 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*, SIGGRAPH '95, pages 229–238, New York, NY, USA, 1995. ACM.

[14] Y. Hel-Or, T. Malzbender, and D. Gelb. Synthesis and rendering of 3d textures. *3rd International Workshop on Texture Analysis and Synthesis*, 2003.

[15] A. Hertzmann, C. E. Jacobs, N. Oliver, B. Curless, and D. H. Salesin. Image analogies. In *Proceedings of the 28th annual conference on Computer graphics and interactive techniques*, SIGGRAPH '01, pages 327–340, New York, NY, USA, 2001. ACM.

[16] B. Julesz. Visual pattern discrimination. *Information Theory, IRE Transactions on*, 8(2):84 –92, 1962.

[17] T. Leung and J. Malik. Representing and recognizing the visual appearance of materials using three-dimensional textons. *Int. J. Comput. Vision*, 43:29–44, June 2001.

[18] M. Levoy, K. Pulli, B. Curless, S. Rusinkiewicz, D. Koller, L. Pereira, M. Ginzton, S. Anderson, J. Davis, J. Ginsberg, J. Shade, and D. Fulk. The digital michelangelo project: 3d scanning of large statues. In *Proceedings of the 27th annual conference on Computer graphics and interactive techniques*, SIGGRAPH '00, pages 131–144, New York, NY, USA, 2000. ACM Press/Addison-Wesley Publishing Co.

[19] L. Liang, C. Liu, Y.-Q. Xu, B. Guo, and H.-Y. Shum. Real-time texture synthesis by patch-based sampling. *ACM Trans. Graph.*, 20:127–150, July 2001.

[20] X. Liu, Y. Yu, and H.-Y. Shum. Synthesizing bidirectional texture functions for real-world surfaces. In *Proceedings of the 28th annual conference on Computer graphics and interactive techniques*, SIGGRAPH '01, pages 97–106, New York, NY, USA, 2001. ACM.

[21] S. Magda and D. Kriegman. Fast texture synthesis on arbitrary meshes. In *Proceedings of the 14th Eurographics workshop on Rendering*, EGRW '03, pages 82–89, Aire-la-Ville, Switzerland, Switzerland, 2003. Eurographics Association.

[22] T. Malzbender, D. Gelb, and H. Wolters. Polynomial texture maps. In *Proceedings of the 28th annual conference on Computer graphics and interactive techniques*, SIGGRAPH '01, pages 519–528, New York, NY, USA, 2001. ACM.

[23] R. Paget and I. Longstaff. Texture synthesis via a noncausal nonparametric multiscale markov random field. *Image Processing, IEEE Transactions on*, 7(6):925 –931, jun 1998.

[24] K. Popat and R. W. Picard. Novel cluster-based probability model for texture synthesis, classification, and compression. In *In Visual Communications and Image Processing*, pages 756–768, 1993.

[25] J. Portilla and E. P. Simoncelli. A parametric texture model based on joint statistics of complex wavelet coefficients. *Int. J. Comput. Vision*, 40:49–70, October 2000.

[26] E. Praun, A. Finkelstein, and H. Hoppe. Lapped textures. In *Proceedings of the 27th annual conference on Computer graphics and interactive techniques*, SIGGRAPH '00, pages 465–470, New York, NY, USA, 2000. ACM Press/Addison-Wesley Publishing Co.

[27] C. Rocchini, P. Cignoni, C. Montani, and R. Scopigno. Multiple textures stitching and blending on 3d objects. In *Eurographics Rendering Workshop*, pages 119–130. Springer-Verlag, 1999.

[28] Soucy, Marc, G. Godin, and M. Rioux. A texture mapping approach for the compression of colored 3d triangulations. *The Visual Computer*, 12:503–514, 1996.

[29] X. Tong, J. Zhang, L. Liu, X. Wang, B. Guo, and H.-Y. Shum. Synthesis of bidirectional texture functions on arbitrary surfaces. In *Proceedings of the 29th annual conference on Computer graphics and interactive techniques*, SIGGRAPH '02, pages 665–672, New York, NY, USA, 2002. ACM.

[30] G. Turk. Generating textures on arbitrary surfaces using reaction-diffusion. *SIGGRAPH Comput. Graph.*, 25:289–298, July 1991.

[31] G. Turk. Texture synthesis on surfaces. In *Proceedings of the 28th annual conference on Computer graphics and interactive techniques*, SIGGRAPH '01, pages 347–354, New York, NY, USA, 2001. ACM.

[32] 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*, SIGGRAPH '00, pages 479–488, New York, NY, USA, 2000. ACM Press/Addison-Wesley Publishing Co.

[33] L.-Y. Wei and M. Levoy. Texture synthesis over arbitrary manifold surfaces. In *Proceedings of the 28th annual conference on Computer graphics and interactive techniques*, SIGGRAPH '01, pages 355–360, New York, NY, USA, 2001. ACM.

[34] Y. N. Wu, S. C. Zhu, and X. Liu. Equivalence of julesz ensembles and frame models. *Int. J. Comput. Vision*, 38:247–265, July 2000.

[35] H. S. Y. Xu, B. Guo. Chaos mosaic: Fast and memory efficient texture synthesis. Technical Report MSR-TR-2000-32, Microsoft Research, April 2000.

[36] S. C. Zhu, Y. Wu, and D. Mumford. Filters, random fields and maximum entropy (frame): Towards a unified theory for texture modeling. *Int. J. Comput. Vision*, 27:107–126, April 1998.