Improving Realism of 3D Texture using Component Based Modeling

Siddharth Kherada, Prateek Pandey, Anoop M. Namboodiri
Center For Visual Information Technology (CVIT)
International Institute of Information Technology Hyderabad, India
siddharth.kherada@research.iiit.ac.in, prateek@students.iiit.ac.in, anoop@iiit.ac.in

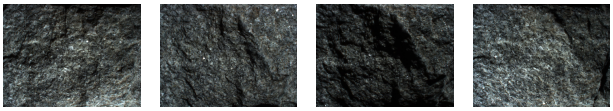
Abstract

3D textures are often described by parametric functions for each pixel, that models the variation in its appearance with respect to varying lighting direction. However, parametric models such as Polynomial Texture Maps (PTMs) tend to smoothen the changes in appearance. We propose a technique to effectively model natural material surfaces and their interactions with changing light conditions. We show that the direct and global components of the image have different nature, and when modeled separately, leads to a more accurate and compact model of the 3D surface texture. For a given lighting position, both components are computed separately and combined to render a new image. This method models sharp shadows and specularities, while preserving the structural relief and surface color. Thus rendered image have enhanced photorealism as compared to images rendered by existing single pixel models such as PTMs.

1. Introduction

Texture refers to a surface characteristic and appearance of an object given by its geometry, density and surface reflectance, and the stochastic variation of these parameters. It is an important cue in trying to achieve photorealistic rendering of 3D models by adding surface details or color to an object or a scene.

Mapping 2D textures or images is the most common method used, which is efficient for most 3D models and scenes, especially where the lighting conditions remain constant. They look best when the object is viewed in similar lighting conditions as when the texture is captured. In practice, the real world surfaces are characterized by phenomena such as inter-reflection, self-shadowing, subsurface scattering, specularity, etc. These properties interact with different lighting directions and therefore the same surface appears different under different lighting condition Fig.1. 2D texture fails to capture these complex reflectance properties of a surface and therefore a rendered surface looks highly unrealistic in case the lighting conditions are changed. In order to produce a realistic rendering it is necessary to capture and model the interaction of the material surface with different lighting conditions. [6] investigates the problem of representation, recognition, synthesis of natural materials and their rendering under arbitrary viewing/lighting conditions.



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

Figure 1. Images of a rough granite surface obtained under varying light conditions. Note the change in surface appearance in each of (a), (b), (c) and (d)

3D textures are a way to model this relation between surface reflectance properties and illumination/viewing conditions. The use of 3D texture modeling results in enhanced realism of the scene. Reflectance texture maps are one of the techniques that can be used to compactly represent the 3D textures. These maps are generated using image relighting techniques [1, 8, 3] in which multiple images are captured under different lighting conditions.

Image based modeling techniques [2, 11, 12] 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. The traditional object models capture the shape information in the meshes, while the reflectance and the surface properties are relegated in the textures. 3D models such as PTM capture the surface properties more faithfully, including the effect of small scale height variation on the surface.

Polynomial Texture Maps [8] belong to the class of UTFs (Uni-directional Texture Function). It is a pixel based technique that concisely models the surface reflectance properties using a polynomial model for the reflectance, dependent on two angular parameters of the lighting direction (l_u and l_v). It uses a biquadratic polynomial function with 6 coefficients per pixel for modeling the reflectance. PTMs

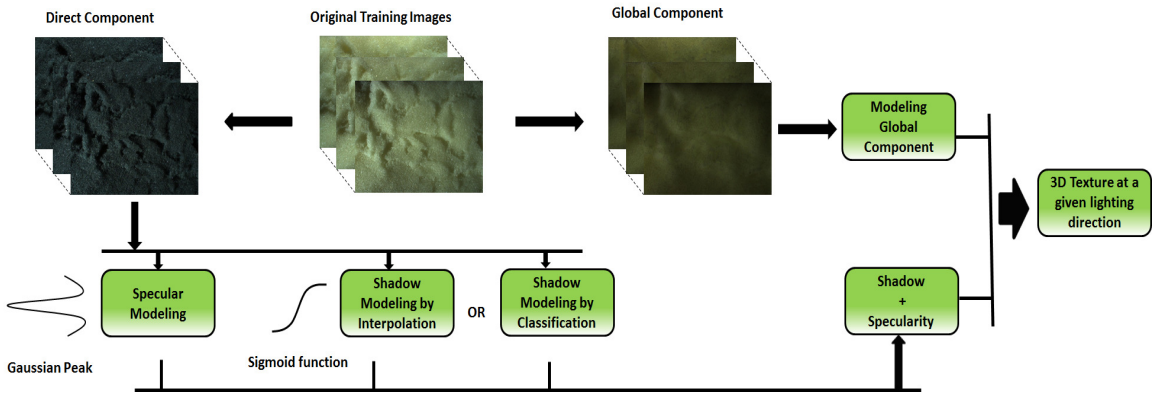


Figure 2. Component Based Modeling (CBM)

reconstruct the color of the surface under varying lighting conditions and models real world phenomenon such as self-shadowing, inter-reflection and sub-surface scattering. They thus introduce enhanced photorealism in texture mapping. Polynomial Texture Mapping is applied in a wide range of archaeological contexts [5]. It also offers advantages over traditional raking light photography for examining and documenting the surface texture and shape of paintings [10]. Recently, PTMs have been used in cultural heritage domain for documenting and virtually inspecting several sets of small objects, such as cuneiform tablets and coins.

But PTM technique causes overall smoothening of light which dampens the effect of specularity and softens sharp shadows. The effect of point light source is reduced and the appearance is always similar to a diffused light source. The current state-of-the art in the field of PTM involves robust method for interpolation of shadow and specularity, Drew *et al*[4]. But they do not model natural material surfaces and their interactions with changing light conditions. Moreover the number of per pixel parameters are too high for real-time rendering. [7] shows that surface normals extracted from the PTM image data structure are of lower quality because of the smoothing caused by the use of biquadratic function, which compromises the directional accuracy of the normals.

In this paper we improve upon the PTM model to overcome the above limitations and generate a complete 3D Texture model that can be evaluated at individual pixels. We propose an approach to image-based lighting interpolation that is based on estimates of geometry and shading from a set of input images. We decompose images captured at different lighting conditions into intrinsic image components; i.e, the *direct* and *global* image components. Each of these components is then further separated to obtain different physical phenomena such as shadows, specularity and luminance. The final image is obtained by combining the individual models together. The method is shown to indeed generate better results for non-observed lighting directions. The main contributions are (1) Direct and Global modeling characterized by shadows, specularity and luminance, (2) separate modeling and hence better capturing of shadows and specularities and (3) per pixel function model to achieve real-time rendering of enhanced 3D textures on GPU.

2. Component Based Modeling (CBM)

The appearance of the texture in a given lighting condition is characterized by shadows, specularity and overall luminance. The luminance is affected by subsurface scattering and inter-reflection properties of the surface. PTM does not separately take these properties into account and models them together using a biquadratic function. However, the nature of variation of the reflected light is significantly different for these phenomena.

In our method, we analyze each of these phenomena separately and capture the results using appropriate models. We first separate the images into two components: One is the *direct* part, which captures the light that is directly reflected by the surface point from the source and other is the *global* part which is due to the illumination of the point from all other points of the scene. Separation of a scene into global and direct part can be done by illuminating the scene with a high frequency binary pattern [9]. Fig. 3 shows the direct and global component of sponge texture.

The shadows and specularities appear very strongly in the direct part, as these are phenomena that involve light that reaches the surface point directly from the light source. The fine details and structure of the material are also very prominently visible in the direct part. The global component contains the lighting of a surface point from other parts in a scene, and hence captures the overall illumination as well as color variations of a surface with lighting direction.

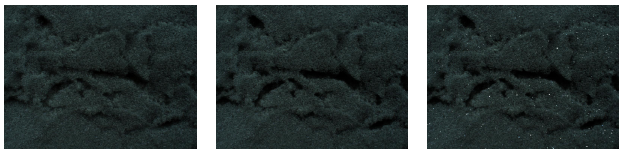
Both direct and global components are separately analyzed to derive the corresponding models and parameters. Given a new lighting direction, we use the two models separately to generate the corresponding components, and combine them to get the final image. Fig. 2 shows the overall

(a) (b) (c)

Figure 3. Components of a sponge image: a) Original image, b) Direct c) Global component.

(a) l_u = .61, l_v = .35 (b) Interpolated (c) l_u = -.61, l_v = -.35

Figure 4. Shadow interpolation in two directions: a,c) images with horizontally varying lighting directions, b) interpolated direct image between the two



(a) (b) (c)

Figure 5. a) Direct component of an image computed using bilinear interpolation, b) after multiplying (a) by the shadow mask, and c) after adding specularity.

component based modeling technique.

2.1. Modeling Direct Component

As noted before, the direct component is affected by the phenomena of self-shadowing and specularity, in addition to the lambertian reflectance of the surface point. Shadows are the points that receive no direct light from the source. However, their luminance value is not completely zero. This is because they get some light from the neighboring pixels because of inter-reflections.

When the image is decomposed into direct and global component, the luminance value of shadow region (due to inter-reflections) appear in global part and thus direct part is left with dark prominent shadow regions whose value is near to zero (Fig. 3(b)). These dark shadow regions can easily be separated out using thresholding which is computationally more efficient.

2.1.1 Shadow Modeling by Interpolation

Consider a pair of images of a surface captured from the same view point, but by moving the light source through a short distance. Each pixel belong to one of the three categories:**a)** Pixels that are not in shadow in either image. **b)** Pixels that are in shadow in both images. **c)** Pixels that are in shadow in only one of the images.

For the first two categories, the pixels that are in shadow continue to remain in shadow and that illuminated remain illuminated. The luminance values of the first two types of pixels can be directly calculated by linear interpolation or higher order interpolation between the values of the corresponding pixels in the given two images. In practice, linear interpolation works well, as the variations are often very limited for the first two types. The value of the interpolated pixel(k), L, at lighting position p_0 is given by:

$$L(k, p_0) = \frac{\omega_2 L_1(k, p_1) + \omega_1 L_2(k, p_2)}{\omega_1 + \omega_2}, \qquad (1)$$

where $\omega_i = |D(p_0, p_i)|$; $D(p_a, p_b)$ gives distance between lighting directions at p_a and p_b.

In case of a pixel that transitions from shadow to light (or the reverse), the transition is quick, though not instantaneous. We model this behavior using a sigmoid function. As the light source moves from the position of the first image (p_1) to the second (p_2), there is a point p_x around which the pixel quickly emerges out of the shadow and then remains illuminated for the rest of the light motion. The transition would be abrupt except for the diffraction of light around the edge causing the shadow. Given the illuminations of shadow (L_s) and non shadow (L_{ns}) pixels, and position p_x at which the transition occurs, the illumination at position p_0 can be approximated by a sigmoid of the form:

$$L(k, p_0) = L_s(k, p_1) + \frac{L_{ns}(k, p_2) - L_s(k, p_1)}{1 + \chi e^{-d}}, \qquad (2)$$

where $d = p_0 - p_x$. The slope of the sigmoid function controls the transient behavior of pixels from shadow to non-shadow region, controlled by the parameter χ.

The only unknown in carrying out the interpolation is the position p_x at which the transition occurs. Consider a pixel k that is in shadow at light position p_1. Let χ_s be the fraction of neighboring pixels of k that are in shadow in the first image, and χ_{ns} be the fraction of neighboring pixels of k that are not in shadow in the second image. We compute these fractions by taking masks of increasing sizes until the $0 < \chi_x < 1$. If χ_{ns} and χ_s are almost equal, then the transition, p_x occurs around midway between positions p_1 and p_2. If $\chi_{ns} >> \chi_s$, then p_x is close to p_1, and $\chi_s >> \chi_{ns}$ indicates that p_x is far from p_1 and close to p_2. We define p_x as:

$$p_x = \frac{\chi_{ns} p_1 + \chi_s p_2}{\chi_{ns} + \chi_s} \qquad (3)$$

The advantage of interpolation is that the physical structure of the material is taken into account while interpolating, leading to realistic estimations of shadows. This is implicitly used while considering the neighborhood information

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

Figure 6. a) Binarized image of cloth shadow, b) Image by classification, c) Image by interpolation, d) Distance image of pixels from classifier boundary. Blue pixels are closest to the hyperplane and include pixels at the edge of a shadow or in the region of diffused shadow. Black color pixels are the farthest from the hyperplane and represent regions of dense shadow.

of a pixel. However approach is both memory and compute intensive as one need to store input images for interpolation, and the computation of each pixel of the shadow mask involves searching an increasing neighborhood of pixels. An alternate method is to decide whether a given pixels falls in shadow or not, independently as a function of just the lighting position. Fig.4 shows the input images and the interpolated image at two different lighting positions.

2.1.2 Shadow Modeling by Classification

In our experiments, we note that most pixels fall under shadow from the effect of at most two neighboring structures. Hence, a biquadratic classifier boundary is adequate to decide whether for a given lighting direction, the pixel will be in shadow or not.

$$Ya = b \tag{4}$$

$$\underbrace{\begin{bmatrix} {y_1}^{(0)} & {y_1}^{(1)} & \ldots\ldots & {y_1}^{(5)} \\ {y_2}^{(0)} & {y_2}^{(1)} & \ldots\ldots & {y_2}^{(5)} \\ \vdots & & & \vdots \\ {y_n}^{(0)} & {y_n}^{(1)} & \ldots\ldots & {y_n}^{(5)} \end{bmatrix}}_{Y} \underbrace{\begin{bmatrix} a_0 \\ a_1 \\ \vdots \\ a_5 \end{bmatrix}}_{a} = \underbrace{\begin{bmatrix} b_1 \\ b_2 \\ \vdots \\ b_n \end{bmatrix}}_{b} \tag{5}$$

where y_i=[${l_u}^2$ ${l_v}^2$ $l_u l_v$ l_u l_v 1], a is the separating hyperplane, b is the margin vector and 'n' is the total number of training images.

The direct component of input images are binarized using thresholding and used as training data. After the classification, each pixel in new image is labeled as shadow or non-shadow. The resulting shadow regions estimated by classifier are very close to original and more accurate than the images that are directly interpolated from the input images Fig. 5(a)-(c).

We use the binary image, obtained after classification, to make a mask where each non-shadow pixel is given a value of 1 and shadow pixels are given values between 0 to 1 based on their distance from the hyperplane. The greater the distance, the farther the pixel is in shadow and thus smaller is the value (Fig. 6). Since the direct component is devoid of color variation, the change in chrominance value is minimal. Thus using a bilinear function

$$L(l_u, l_v) = \alpha l_u + \beta l_v + \gamma, \tag{6}$$

an interpolated image is generated (Fig. 5(a)). This image is then multiplied with shadow mask.

Using the above function for interpolation leads to the smoothening of shadows. Therefore we multiply this image with a mask described above to get the shadowed new image (Fig. 5(b)). The value of non-shadow pixels are not affected but the values of the shadow pixels are attenuated by the multiplication with the shadow mask. The classification technique thus enables us to render each pixel independently, increasing the speed of rendering and making it suitable for processing on the GPUs. The pseudo code for shadow modeling by classification is given in Algorithm 1.

Algorithm 1 Shadow modeling by classification

Input: Binarized direct component training images

1: Take shadow pixels as negative samples and non-shadow pixels as positive samples.
2: Learn the hyper plane (Ya=b) per pixel using pseudo inverse technique.
3: Classify pixels for a given lighting direction

$$Img(i,j) = \begin{cases} 1 & \text{if } Ya > 0 \\ 0 & \text{otherwise} \end{cases}$$

Output: shadow mask image

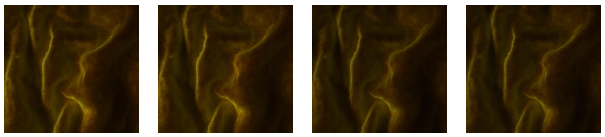
2.1.3 Modeling the Specularity

Specularity is the visible appearance of specular reflections. It determines the brightness and location of specular highlights, given a lighting direction. In case of PTMs, the ability to model the specularity is sacrificed due to fixed viewing direction. PTMs use a biquadratic interpolation model due to which the intermittent highlights, inherently present in many texture surfaces, are completely washed out.

We model the specular highlights separately from the base reflection and shadowing in the direct component. The value of pixels showing these highlights fall off very sharply as lighting direction is changed. One could use any sharp falling function such as a gaussian with very small variance or an exponential to model it. We model specular highlights, S, as:

$$S = \eta.exp - \left[\frac{(l_u - \mu_x)^2 + (l_v - \mu_y)^2}{\delta}\right], \tag{7}$$

where μ_x and μ_y are the lighting direction coordinates at which specularity is maximum, l_u and l_v are the current lighting directions. η and δ are the parameters that control



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

Figure 7. (a) Original Global Image (b) Global Image modeled by Gaussian function (c) Global Image modeled by Biquadratic (d) Global Image modeled by Parabola.

the magnitude and fall-off of this function. The highlights also can have a tint based on the nature of reflection. In this case, one can multiply the above function with a single chrominance value to achieve realistic estimation.

We note that the modeling of highlights is tricky as one can observe highlights only if one of the original images contain it. Hence the highlights estimated are often inaccurate, although realistic (Fig. 10(d),(e)). Fig. 5(c) shows the final image after multiplying the bilinearly interpolated image with shadow mask and adding specularity.

2.2. Modeling Global Component

The global component of the image is characterized by subsurface scattering, secondary illumination, diffuse interreflections, volumetric scattering and translucency. These are not sharply varying phenomena and therefore the variation of luminance can be modeled using an appropriate function. However, the inherent interaction between different parts of the surface in global illumination means that the chrominance of a point can change with a change in lighting direction.

As we separate the modeling of global component, the color values of the image rendered are closer in value to the original image and better than the images generated by PTM. From our experiments on various surfaces, the global component of illumination tends to be maximal when the illumination is perpendicular to the surface, and drops off in a symmetric fashion. We experiment with the following functions for modeling the global component: (a) Gaussian (b) Biquadratic polynomial, and (c) Paraboloid.

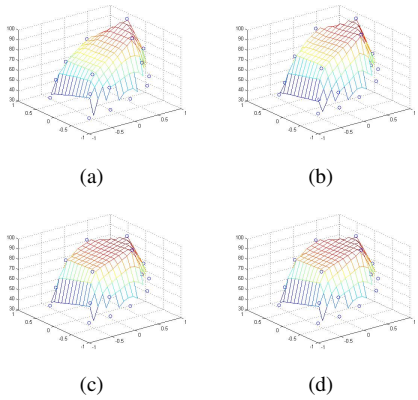
In (a), we model the luminance as a gaussian function of lighting direction:

$$L(l_u, l_v) = K \exp -(al_u^2 + bl_v^2 + cl_u l_v + dl_u + el_v + f) \quad (8)$$

The above system of equations can be solved using SVD and the coefficients a, b, c, d, e, and k, can be estimated per pixel. Biquadratic polynomial, also used in modeling PTMs [8], can be a good choice here because of the absence of sharply varying features.

$$L(l_u, l_v) = al_u^2 + bl_v^2 + cl_u l_v + dl_u + el_v + f \quad (9)$$

The paraboloid may not be as accurate as above functions and can lead to some smoothening but they are computationally efficient with 5 coefficients per pixel.



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

Figure 8. Comparison of luminance at a pixel as modeled by different functions: a) Original function plot at that pixel b) By Gaussian c) By Biquadratic d) By Parabola.

$$L(l_u, l_v) = al_u^2 + bl_v^2 + cl_u + dl_v + e \quad (10)$$

Fig. 7(a)-(d) shows the global component as modeled by each of the above functions. We see that the gaussian model provides the most accurate estimation of global component although all three models are similar in performance to visual inspection. One could hence use the paraboloid for purposes of efficiency and storage.

Fig. 8(a)-(d) shows the luminance of this pixel as a function of lighting direction when modeled using the different functions mentioned above. Gaussian provides the most accurate estimation especially at the peak values. The mean squared error over a sampled set of points from different surfaces is shown for comparison in table 1. Statistic ally and visually, the Gaussian model best fits the observations.

Table 1. Root Mean Square Error Comparison

Dataset	Gaussian	Biquadratic	Parabolic
Sponge	2.70	3.64	3.26
Cloth	3.60	6.01	4.41
Granite	2.81	3.99	3.25
Sand	2.66	4.09	3.97

3. Data Acquisition

We collect multiple images of a static object with a static camera under varying lighting conditions. The camera is mounted vertically above a table that holds the surface. Since the camera is fixed, we avoid the need for any camera caliberation. The scene is illuminated using a high frequency checkerboard pattern with the help of the projector. The projector (light source) is moved to different lighting positions for the purpose of obtaining images with different lighting directions. The distance of the projector from

the scene remains fixed, and only its height and position is changed. This enables us to capture multiple images with varying light source direction from a hemispherical set of world coordinates. We capture images from 30 different lighting directions and for each lighting direction, using component separation technique described in [9], we separate the image into its global and direct components. We capture 5-6 additional images which are used as benchmark images for comparing results.

4. Experimental Results and Analysis

The component based modeling proposed in this paper has been applied on various natural material textures. We present qualitative and quantitative assessment on texture images as obtained from our method and that obtained from the PTM over different natural material surfaces. In Fig. 10(a)-(l) [1], we compare ground truth images with images obtained from our technique and with PTMs. The texture of sand when modeled using CBM technique, preserves prominent shadow regions where as these regions are significantly washed out in PTM images(Fig. 10(a)-(c)) The sponge texture (Fig. 10(d)-(f)) shows a very noticeable difference between the two techniques.

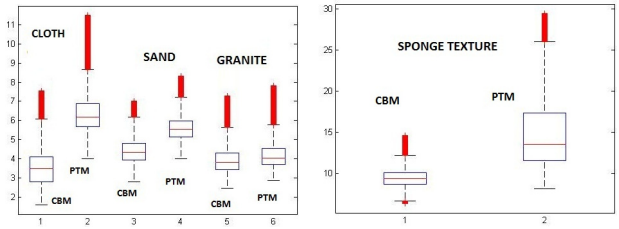
In the PTM images, there are no sharp shadows, the specularities are washed out and surface relief is smoothened to some extent whereas in CBM, structural details are preserved making it look more photorealistic. PTM smoothens sharp shadows, while they are preserved in the CBM rendered images.

For quantitative comparison, we capture additional images from known lighting directions during the capture phase. Generic measures such as PSNR only gives the average differences, and are not visually significant. We compute the absolute differences between each pixel values and analyze the distribution of these values. The differences between the original image and the image rendered using CBM and PTM are plotted as boxplot(Fig. 9).

It is clear from the figure that the average per pixel error and the number of outlier points are less in the image rendered using CBM as compared to those rendered using PTM. However, one should note that the PTM based models miss the specularities completely, while CBM is possibly rendering some of the specularities at incorrect positions. This would result in a higher quantitative error for CBM, while the visual appearance is improved.

In case of cloth texture (Fig. 9(a)-1,2), one can observe that the number of outliers are quite high in case of PTM. This is because fine details and the sharpness of edges in shadow region are lost in PTM. The root mean square error in case of CBM is 3.5 whereas in case of PTM its 6.2. If outliers are included then rms becomes 4.9 for CBM and

[1] View images in soft copy for better clarity



(a) (b)

Figure 9. Error comparison between CBM and PTM over different surface textures. Red bars indicate outliers. The red line in the box is the mean and the blue lines are the 25th and 75th percentile.

shoots upto 10.1 for PTM.

For white granite (Fig. 9(a)-5,6), average error is nearly same for both. PTM is able to render it well as the surface has less shadow and structural variations. However, if we consider sponge texture, the PTM performs quite badly with average error of around 14 and 75th percentile at 17 whereas average error of CBM is around 8 with 75th percentile at 10(Fig. 9(b)). Sponge is a highly textured surface with specularity and prominent shadows. PTM produces bad results as it tends to smoothen out the surface relief. But CBM accurately captures all structural details and thus rendered image is closer to the original. Modeling the shadows and specularity separately in CBM also allows us to make rendered images with multiple light sources, more realistic.

But this improvement is achieved at the cost of more coefficients per pixel as compared to 6 coefficients in PTM. CBM uses a total of 19 coefficients as compared to [4] which requires the estimation of parameters- α (a scalar), β (3-vector) and γ (n-vector). These parameters are estimated per pixel, separately for shadow and specularity modeling. Since 'n' is generally 40-50, therefore total coefficients are very high as compared to ours.

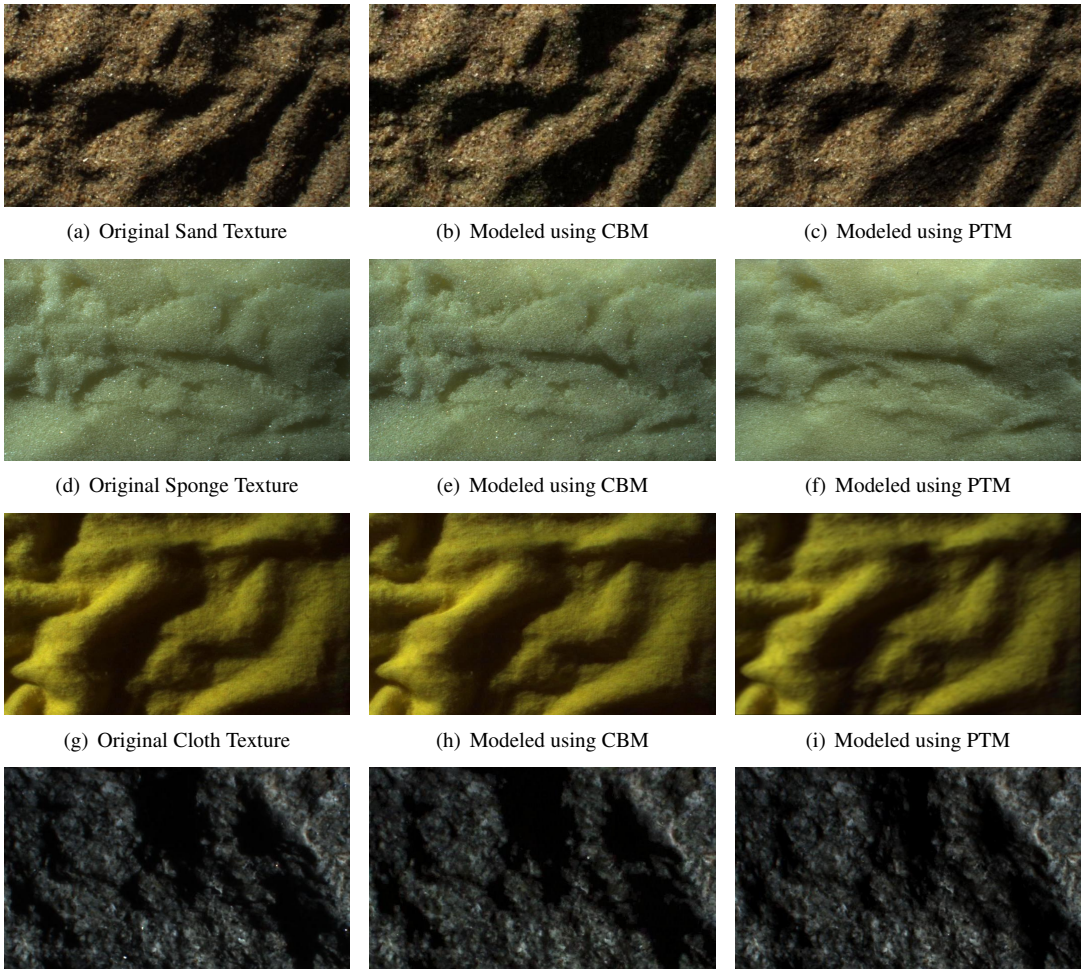
5. Conclusion and Future Work

This paper presents the technique which results in enhanced photorealism, preserving sharp shadows and specular properties from smoothening out and making point source effect more prominent. The separate model for luminance estimation provides us with color values which are in close agreement with the color values of the original image. Results obtained on re-rendering the input images show a considerable improvement over original PTM technique.

In future, we will port the rendering algorithms on a GPU to achieve real-time rendering with CBM textures. We aim at extending our algorithm to support synthesis of reflectance function texture.

References

[1] M. Ashikhmin and P. Shirley. Steerable illumination textures. *ACM Transactions on Graphics*, 21:1–19, 2002.



(a) Original Sand Texture (b) Modeled using CBM (c) Modeled using PTM

(d) Original Sponge Texture (e) Modeled using CBM (f) Modeled using PTM

(g) Original Cloth Texture (h) Modeled using CBM (i) Modeled using PTM

(j) Zoomed part of White Granite (k) Modeled using CBM (l) Modeled using PTM

Figure 10. Comparison of rendering results from Component Based Modeling and PTM techniques. CBM images have sharp shadows and specularity and also preserve the appearance of surface relief.

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

[3] K. Dana, B. V. Ginneken, S. Nayar, and J. Koenderink. Reflectance and texture of real-world surfaces. *ACM Transactions on Graphics*, 18(1):1–34, 1999.

[4] M. Drew, N. Hajari, Y. Hel-Or, and T. Malzbender. Specularity and shadow interpolation via robust polynomial texture maps. *In Proceedings of the British Machine Vision Conference*, 2009.

[5] G. Earl, K. Martinez, and T. Malzbender. Archaeological applications of polynomial texture mapping: Analysis, conservation and representation. *Journal of Archaeological Science*, pages 2040–2050, 2010.

[6] T. Leung and J. Malik. Representing and recognizing the visual appearance of materials using three-dimensional textons. *International Journal On Computer Vision*, 43(1):29–44, 2001.

[7] L. MacDonald and S. Robson. Polynomial texture mapping and 3d representations. *International Archives of Photogrammetry, Remote Sensing and Spatial Information Sciences*, 38, 2010.

[8] T. Malzbender, D. Gelb, and H. Wolters. Polynomial texture maps. *In Computer Graphics, SIGGRAPH Proceedings*, pages 519–528, 2001.

[9] S. Nayar, G. Krishnan, M. Grossberg, and R. Raskar. Fast separation of direct and global components of a scene using high frequency illumination. *Proceedings of ACM SIGGRAPH*, 2006.

[10] J. Padfield, D. Saunders, and T. Malzbender. Polynomial texture mapping: A new tool for examining the surface of paintings. *ICOM Committee for Conservation*, 1:504–510, 2005.

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

[12] M. Soucy, G. Godin, R. Baribeau, F. Blais, and M. Rioux. Sensons and algorithms for the construction of digital 3d colour models of real objects. *Image Processing Proceedings*, pages 409–412, 1996.