A Three-tier Hierarchical Model for Capturing and Rendering of 3D Geometry and Appearance from 2D Images

Martin Jagersand and Neil Birkbeck and Dana Cobzas
CS, University of Alberta, Canada
{jag,birkbeck,dana}@cs.ualberta.ca

Abstract

We propose a three-scale hierarchical representation of scenes and objects and show how this representation is suitable for both computer vision capture of models from images and efficient photo-realistic graphics rendering. The model consists of: (1) a conventional triangulated geometry on the macro-scale; (2) a displacement map, introducing pixel-wise depth with respect to each planar model facet (triangle) on the meso level; (3) a photo-realistic micro-structure represented by an appearance basis spanning viewpoint variation in texture space. We implement a capture and rendering system for this model. Conventional Shape-From-Silhouette or Structure-From-Motion is used to capture the coarse macro geometry, variational shape and reflectance estimation for the meso-level, and texture basis optimization for the micro level. For efficiency the meso and micro level routines are both HW accelerated. Photo-realistic capture of complex scenes is thus possible in a few minutes using budget cameras and PC's, and rendering is real-time. Experimental results and videos show models from regular images of humans and objects.

1. Introduction

Capturing scene and object models from images is one of the main areas in computer vision. Much of the research has focused on either geometry, as in multi-view stereo methods[26], or appearance, as in image-based modeling[27]. Some applications focus on one or the other, e.g., robotics needs geometry, 3D TV is primarily concerned with accurate appearance, while in many applications both geometry and photorealism is desired. In this paper the focus is on capture and representation of models suitable for use in conventional modeling software such as Maya and Blender. A useful application is to be able to capture natural objects for synthesis into new scenes in e.g. virtual heritage, interior design, or animation in computer games.

In the past few years a variety of rich BRDF representations for appearance have been proposed[6, 36, 21], and elaborate systems to capture and cluster BRDF material properties exist[19, 15]. However, a problem with these works are that they either study only flat material samples or assume an accurate (usually a-priori) geometry. They do not take into account the effects of geometric errors in the BRDF estimation. However, almost all out-of-the lab appearance modeling from images will have to deal with significant errors. Rather than high budget, high end lab or studio applications, we seek to push modeling from images downmarket, so reasonably photo-realistic results can be obtained with a home PC and consumer camera (from web cams to regular digital cameras). This would be an enabling technique for e.g. bringing everyday objects and scenes into virtual spaces.

While there has been lots of research on methods and details of 3D modeling from images, there seems to be preciously few system solutions available. KU Leuven's 3Dwebservice[34] allows users to submit images, runs Structure-From-Motion (SFM) and multiview stereo on a cluster of CPU's, then sends back geometry and cameras. However, much manual post processing is needed to achieve usable graphics models. Much less computationally demanding systems are based on Shape-From-Silhouette (SFS), e.g. [31, 22, 37]. A similar system but with more limited texturing was commercialized in 2005 by `www.3dsom.com`. Here we use both SFM and SFS for a coarse *macro* geometry recovery, but we further refine the geometry and add reflectance/appearance processing.

A central thesis in our method is that by employing a multi-tier representation, precision requirements on each level can be relaxed. Multi scale methods have been employed before in both vision and graphics. Perhaps the most common use is to accelerate rendering. Becker and Max [2] smoothly switch between three types of representations (BRDF, bump map, displacement map) as required by the amount of visible surface detail. Sloan et al. [29] presents an efficient two layer model. A main difference in our approach is that we capture and retain representations of all three levels in the model and use them simultaneously in rendering to generate the final images. Similarly, two-level models are common in image-based rendering (IBR). In the "Unstructured Lumigraph"[4] the lightfield is parametrized on a geometry proxy. This is much more efficient than on e.g. a box as in the original Lumigraph.

We present a three-tier model (Fig. 1) with a heterogeneous representation, such that each level adds detail (geometric or appearance) to the previous levels. In particular, we define representations on the following three scales:

- The *macro* scale describes a whole scene or object using a coarse, conventional (triangulated) geometry.
- The *meso* scale level represents geometric surface de-

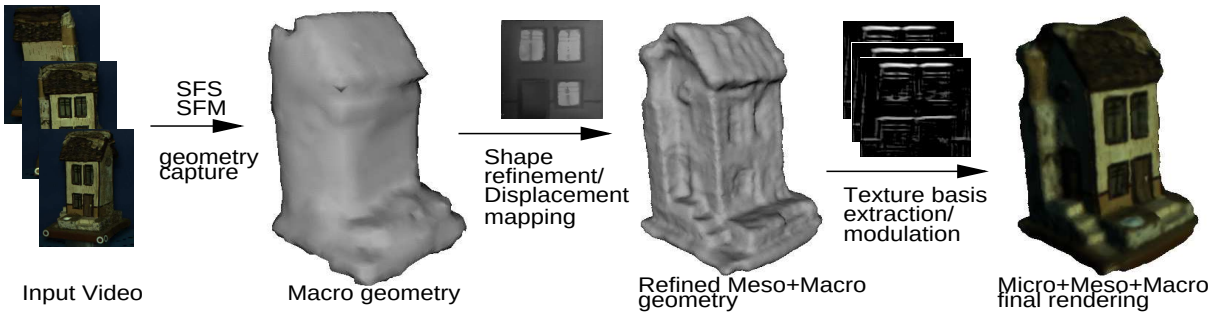

Figure 1. Computation of our *macro*, *meso*, *micro* representation through stepwise refinement, Video 1 and 2[1]

tails, generally at a visible scale (e.g. one to several pixels) through a displacement map.

- The *micro* scale level captures fine scale both geometric and reflectance properties (subpixel and up to few pixels). We use a texture basis similar to the BTF, but our basis is captured off the object and therefore also represents correct parallax, silhouettes and interreflections.

2. Capturing and rendering the 3-tiered model

2.1. Macro level: Coarsely Triangulated Geometry

Classic photogrammetry recovers 3D from 2D point correspondences using calibrated cameras. In the past decade much work was devoted to 3D recovery from uncalibrated images[16]. Despite this, no system can recover accurate dense geometry robustly and reliably from general scenes. One of the few publicly accessible systems is KU Leuven's 3Dwebservice[34], for which one can upload image sets of scenes and get back 3D reconstructions. It sequences SFM, auto-calibration, and dense stereo. The procedure is computationally demanding and running in parallel on a computer network, reconstructions often take hours to complete. Practically care must be taken in selecting both scenes and viewpoints for the system to work well. Nonetheless it is a representative of the state of the art in SFM based systems.

Shape-from-silhouettes (SFS)[28], on the other hand is a very robust method to obtain a visual hull geometry. It only requires the object silhouette and the calibration of the cameras. Besides it is quite robust to silhouette or calibration errors. In our system we implement an efficient algorithm for silhouette carving using an orthogonal ray set and Marching Intersections[32] algorithm. This decreases storage cost, and improves geometric precision (by recording silhouette intersections exactly on the rays) compared of the conventional discrete voxel representation. Computation time is less than one minute for all cases shown in this paper. Given the *macro* geometry either computed from SFS or imported from SFM we can proceed to the next level.

2.2. Meso level: Depth from Macro Geometry

Recently, state of the art computer vision surface reconstruction techniques have proved successful to refine the above models. Most assume simple reflectance (image constancy) and focus on obtaining an accurate geometry given a precisely calibrated setup (e.g. [24, 14], many more are surveyed in[26]). Few works consider general reflectance models [30, 38, 39]. We use a surface evolution method to compute the detailed *meso* geometry and simultaneously estimate a reflectance model.

Assuming that we have a set of images, I_i, the corresponding camera matrices, $\mathbf{P}_i$, and a low-res input geometry obtained from either SFM or SFS, the meso-scale structure can be obtained by finding the geometry that best matches the input images. We use a similar variational formulation as Faugeras & Keriven [11], where our goal is to find a surface, S, that minimizes the following functional:

$$F(S) = \int_S f(\mathbf{X}, \mathbf{N}) dA \tag{1}$$

where $\mathbf{X}$ is a point on the surface, $\mathbf{N}$ is the unit surface normal at that point, and dA indicates integration over the surface of the object. The function f measures the photo-consistency between the surface S and the input images. One common consistency measure is computed as the standard deviation of the image colors that a surface point projects onto. This measure assumes constant lighting and a perfectly matte surface. As we are interested in capturing the geometry of non-Lambertian objects, we instead use a more general form of f that measure the consistency of the image observations of the surface under the lighting and view conditions at the time of capture. A consistency measure of this form could be obtained through a non-Lambertian parametric model of reflectance or alternatively by enforcing a rank constraint on the observations [30]. We chose the former approach and derive our consistency measure based on the Phong model of reflectance.

Under the Phong model, we assume that the k brightest image observations of a point are due to specular reflection; the remaining intensity observations are the result of Lambertian reflectance. The photo-consistency function f evaluated at a point $\mathbf{X}$ with unit surface normal $\mathbf{N}$ is then

$$f(\mathbf{X}, \mathbf{N}) = \min_{\alpha} \left(\sum_i h_i |I_i(\mathbf{P}_i \mathbf{X}) - \int_\theta \int_\phi \alpha \mathbf{N} \cdot L(\theta, \phi)|^2 \right) \tag{2}$$

Figure 2. A toy house rendered with a dynamic texture under two light conditions.

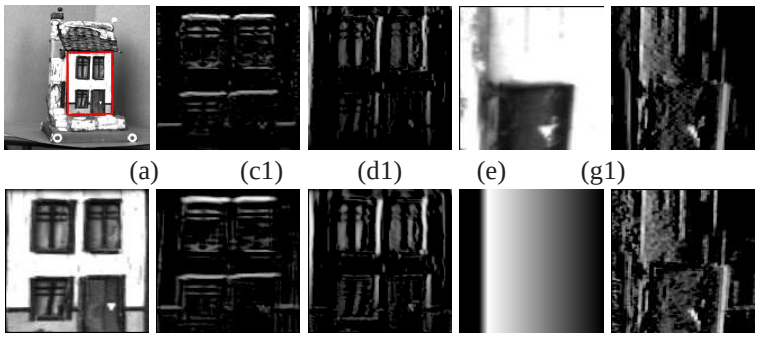

Figure 3. Comparison of analytical and estimated basis for geometric variability. Plane variability: (a) original quadrilateral; (b) warped texture; (c1),(d1) analytical basis ($\mathbf{b}_1, \mathbf{b}_4$ from Eq. 7); (c2),(d2) corresponding recovered $\tilde{\mathbf{b}}$-basis. Parallax variability: (e) reference texture image; (f) depth map; (g1) analytical basis; (g2) recovered basis by PCA

with $\mathbf{P}_i\mathbf{X}$ denoting the projection of point $\mathbf{X}$ in image i and α denoting the unknown albedo. The function h_i is a binary valued function returning 0 if $\mathbf{X}$ is occluded or $I_i(\mathbf{P}_i\mathbf{X})$ is one of the k brightest observations and returning one otherwise. It therefore filters specular pixels. $L(\theta, \phi)$ indicates the incident light from a particular direction. The light is modeled as a single point light source. Note that the function f is assigned the residual after fitting the best possible albedo to the image observations.

A surface that minimizes Eq. 1 is found by moving the initial input geometry by the Euler-Lagrange form derived from Eq. 1. We used a simplified version of the evolution equation (ignoring higher order terms) given by:

$$S_t = (2\kappa f - \nabla f)\mathbf{N} \tag{3}$$

where κ is the mean curvature of the surface.

Up until this point, we have been treating the surface as a continuous object, yet in practice it is necessary to discretize both the surface representation and the evolution equation. A modern direct approach involves recovering the displacements from the base geometry by adjusting the displacement values according to the evolution equation[3]. This approach is similar to variational disparity map estimation, but now on the model facets instead of image plane. More conventional approaches move the vertices of the mesh and subdivides the triangles of the mesh whenever necessary (as done by [9]). We have implemented both direct displacement estimation on triangles and a mesh refinement method. For the latter once we have obtained a refined geometry it is a straightforward process to obtain the displacement for the macro geometry (e.g. by ray triangle intersections).

The displacement mapped *meso* representation can be efficiently rendered using existing hardware accelerated techniques[17]. Our current implementation computes a number of sample points along each view ray through the displacement mapped volume in a pixel shader. The two sample points that are found to be on either side of the intersection of the displaced geometry are used to determine a planar approximation of the displaced surface. The ray intersection with this planar surface gives the proper texture sample for that ray, effectively ray-tracing parallax.

2.3. Micro level: BTF and view dependent textures

The purpose of the micro-scale representation is to capture intensity variation on the pixel level. This includes the view dependency of potentially complex light surface interactions and subpixel surface structure. In practice, for models captured from images another important purpose is to compensate for discrepancies between captured geometry and true object surfaces. Hence, the first purpose is similar to that of Bi-directional Texture Function (BTF) representations [6] and the second similar to view-dependent textures (VDTM) [7]. In a parellel line of research Freeman, Adelson and Heeger noted that small motions could be modulated using a spatial basis[12]. This was extended to image synthesis of whole motion sequences using a PCA basis[18], and later used to animate also stochastic motion [8]. The above works were all representing variation on one image plane, but others realized that it is more efficient to represent the variation on the surface of a triangulated model [13, 5]. Both in spirit and actual implementation all these representations are quite similar in their use of a set of basis textures/images to modulate a new texture.

Below we will derive a spatial basis (a set of textures) representing the micro layer information. Intuitively, this can be thought of as a way of compressing a photo-consistancy residual, but as we shall see, an analytic treatment gives us insight into more specific and compact representations with better modulation/interpolation properties of new views.

Textures T_k are extracted from scene images I_k through a texture coordinate transformation (Sect.3) $T(\mathbf{x}) = I(\mathcal{W}(\mathbf{x};\mu))$ where $\mathcal{W}(\mathbf{x};\mu)$ is a projective homography transform (warp), and μ is a vector of the 8 homography parameters. In view dependent texturing an approximation of the texture at a particular view direction $\hat{T}$ is generated by interpolating/blending the closest sample textures $\hat{T} = \sum a_k T_k$. Flattening all our texture images into col-

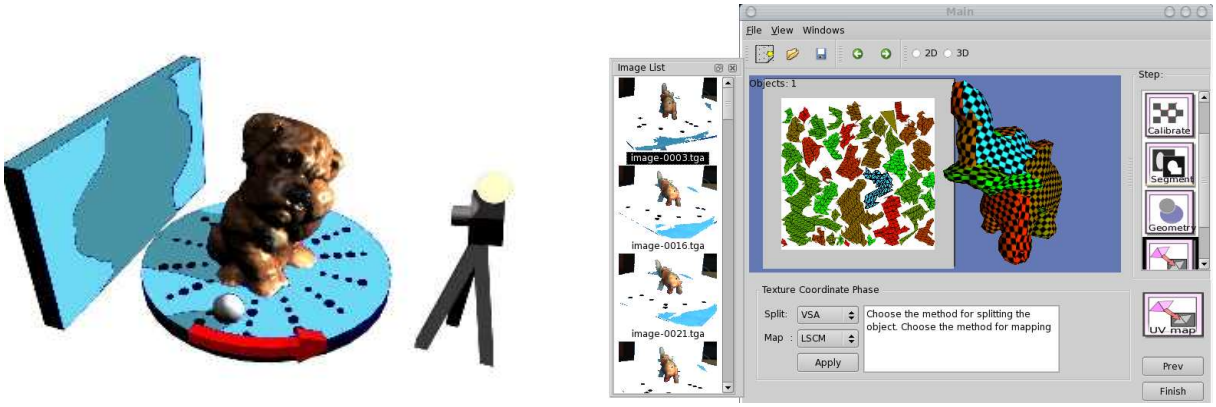

Figure 4. Left: experimental capture-setup. Right: GUI for our capture system illustrating consistent texture space.

umn vectors and arranging them in a matrix $\tilde{T}$ we can thus write $\hat{\mathbf{T}} = [\mathbf{T}_1 \dots \mathbf{T}_n]\mathbf{a} = \tilde{T}\mathbf{a}$. Now if the captured geometry is reasonably good $\tilde{T}$ will be highly compressible and we can derive a lower dimensional basis B s.t. $\tilde{T} \cong BY$ e.g. through PCA. Hence a scene or object that is captured from hundreds of views can be re-rendered using much fewer basis vectors in B. The final texture modulation equation becomes $\hat{\mathbf{T}} \approx BY\mathbf{a} = B\mathbf{y}$ where $\mathbf{y}$ is now in a transformed texture modulation space.

Through a bit of analysis a better insight into how to compute B and what it represents can be obtained. The key is that any smooth intensity change described by a function $I(f(x,\mu))$ can be linearized about the current state μ. Specifically relevant to the capture and rendering of natural scenes from images are rendering errors caused by imperfections in the captured geometry or its alignment with the images. The former causes parallax errors when the true and modeled object surfaces differ, and the latter causes planar errors (in texture space) due to the texture not being sourced from the right coordinates in the input image set. It can be shown (See Appendix) that the texture coordinate slip is modeled by an 8D basis B_h and parallax to a first order is modeled by a 2D basis B_p. Furthermore, it has been shown that the view-dependency of light, can be represented to 98% through a 9D basis of spherical harmonics[25], here denoted B_l. An example rendering is in Fig. 2.

By composing all these linear variabilities we arrive at a formula for view dependent textures from the simultaneous effects of geometric and light variability represented in a 20-dimensional spatial basis B.

$$\hat{\mathbf{T}}_k = [\mathbf{T}_0, B_h, B_p, B_l][1, y_2, \dots, y_{20}]^T = B\mathbf{y}_k \quad (4)$$

with respect to a reference texture T_0 (chosen e.g. from one view central in the sample set).

To practically compute the texture basis we obtain a pose-labeled texture set from an input video sequence $\tilde{T} = [T_1 \dots T_n]$. Just as derivatives can be either analytical, or estimated by discrete differences, here we show how to estimate the texture basis. Note that B is contained as a subspace in $\tilde{T}$, i.e. $\mathrm{span}(B) \subset \mathrm{span}(\tilde{T})$. To be able to modulate textures from new viewpoints, we wish to extract a compact approximation of B. If there was no other variability in the video sequence, $\tilde{T}$ would span exactly B. In practice $\tilde{T}$ is full rank and contains variability also due to noise etc. Our strategy is to extract from $\tilde{T}$ a linear subspace $\tilde{B} = [\tilde{\mathbf{b}}_1 \dots \tilde{\mathbf{b}}_r]$ somewhat larger than B using PCA. Hence we pick a 20 to 32-dimensional subspace from the hundreds or more video images in $\tilde{T}$.

To validate that this 20-dimensional subspace actually contain B_p and B_h (as derived in Appendix) we computed both the analytical and PCA based variability for some texture elements on the house in Fig. 1. We found that $\tilde{B}$ contained 99.5% of the variability in the analytical basis B. Additionally, through a basis transform, the columns of $\tilde{B}$ aligned with a known B, and as illustrated in Fig. 3, the analytical and estimated basis vectors look virtually identical.

The important conclusion to draw here is that when an appropriate size texture subspace-basis is estimated from a dense video sequence it will span the analytical basis. Unlike VDTM, where regular images are blended, this basis contains derivatives of images and Eq. 4 can thus be interpreted as a first order Taylor expansion, allowing continuous modulation of texture changes to correctly interpolate intermediate views instead of fading between images.

3. System and Experiments

We developed a software integrating the steps from images to to our multi-tier model into a procedure taking only a few minutes in most cases, see Video 1[1] . To quickly capture views from all sides of an object we use a rotating platform (Radio Schack TV stand). Our SW can take live video from an IEEE1394 camera, (we use a Unibrain web cam and Pt-Grey Scorpion 20SO) or import digital image files from a still camera. Camera calibration is obtained with a pattern, light using a specular ping-pong ball and object silhouettes through bluescreening, Fig. 4. The geometry is then computed as in Sect. 2.1. Alternatively, a geometry can be imported from KU Leuven's 3D Webservice[34].

While in computer vision it is common to texture directly from images, in applications a unified texture space is de-

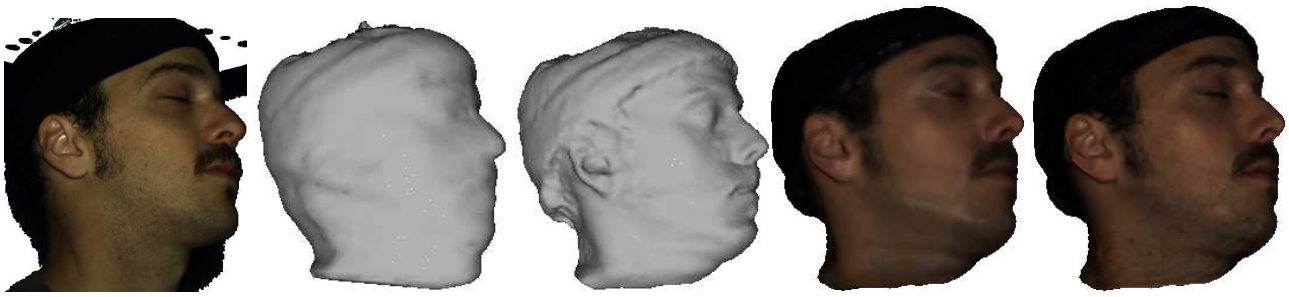

Figure 5. From left to right: an input image, SFS model, refined geometry, a rendering with the estimated Phong model, rendering with Phong model + residual dynamic texture. Notice the increase in texture quality in the renderings with the dynamic texture. See video 4.

sired and often necessary. To automatically compute texture coordinates, the object geometry is first split along high curvature regions, then each region is flattened using a conformal mapping[20], and packed into an OpenGL texture square (the GUI screenshot in Fig. 4 illustrates an example of this mapping). In the dynamic texture basis computation, all input images are transformed into and processed in this space.

Finally, the complete model of geometry and texture basis can be exported, either for inclusion in Maya or Blender, for which we have written a dynamic texture rendering plugin, or direct real-time rendering. The DynTex basis (the largest component of our three-tier model) compresses well using jpeg, and model storage size (typ. 50kB-5MB total) is proportional to size and number of images/basisvectors. For real time rendering, we perform the texture modulation in graphics HW using either register combiners or when available, fragment shaders. This allows real-time rendering of scenes with about a dozen objects even on consumer PC's.

We are not limited to small objects. As mentioned we can import geometries from 3Dwebservice[34]. In Video 3[1] and Fig. 6, we show a preachers chair captured in situ from the Seefeld church. Video 4[1] and Fig. 5 shows a human face captured by having the person sit on a rotating office chair with the calibration pattern taped to a hat, and as seen in Video 1[1] , we can also capture whole persons using a bigger rotating platform.

Figure 6. Seefeld Kanzel: image, geom, DynTex rendering

Experimental render quality comparison

In our comparative experiments we set out to validate the three-tier model on a set of scenes starting with simple geometries and little appearance difficulty and finishing with ones where both geometry and appearance challenge today's methods (including ours). Unlike comparisons of geometry alone, numeric errors are not indicative of perceptual quality. Furthermore, a static image does not show how light and specularities move. Therefore we rely mainly on the video renderings to argue photo-realistic results. In each case, a set of input images were acquired using the turntable setup. Half were used to compute the model, and the other half (from different viewpoints) were used as reference in the comparison videos, and intensity error computation. We downloaded the temple images, and captured the others using the PtGrey camera at 800x600 resolution. Due to the calibration pattern taking up image space, the effective object texture resolution is however closer to web-cam VGA (640x480) resolution. A DynTex basis of 20 basis vectors is compared to VDTM texturing by blending textures sourced from 20 input images, and unstructured lumigraph rendering. In the lumigraph, for each texturespace pixel a list of ray color and u,v index is stored. The lumigraph is then computed on the geometry by picking the 20 rays per texture pixel that minimize the reprojection error over all input images. (Note: Unlike the VDTM and DynTex basis, jpeg compression does not work for the ray indices).

For a comparison to existing literature, we start with the temple scene from[26] (Fig. 7, I.) A close approximation to the true geometry is computed using SFS already at the *macro* stage by our system, with 90% reconstructed within 1.7mm of ground truth, a further refinement improves this to 1.1mm . Lacking a way to explicitly recover light in these images we cannot apply our variational reflectance and geometry method in Sect. 2.2, but first do a volume refinement, Hernandez et.al.[10], and then use the variational method to fit a surface to this. Our geometry is not quite as good as Hernandez et.al. (0.5mm), but comparing texture renderings for the initial SFS model with those of the refined model, there is next to no perceptual difference. Likewise for this simple

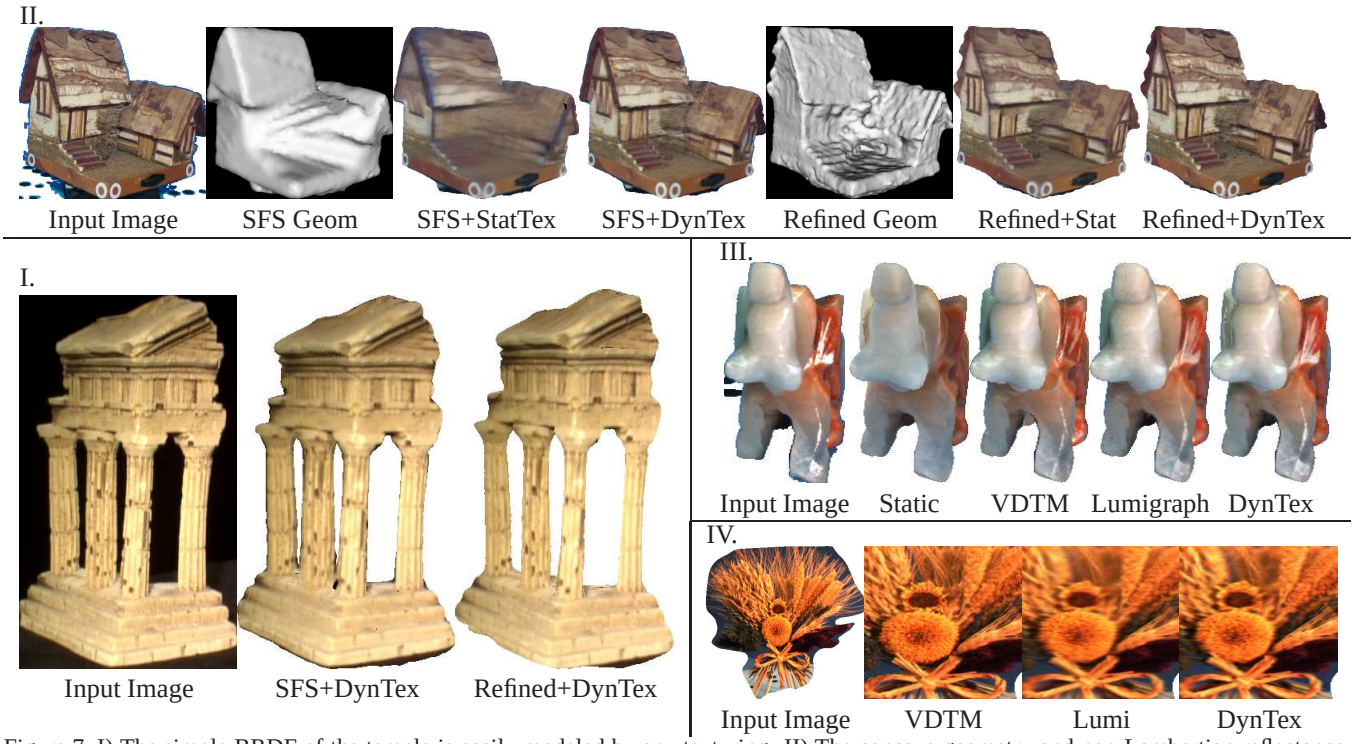

Figure 7. I) The simple BRDF of the temple is easily modeled by any texturing. II) The concave geometry and non-Lambertian reflectance of the house showcase the effectiveness of the DynTex. III) The DynTex reproduces complex view-dependent specularities that are lost in a single texture. IV) Results for an object with complex micro-geometry rendered on a proxy geometry.

BRDF we find little perceptual or numerical error difference between using just a conventional static single texture or any of the view dependent textures (see Video 5[1]). Here light is not separated and textures encode light and reflectance combined.

Our second data set is of a house, with wood, bark and moss materials, and a more complex both macro and micro structure. In this case we have a significantly different geometry between the SFS and refined case with the 90%-tile between them at 15mm, (Size of house is 140mm.), and hence the *meso* refinement step is crucial. Also can be seen in Video 6[1] and Fig. 7 II now the static texture on either geometry compares badly to the *micro* DynTex.

Third we try an elephant carved in jade. This has a complex reflectance with both specularities and subsurface scattering. Here a single texture gives a dull flat appearance. VDTM is perceptually better, but a close analysis shows that some specularities are missing (e.g. on ears in Fig.7 III), and others have incorrect gradients. The DynTex and unstructured lumigraph show slightly better results both visually and numerically for difficult (particularly specular) views, with a max intensity error of 6% compared to 10% for the standard view dependent texture and 19% for a static texture, Fig. 7 III (Video 7[1]).

Finally, we show an example of a straw wreath, where obtaining a good geometry is very difficult (Fig. 7 IV, Video 8[1]). Here, a purely image-based method can represent a dense sample of the rayset, but at a huge storage (gigabytes) cost. We extract a rough visual hull, and notice that view and dynamic textures render a view that is somewhat corrupted. For the view texture two input images are blended on top, creating a wreath with more straws. Both the DynTex and Unstructured Lumigraph code view dependency in texture space (though in different ways). These instead blur detail.

Summarizing the experiments we find that for simple reflectance and geometry, any texturing method works well, while for more complex cases, view-dependent appearance modeling helps, and for the two most complex cases the DynTex has a better performance than VDTM. Both of these can be rendered using texture blending. The unstructured lumigraph has similar performance to the DynTex, but at a much higher storage cost, and would require a complicated plugin to render in with Maya or Blender. The maximum image errors and error variance are summarized in Table 1. The variance indicates smoothness of texture modulation over viewpoint changes. Perceptually a high value manifests itself as a jumpy appearance change. An example of viewpoint error variation can be seen in Fig. 8. The jumpy appearance of the VDTM is due to it working better when close to an image in the reference set. Finally we show an example of composing several of the objects and two persons using Blender into a AR scene, using a cylindrical panorama of a city as backdrop, Fig. 9, Video 1[1] .

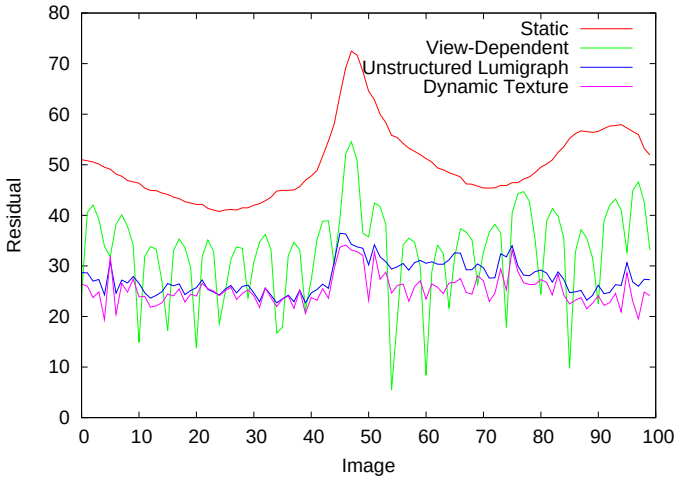

Figure 8. Viewpoint variation of rendering error for the wreath

Figure 9. Several objects and persons composed in Blender

	macro	*meso*	*micro*
representation	triangulated mesh	displ. map [17] relief tex [23]	BTF [6], PTM [21], tensor. tex. [13, 33] VDTM [7],light fields [4, 36]
capture	str.-from-motion (SFM) [16] shape-from-silh. (SFS)[28]	multi-view stereo [26] shape-from-shading , photometric stereo[35] variational methods [11]	BTF [6] VDTM [7] dynamic textures [18, 5, 13, 8] light fields [4, 36]

Table 2. Example of different possible graphics representations for the three layers and matching computer vision capture techniques.

err (var)	temple	house	eleph.	wreath
Static	10.8(1.5)	11.8(1.2)	19.0(1.4)	28.4(2.8)
VDTM	8.3(1.9)	9.8(1.3)	10.1(1.9)	21.4(3.5)
Lumigr	10.8(2.5)	9.8(1.2)	5.9(0.7)	14.3(1.3)
DynTex	7.3(1.0)	9.4(1.0)	6.6(0.7)	13.4(1.2)

Table 1. Numerical texture errors and variance. %-scale.

4. Discussion and Conclusions

We have proposed and demonstrated the use of a 3-tier model, where on the *macro* scale level a coarse approximate 3D geometry is captured from 2D images using SFS or SFM. This geometry serves as an initial approximation for and simplifies the image-based refinement using a Phong photo-consistency functional of a detailed surface geometry. On the *meso* level the detailed surface is represented as a displacement map. Finally, on the *micro* scale level, complex light and small geometric residual is represented as a stationary basis (set of textures), which at render time is modulated to play a movie representing the view dependent variation.

Another way to view the 3 scale tiers is that with respect to input video/image sequences, the macro level partially stabilizes the video when projected into texture space. This puts it within the convergence range of the surface refinement algorithm, which in turn improves the stabilization sufficiently that a quite small set of stationary basis images/textures can fully capture any perceptible difference between input reference video and rendered images. Furthermore, as we have shown that the basis textures are the first order term of a Taylor expansion (not just any compressed sample textures), correct intermediate poses can be interpolated.

While 2-tier representations have been used in the past, we find that the addition of a 3rd level has significant advantages. It breaks the difficult problem of 3D capture from 2D images into more manageable pieces. It results in more compact representations, and these representations map well to the capabilities of graphics HW rendering. While we showed a specific example of algorithms for capture and rendering at each level we also pointed out (Table 2) that many other techniques in the literature could independently be substituted on each level. Hence we believe that the 3-tier model will be more persistent and important than any of its component methods.

References

[1] Movies of the experiments are available at http://www.cs.ualberta.ca/∼vis/scalesmod. 2, 4, 5, 6

[2] B. G. Becker and N. L. Max. Smooth transitions between bump rendering algorithms. In *SIGGRAPH '93*. 1

[3] N. Birkbeck, D. Cobzas, and M. Jagersand. Object centered stereo: displacement map estimation using texture and shading. In *3DPVT*, 2006. 3

[4] C. Buehler, M. Bosse, L. McMillan, S. Gortler, and M. Cohen. Unstructured lumigraph render. In *SIGGRAPH'01*. 1, 7

[5] D. Cobzas, K. Yerex, and M. Jagersand. Dynamic textures for image-based rendering of fine-scale 3d structure and animation of non-rigid motion. In *Eurographics*, 2002. 3, 7

[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):1–34, 1999. 1, 3, 7

[7] P. E. Debevec, C. J. Taylor, and J. Malik. Modeling and rendering architecture from photographs: a hybrid geometry- and image-based approach. In *SIGGRAPH '96*. 3, 7

[8] G. Doretto, A. Chiuso, Y. N. Wu, and S. Soatto. Dynamic textures. *Int. J. Comput. Vision*, 51(2):91–109, 2003. 3, 7

[9] Y. Duan, L. Yang, H. Qin, and D. Samaras. Shape reconstruction from 3d and 2d data using pde-based deformable surfaces. In *ECCV*, 2004. 3

[10] C. H. Esteban and F. Schmitt. Silhouette and stereo fusion for 3d object modeling. *CVIU*, 96(3):367–392, 2004. 5

[11] O. Faugeras and R. Keriven. Variational principles, surface evolution, pde's, level set methods and the stereo problem. *IEEE Trans. Image Processing*, 7(3):336–344, 1998. 2, 7

[12] W. T. Freeman, E. H. Adelson, and D. J. Heeger. Motion without movement. In *SIGGRAPH'91, pp27-30*. 3

[13] R. Furukawa, H. Kawasaki, K. Ikeuchi, and M. Sakauchi. Appearance based object modeling using texture database: acquisition, compression and rendering. In *EGRW '02*. 3, 7

[14] Y. Furukawa and J. Ponce. Accurate, dense, and robust multi-view stereopsis. In *CVPR*, 2007. 2

[15] A. Ghosh, S. Achutha, W. Heidrich, and M. O'Toole. Brdf acquisition with basis illumination. In *ICCV*, 2007. 1

[16] R. I. Hartley and A. Zisserman. *Multiple View Geometry in Computer Vision*. Cambridge University Press, 2000. 2, 7

[17] J. Hirche, A. Ehlert, S. Guthe, and M. Doggett. Hardware accelerated per-pixel displacement mapping. In *GI '04*. 3, 7

[18] M. Jagersand. Image based view synthesis of articulated agents. In *CVPR'97*. 3, 7

[19] H. P. A. Lensch, J. Kautz, M. Goesele, W. Heidrich, and H.-P. Seidel. Image-based reconstruction of spatial appearance and geometric detail. *ACM Tr. Graph.*, 22(2):234–257, 2003. 1

[20] B. Levy, S. Petitjean, N. Ray, and J. Maillot. Least squares conformal maps for automatic texture atlas generation. *ACM Trans. Graph.*, pages 362 – 371, 2002. 5

[21] T. Malzbender, D. Gelb, and H. Wolters. Polynomial texture maps. In *SIGGRAPH '01*. 1, 7

[22] W. Matusik, H. Pfister, A. Ngan, P. Beardsley, R. Ziegler, and L. McMillan. Image-based 3d photography using opacity hulls. *ACM Trans. Graph.*, 21(3):427–437, 2002. 1

[23] M. M. Oliveira, G. Bishop, and D. McAllister. Relief texture mapping. In *SIGGRAPH'00*. 7

[24] S. Paris, F. Sillion, and L. Quan. A surface reconstruction method using global graph cut optimization. *International Journal of Computer Vision*, 2005. to appear. 2

[25] R. Ramamoorthi and P. Hanarahan. On the relationship between radiance and irradiance: determining the illumination from images of a convex lambertian object. *J. Optical Soc. Am. A*, 18(10):2448–2459, 2001. 4

[26] Seitz, Curless, Diebel, Scharstein, and Szeliski. A comparison of multiview stereo reconstruction algorithms. In *CVPR*, 2006. 1, 2, 5, 7

[27] H. Shum, S. Kang, and S. Chan. Survey of image-based representations and compression techniques. 13(11):1020–1037, November 2003. 1

[28] G. G. Slabaugh, W. B. Culbertson, T. Malzbender, and R. W. Schafer. A survey of methods for volumetric scene reconstruction from photographs. In *International Workshop on Volume Graphics*, 2001. 2, 7

[29] P.-P. Sloan, X. Liu, H.-Y. Shum, and J. Snyder. Bi-scale radiance transfer. *ACM Trans. Graph.*, 22(3), 2003. 1

[30] S. Soatto, A. Yezzi, and H. Jin. Tales of shape and radiance in multi-view stereo. In *ICCV*, 2003. 2

[31] R. Szeliski. Shape from rotation. In *CVPR'91,pp625-630*. 1

[32] M. Tarini, M. Callieri, C. Montani, and C. Rocchini. Marching intersections: an efficient approach to shape from silhouette. In *Proceedings of VMV 2002*, 2002. 2

[33] M. A. O. Vasilescu and D. Terzopoulos. Tensortextures: multilinear image-based rendering. *ACM Trans. Graph.*, 23(3):336–342, 2004. 7

[34] M. Vergauwen and L. V. Gool. Web-based 3d reconstruction service. *Mach. Vision Appl.*, (17):411–426, 2006. 1, 2, 4, 5

[35] M. Weber, A. Blake, and R. Cipolla. Towards a complete dense geometric and photometric reconstruction under varying pose and illumination. In *BMVC*, 2002. 7

[36] D. N. Wood, D. I. Azuma, K. Aldinger, B. Curless, T. Duchamp, D. H. Salesin, and W. Stuetzle. Surface light fields for 3d photography. In *SIGGRAPH '00*. 1, 7

[37] K. Yerex, N. Birkbeck, and M. Jagersand. A quick and automatic image based modeling and rendering system. In *Eurographics, Short Pres.*, 2004. 1

[38] T. Yu, N. Xu, and N. Ahuja. Recovering shape and reflectance model of non-lambertian objects from multiple views,. In *Proceedings of Computer Vision and Pattern Recognition (CVPR'04)*, volume 2, pages 226–233, February 2004. 2

[39] T. Yu, N. Xu, and N. Ahuja. Shape and view independent reflectance map from multiple views. In *ECCV (4)*, pages 602–616, 2004. 2

A. Analytical derivation of the texture basis

Given a warp function $\mathbf{x}' = \mathcal{W}(\mathbf{x}, \mu)$ we study the residual image variability introduced by the imperfect stabilization achieved by a perturbed warp $\mathcal{W}(\mathbf{x}; \hat{\mu})$, $\Delta T = T(\mathcal{W}(\mathbf{x}; \hat{\mu}), j) - T(\mathcal{W}(\mathbf{x}; \mu))$. Similar image variability has been used for visual tracking. Denoting $\hat{\mu} = \mu + \Delta\mu$ we rewrite ΔT as a first order approximation (dropping j):

$$\begin{aligned}
&\Delta T = T(\mathcal{W}(\mathbf{x}; \mu + \Delta\mu)) - T_W \\
&= T(\mathcal{W}(\mathbf{x}; \mu)) + \nabla T \frac{\partial \mathcal{W}}{\partial \mu} \Delta\mu - T_W \\
&\approx \nabla T \frac{\partial \mathcal{W}}{\partial \mu} \Delta\mu \\
&= \left[\frac{\partial T}{\partial u}, \frac{\partial T}{\partial v}\right] \begin{bmatrix} \frac{\partial u}{\partial \mu_1} & \cdots & \frac{\partial u}{\partial \mu_k} \\ \frac{\partial v}{\partial \mu_1} & \cdots & \frac{\partial v}{\partial \mu_k} \end{bmatrix} \Delta[\mu_1 \dots \mu_k]^T
\end{aligned} \tag{5}$$

Textures facets $\mathbf{T}$ are warped onto the rendered views using a a projective homography.

$$\begin{bmatrix} u' \\ v' \end{bmatrix} = \mathcal{W}_h(\mathbf{x}_h, \mathbf{h}) = \frac{1}{1 + h_7 u + h_8 v} \begin{bmatrix} h_1 u & h_3 v & h_5 \\ h_2 u & h_4 v & h_6 \end{bmatrix} \tag{6}$$

Specializing Eq. 5 with the derivatives of $\mathcal{W}_h$ we get:

$$\begin{aligned}
&\Delta\mathbf{T}_h(u, v) \\
&= \tfrac{1}{c_1}\left[\tfrac{\partial \mathbf{T}}{\partial u}, \tfrac{\partial \mathbf{T}}{\partial v}\right] \begin{bmatrix} u & 0 & v & 0 & 1 & 0 & -\frac{uc_2}{c_1} & -\frac{vc_2}{c_1} \\ 0 & u & 0 & v & 0 & 1 & -\frac{uc_3}{c_1} & -\frac{vc_3}{c_1} \end{bmatrix} \begin{bmatrix} \Delta h_1 \\ \vdots \\ \Delta h_8 \end{bmatrix}
\end{aligned} \tag{7}$$

where $c_1 = 1 + h_7 u + h_8 v$, $c_2 = h_1 u + h_3 v + h_5$, and $c_3 = h_2 u + h_4 v + h_6$. Finally, arranging all pixels (u, v) above into column vectors, and identifyingthe homography parameters $[\Delta h_1 \dots \Delta h_8] = [y_1 \dots y_8]$ as our texture modulation coefficients $\mathbf{y}$ we obtain

$$[\mathbf{b}_1 \dots \mathbf{b}_8][y_1, \dots, y_8]^T = B_h \mathbf{y}_h \tag{8}$$

Similarly, by analytically expressing the pixel parallax warp an equivalent 2D basis B_p can be derived linearly relating intensity change as a function of view angle.