Image-Based Modeling Using Viewpoint Entropy

Pere-Pau Vázquez[†] Miquel Feixas[†] Mateu Sbert[†]
Wolfgang Heidrich[*]

[†] Institut d'Informàtica i Aplicacions, University of Girona, Spain
[*] University of British Columbia, Vancouver, Canada

Contact: Pere-Pau Vázquez
Address: Institut d'Informàtica i Aplicacions
Campus Montilivi, EPS, E-17071, Girona, SPAIN,
Tel: 34-972-418761 Fax: 34-972-418792
e-mail: pvazquez@ima.udg.es

Abstract

We present a new method to automatically determine the correct camera placement positions in order to obtain a minimal set of views for Image-Based Rendering. The viewpoints should cover all visible polygons with an adequate quality, so that we sample the polygons at enough rate. This allows to avoid the excessive redundancy of the data existing in several other approaches. The localisation of interesting viewpoints is performed with the aid of an information theory-based measure, dubbed *viewpoint entropy.* This measure can be used to determine the amount of information seen from a viewpoint. We have also developed a greedy algorithm that aims to minimise the number of images needed to represent a scene.

In contrast to other approaches, our system uses a special preprocess for textures to avoid artifacts appearing in partially occluded textured polygons. Therefore no visible detail of these images is lost.

Keywords: Image-Based Modeling, Image-Based Rendering, Viewpoint Selection, Entropy.

1 Introduction

Image-Based Rendering (IBR) [1, 2] has received a great interest since the last decade. It allows to compute realistic images at low cost thanks to the use of precomputed ones. However, most of the research has been focused on finding fast and reliable reconstruction algorithms instead of dealing with the important issue of correctly sampling the scene with the minimum number of images.

In this paper we present a new method to automatically select the camera positions that allow to minimise the amount of images used as representation. Such an algorithm should fulfill two conditions: all visible polygons must be covered, and the sampling rate should be high enough to allow for further reconstruction.

To decide which images are important we will use an information theory-based [3, 4] measure called *viewpoint entropy* [5]. This measure can be used to determine the amount of information captured from a viewpoint. Viewpoint entropy will be used together with a greedy algorithm to choose the minimal set of views that captures the maximum information on the scene. Moreover, we add a preprocess for textures that avoids visual artifacts produced when textured polygons are partially occluded in all available views.

The rest of the paper is organised as follows: In Section 2 we will present the previous approaches to this problem, Section 3 will present our method, in Section 4 we discuss the results, and finally, in Section 5 we conclude pointing out some lines of future research.

2 Previous Work

The problem of selecting an optimal set of images for Image-Based Rendering has not attracted much research. Most of the techniques use a fixed set of camera positions to capture the images of the scene (a notable exception is [6]). Then, these images are used to further rendering. Thus, in many cases the amount of redundant data is high, and being the capturing process costly, it becomes interesting to find a cheap way to determine which images are useful and which are not. This problem can be stated as trying to recover the maximum information of a scene with the minimum number of images. A similar problem has been examined by the robotics and AI communities, under the names of *sensor planning* or *next best view.*

2.1 Non Image-Based Modeling methods

The problem of finding a good view direction to help the user understand a scene has already been treated in Computer Graphics, but not with an Image-Based Modeling purpose in mind. Kamada and Kawai [7] consider a viewing direction to be good if it minimizes the number of degenerated faces under orthographic projection. This method fails when comparing scenes with equal number of degenerated faces and it does not ensure that the user will see a large amount of detail, as discussed in [8]. Barral et al [8] and Dorme [9] modify Kamada's coefficient in order to cope with perspective projections. Then they create a heuristic with some other parameters that weight both the number of faces seen from each point and the projected area, moreover they add an exploration parameter which accounts for the faces already visited. This way they define an evaluation function that allows to explore the scene in real time. However, they admit that they have not been able to determine a good weighting scheme for the different factors. This causes some problems with objects containing holes, as these are not captured properly by the algorithm.

In the robotics literature, the goal of selecting a small set of cameras which allow to observe all object surfaces has also been studied. Usually this problem is stated as: Determine where to place the $(N + 1)$st camera positions given N previous camera locations, for $N \geq 0$. Most approaches identify the next best view as the one that reveals the maximal amount of unknown detail of the scene being treated. Different assumptions are made in next best view systems in order to simplify the problem. Several systems require a CAD model of the scene to be known a priori. The two main approaches are: search-based and silhouette-based.

Search-based methods use optimization criteria to search a group of potential viewpoints of the next best view. Many of these methods employ range images to carve away voxels in a volumetric space. Wong *et al* [10] present an algorithm that searches all possible viewpoints, and selects the next best view as the one that can carve the most empty space voxels. This system is effective, but as pointed out by Massios and Fisher [11], such an approach may result in views that observe surfaces at very oblique angles, which is undesirable in IBR, as it yields poor sampling of colour in those surfaces.

Other approaches use the silhouettes of objects. For example, Abidi [12] develops a method that employs information theory. For a given view, a silhouette is divided into segments of equal lengths. Then, an information measure that computes the geometric and photometric entropy is found for each segment. The segment with the minimal entropy is chosen to select the next best view. This method assumes that by moving the camera to better observe the segment containing the least information, more information about the scene will be captured. Silhouette-based methods can often compute next best views more quickly than search-based approaches, however, it is not always possible to generate an accurate silhouete in an image for an arbitrary (for example indoor) scene.

2.2 Image-Based Modeling and Rendering

There are few papers which refer to the viewpoint selection process for Image-Based Modeling. Grossman and Dally [13] use 32 orthographic projections of an object. McMillan and Bishop [1] use cylindrical reference views placed on a regular grid in a scene. As none of these methods take care of sampling all surfaces, this can result in important regions of the scene remaining invisible to photographs, resulting in gaps or holes during the rendering process.

Stürzlinger [14] creates a method for sampling all visible surfaces but does not address the problem of adequate coverage. Lischinski and Rappoport [15] use 6 perpendicular depth images placed on the boundaries of a cube (LDC). Fleishman *et al* [6] present an algorithm that adequately samples the surfaces visible from a certain walking region by placing the camera on a large number of positions on the boundary of the walking zone. The coverage quality criterion for a polygon is based on the projected area on a hemisphere for a camera position. The set of cameras is selected by choosing the cameras that sample a higher number of polygons at appropriate rate. Although this method is well-suited for the problem it addresses, the ordering of the cameras is guided by the amount of polygons sampled. If we had a scene with certain regions covered with a lot of very small polygons, this method could first sample parts of the scene that cover small areas instead of chosing other regions which cover larger portion of an image with less (or closer) polygons.

Hlavac et al [16] use a set of images to represent an object. Their objective is obtaining an IBR representation to be rendered by interpolation. Consequently they choose a set of reference images positioned around the object in intervals that guarantee error bounds below some threshold during reconstruction of intermediate views. However, this method only applies to single objects, instead of scenes, and their measure can only be used to compare two images, of the same object, and it is useless for views which show different parts of the same scene. Xiang *et al* [17] study the sampling of the plenoptic function for light fields from a spectral analysis of light field signals and using the sampling theorem. The authors determine the minimum sampling rate for light field rendering.

3 Automatic Camera Placement

In order to obtain a good set of cameras to sample a scene we need a measure of the goodness of an image. Fleishman *et al* [6] consider an image to be good if it samples a high number of surfaces with a certain quality. In contrast to this, we consider a view to be good if it shows a high amount of information coming from the surfaces that are covered with good quality. The amount of information can be measured with the use of *viewpoint entropy.* Viewpoint entropy has been successfully applied to compute the best views of an object and also to automatically determine a navigation path around an object or scene for Scene Understanding purposes [5]. We proceed now to describe viewpoint entropy measure.

3.1 Viewpoint Entropy

The *Shannon entropy* [3, 4] of a discrete random variable X with values in the set $\{a_1, a_2, ..., a_n\}$ is defined as

$$H(X) = -\sum_{i=1}^{n} p_i \log p_i,$$

where $p_i = Pr[X = a_i]$, the logarithms are taken in base 2 and $p_i \log p_i$ is equal to 0 for $p_i = 0$ for continuity reasons, as stated in [3, 4]. As $-\log p_i$ represents the *information* associated with the result a_i, the entropy gives the average *information* or the *uncertainty* of a random variable. The unit of information is called a *bit*.

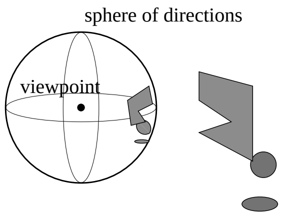

Figure 1: Viewpoint entropy is measured by projecting all the polygons in a bounding sphere centered in the viewpoint. The projected areas of every polygon are used as probability distribution of the entropy function.

To define the viewpoint entropy we use as probability distribution the relative area of the projected faces over the sphere of directions centered in the viewpoint (see Figure 1). Thus, the *viewpoint entropy* is defined [5] as

$$H_p(X) = -\sum_{i=0}^{N_f} \frac{A_i}{A_t} \log \frac{A_i}{A_t},$$

where N_f is the number of faces of the scene, A_i is the projected area of face i, A_t is the total area covered over the sphere, and A_0 represents the projected area of background in open scenes. In a closed scene, or if the point does not *see* the background, the whole sphere is covered by the projected areas and consequently $A_0 = 0$. Hence, A_i/A_t represents the *visibility* of face i with respect to the point p. It is important to remark that the area A_i/A_t is proportional to the cosine of the angle between the normal of the surface and the line from the point of view to the object, and it is inversely proportional to the square distance from the point of view to the face. Therefore, A_i/A_t grows when the face is seen at a better angle and at a shorter distance.

The maximum entropy is obtained when a certain point can *see* all the faces with the same relative projected area A_i/A_t. So, if the object does not see the background, we will have a maximum entropy of $\log N_f$. By optimizing the value of entropy in our images, we are trying to capture the maximum number of faces under the best possible orientation. We define the *best* viewpoint as the one that has maximum entropy, i.e. maximum geometric information captured.

The computation of the viewpoint entropy can be done with the aid of graphics hardware using OpenGL, in a similar way to Barral et al [8]. The projected area of each face is computed by summing up all the pixels that belong to that face, weighted by the solid angle subtended by the pixel. To distinguish between the different polygons, the faces are colour-coded in an item buffer,

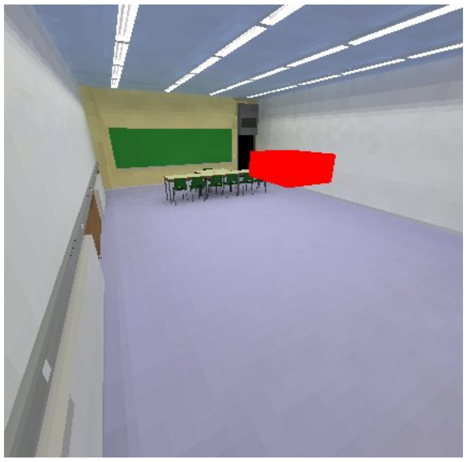

Figure 2: Scene of a classroom. The polyhedron in red depicts the walking region. The cameras are placed over its boundaries.

and to cover all the surrounding of a viewpoint six different views are used. We can also restrict to compute the viewpoint entropy of a single projection [5], by using a perspective image and changing A_t by the corresponding value of the perspective frustra.

3.2 Image-Based Modeling using Viewpoint Entropy

In this Section we present an algorithm to select a set of images that accurately represent the scene to be rendered. As Fleishman et al [6] point out, it is important to notice that adequate coverage of every surface of a scene is only possible if we can restrict the user to walk in a region empty of objects. Otherwise, if the user can approach arbitrarily close to any surface no sampling rate can guarantee a lower bound on the coverage quality. We have also used a bounding box to define the walking region (see Figure 2). To determine a set of good viewing positions our algorithm uses three steps:

1. Select the positions of the camera on the bounding box representing the walking region.
2. Compute relative projected area of each polygon from all the camera positions using graphics hardware.
3. Select the best camera positions.

The first step consists in selecting a set of points placed in regular positions on the boundary of the walking region.

The most important step is the second one. We compute five projections from the selected viewpoints. These five views cover a cube all around the viewpoint but the view which points inside the waking region. Throughout all this process we store the contribution to entropy of every visible polygon seen from each camera. In addition to this, we also store in an array the maximum projected area of each face. This information will then be used to decide which cameras are chosen first. The maximum values array will be used to determine the coverage quality of a polygon in a certain view. For a concrete scene projection, the coverage quality Q of a polygon P will be computed as $Q = (A/A_{max}) * 100$ view, where A is the actual projected area and A_{max} is the maximum projected area of the polygon in all views.

The third step performs the actual selection of the best views. This is carried out with a loop iteration where viewpoint entropy is computed for each camera position, and taking the position with higher value, and masking out the already visited polygons. However, when computing viewpoint entropy, instead of considering all visible polygons, we only use the ones which present adequate coverage. That is, we compute the amount of information captured from each view coming from the polygons which are accurately sampled (i.e. their relative projected area is above a certain, user-defined, percentage of the maximum). This does not require any extra scene projection as we saved in step two the contributions to entropy of every polygon for each view. The loop stops when all visible faces have been captured. This algorithm is depicted in Algorithm 1.

Algorithm 1 Computes the minimum set of views which samples adequately all the polygons in a scene.

```
Select a set of points placed in regular positions on the boundary of the
walking region
for all the points do
  Store an array with the projected area of each face from the point
  Update the array of maximum area projected for each face
end for
Recompute the entropies using only the faces properly covered according to
the coverage quality given by the user
Select the point with maximum entropy
Accumulate the visited faces in a bitmap
while not finished do
  Recompute entropies using only the faces properly covered which have not
  been visited yet
  Select the point with maximum entropy
  Accumulate the visited faces in a bitmap
  finished ← isFinished(numberofVisitedFaces)
end while
```

This method guarantees that we select first the camera positions which provide higher information on the scene. Previous methods [14] do not care on the accuracy of the views, or use as the best camera position the one which *sees* the higher number of polygons at good rate [6]. If some of the polygons

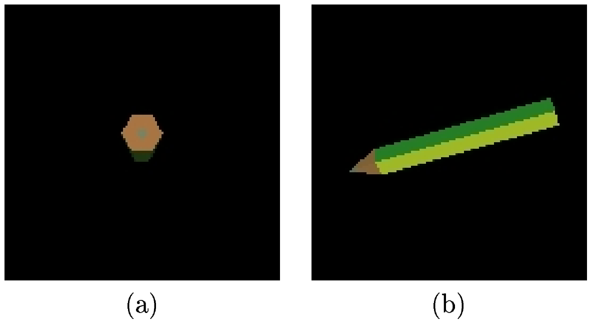
Figure 3: Two different views of a pencil. In image (a), a higher number of polygons are captured properly, but the amount of information they provide is lower than in picture (b), where only two polygons are captured properly but they give higher amount of information on the object.

appear small (are small or are far), this can lead to select an image which shows a high number of small polygons but which cover a small portion of the image, instead of choosing a different view with less but larger polygons. We can see an example of this in Figure 3. Figure 3a shows a view of a pencil which sees several polygons at a good sampling rate, although the coverage of the image is smaller than in Figure 3b where only two of the polygons are captured with appropriate rate, but covering a larger portion of the image. Consequently, if the stopping condition is changed (for example we have a limited number of cameras, or we want to stop when a certain percentage of all visible faces have already been captured), the method ensures that the set of views will show a high amount of information on the scene. Notice that determining the optimal set is NP, as it is related to the Art Gallery problem [18]. With our method we obtain a good suboptimal, which is enough.

3.3 Texture sampling

Up to now we have seen how to select a minimum set of views which captures all visible polygons in a scene with adequate coverage. However, handling with textures is more difficult. When a polygon is partially occluded in all available views, our method guarantees that we are obtaining the best projection. As large polygons of the scene are first discretised, it is likely that the whole polygon will appear in any view. However, if this does not happen, the polygon will be undersampled. In this case methods such as splatting can fill the resulting holes, but the colour used will be roughly the colour of the nearest point belonging to the same polygon. If the undersampled polygon is textured, this can lead to creating visual artifacts, as the colours used to fill the gaps might be incorrect. We can see this with an example. We have built a scene with a textured polygon and an object. The camera moves in horizontal direction in front of the objects (see Figure 4), and the polygon is partially occluded in all views. Figure 5 shows

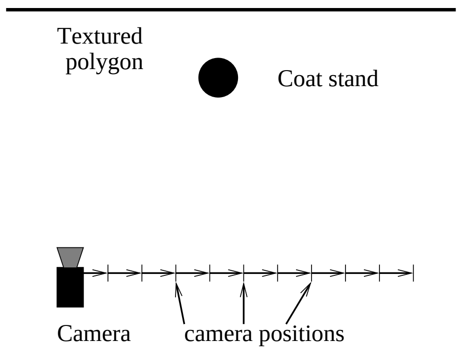

Figure 4: The path followed by the camera in front of the textured polygon and the object.

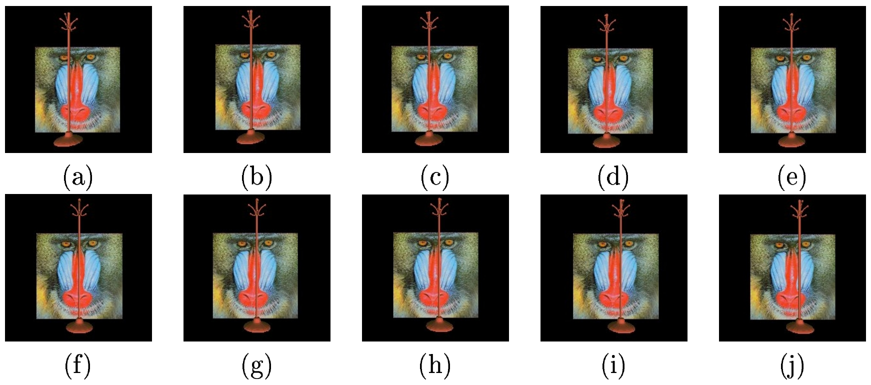

Figure 5: The ten initial images used to sample a partially occluded textured polygon. Figure (f) shows the better projection of the polygon, and will be the one selected to sample it if we do not take into account the texture. Note that there is no view showing the whole polygon without occlusions.

the captured set of views of this scene. If we were only considering the textured polygon, image 5f will be the one chosen to capture the polygon. However this would produce a hole as in Figure 6, which is difficult to fill correctly. Although usually such a big polygon will be discretised, note that this situation can happen with densely populated scenes. Moreover, discretising the polygon according to the higher frequency of the texture a huge number of unnecessary very small polygons.

In order to avoid these artifacts we segment the texture using a region growing algorithm [19] and then colour code it. This results in a polygon that can be added to the rest of the scene. Then we use the algorithm described in the previous Section to ensure appropriate sampling for every region of the image. Figure 7 shows an example. In Figure 7a the texture has been segmented using the region growing algorithm. The texture colour coded appears in 7b.

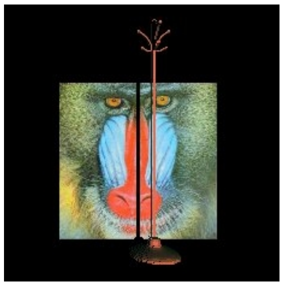

Figure 6: A hole appearing behind the coat stand. The gap is difficult to fill as we should use the correct colours appearing in the texture.

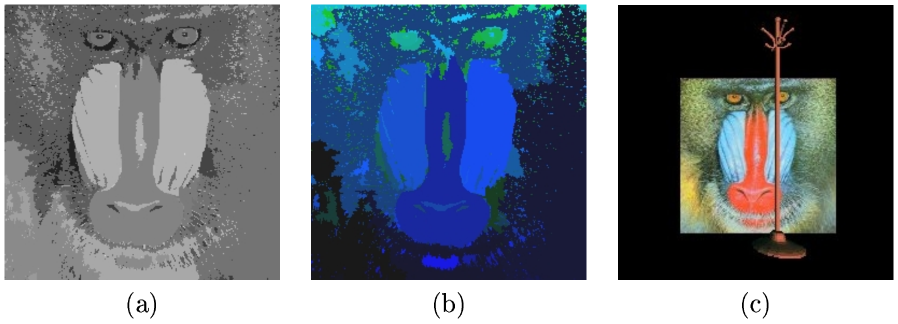

Figure 7: Processing the texture. Figure (a) shows the texture after being segmented with a region growing algorithm. (b) shows the colour-coded segmented texture. Finally, (c) shows the results. This figure has been created using the results of our algorithm with the colour-coded texture. Note that the gap behind the coat stand does not appear now.

As a consequence, views $5f$, $5b$ and $5j$ are selected. The result combining the information of these views appears in Figure $7c$.

4 Results

Texture segmentation can be done as a preprocess, although the region growing algorithm only takes some seconds. Colour-coding the texture is done while loading the segmented image. The number of new regions we achieve with this method is far below than the number of polygons that would produce a discretisation of the polygon according to the higher frequency of the texture. We have made several tests with our method and the results appear in Figure 8. For the classroom scene 51 camera postions were selected from the initial set of 150 possible views. The process takes less than two minutes in a Pentium III with 64 Mb of memory and a Nvidia TNT2 graphics card with 32 Mb of memory. Figures 8(a) to 8(d) show different views of the classroom. The representation we

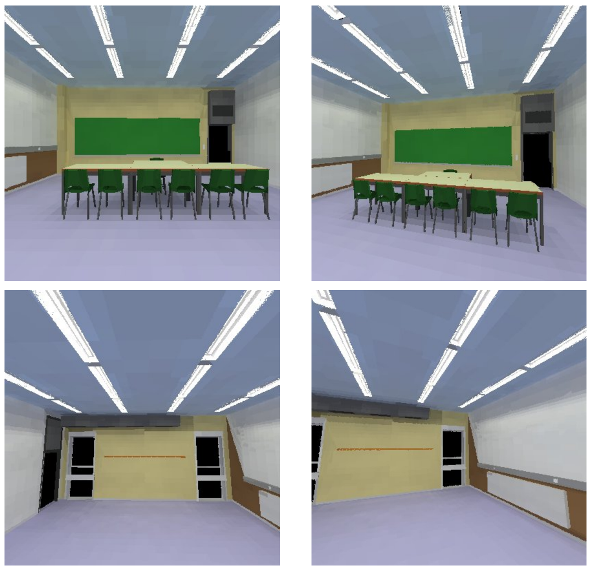

Figure 8: Examples of the classroom. The LDIs representation was created using the views selected with our method.

store is a Layered Depth Image [20], which is computed with Render Park [21] and captured by using graphics hardware. The rendering method consists in a simple projection of the captured points using graphics hardware, but our output can also be rendered with more complex systems such as QSplat [22]. Note that the quality of the result is high with our simple rendering algorithm. Our quality criterion is a percentage of the maximum projected area in all views. Either the quality criterion or the maximum number of images to compute can be set by the user.

5 Conclusions and Future Work

We have presented a new method to automatically build an Image-Based model of a scene. We make use of a measure called viewpoint entropy that determines the amount of information seen from a point. Our system also takes care of textures by using a region growing segmentation and posterior colour coding of the regions of the texture. This way we avoid visible artifacts caused by polygons which are partially occluded in all views. This situation is likely to happen in very crowded scenes or when textured polygons are not sufficiently discretised. The selection of the most important views is carried out by a greedy algorithm that obtains a set of views which cover all visible polygons with a quality defined by the user. Our method avoids redundancy in the data and, as it works using an item buffer and hardware-acceleration, we save illumination computations, which are only calculated for the selected views.

In our Future Work we will focus view-dependent illumination. A new measure has to be defined in order to cope with photometric effects. This will make our system more general to cope with any kind of materials.

Acknowledgements

This work was partially supported by BR98/1003 grant of Universitat de Girona, SIMULGEN ESPRIT project #35772, and TIC 2001-2416-C03-01 of the Spanish government. The authors want to thank Hartmurt Schirmacher for his valuable advice, and Jordi Freixenet from University of Girona for providing us with the code of region growing segmentation.

References

[1] L. McMillan and G. Bishop. Plenoptic modeling: An image-based rendering system. *Proc. of SIGGRAPH 95*, pages 39–46, August 1995.

[2] L. McMillan. *An Image-Based Approach to Three-Dimensional Computer Graphics, Ph.D. Dissertation.* PhD thesis, April 1997.

[3] R.E. Blahut. *Principles and Practice of Information Theory.* Addison-Wesley, 1987.

[4] T.M. Cover and J.A. Thomas. *Elements of Information Theory.* Wiley, 1991.

[5] Pere-Pau Vázquez, Miquel Feixas, Mateu Sbert, and Wolfgang Heidrich. Viewpoint selection using viewpoint entropy. In T.Ertl, B. Girod, G.Greiner, H. Niemann, and H.-P. Seidel, editors, *Vision, Modeling, and Visualization 2001*, 2001.

[6] S. Fleishman, D. Cohen-Or, and D. Lischinski. Automatic camera placement for image-based modeling. *Computer Graphics Forum*, 19(2):101–110, Jun 2000.

[7] T. Kamada and S. Kawai. A simple method for computing general position in displaying three-dimensional objects. *Computer Vision, Graphics, and Image Processing*, 41(1):43–56, January 1988.

[8] P. Barral, G. Dorme, and D. Plemenos. Scene understanding techniques using a virtual camera. In A. de Sousa and J.C. Torres, editors, *Proc. Eurographics'00, short presentations*, 2000.

[9] G. Dorme. *Study and implementation of 3D scenes comprehension techniques.* PhD thesis, 2001.

[10] L. Wong, C. Dumont, and M. Abidi. Next best view system in a 3-d object modeling task. In *Proc. International Symposium on Computational Intelligence in Robotics and Automation (CIRA)*, pages 306–311, 1999.

[11] N.A. Massios and R.B. Fisher. A best next view selection algorithm incorporating a quality criterion. In *Proc.of the Britsh Machine Vision Conference*, 1998.

[12] B. Abidi. Automatic sensor placement. In *Proc. Intelligent Robots and Computer Vision: Algorithms, Techniques, Active Vision, and Materials Handling*, pages 387–398, 1995.

[13] J.P. Grossman and William J. Dally. Point sample rendering. In George Drettakis and Nelson Max editors, editors, *Rendering Techniques'98*, pages 181–192. Springer-Verlag, 1998.

[14] W. Stuerzlinger. Imaging all visible surfaces. In I. Scott MacKenzie and James Stewart, editors, *Proc. of the Conference on Graphics Interface (GI-99*, pages 115–122, Toronto, Ontario, June 2–4 1999. CIPS.

[15] D. Lischinski and A. Rappoport. Image-based rendering for non-diffuse synthetic scenes. In George Drettakis and Nelson Max editors, editors, *Rendering Techniques'98*, pages 301–314, 1998.

[16] V. Hlavac, A. Leonardis, and T. Werner. Automatic selection of reference views for image-based scene representations. In *Lecture Notes in Computer Science*, pages 526–535, New York, NY, 1996. Springer Verlag. Proc. of European Conference on Computer Vision '96 (ECCV '96).

[17] Jin-Xiang Chai, Xin Tong, Shing-Chow Chan, and Heung-Yeung Shum. Plenoptic sampling. In *SIGGRAPH 2000, Computer Graphics Proceedings*, pages 307–318. ACM Press / ACM SIGGRAPH / Addison Wesley Longman, 2000.

[18] J. O'Rourke. *Art Gallery Theorems and Algorithms.* Oxford University Press, New York, 1987.

[19] Robert M. Haralick and Linda G. Shapiro. *Computer and Robot Vision, vol. 1.* Addison-Welsey: Reading, MA, 1992.

[20] L. He J. Shade, S. Gortler and R. Szeliski. Layered depth images. In *Computer Graphics Proceedings (Proc. SIGGRAPH '98)*, pages 231–242, July 1998.

[21] Philippe Bekaert, Frank Suykens de Laet, Pieter Peers, and Vincent Masselus. Renderpark: A test-bed system for global illumination. available under http://www.cs.kuleuven.ac.be/cwis/research/graphics/RENDERPARK/.

[22] Szymon Rusinkiewicz and Marc Levoy. QSplat: A multiresolution point rendering system for large meshes. In Kurt Akeley, editor, *Siggraph 2000, Computer Graphics Proceedings*, Annual Conference Series, pages 343–352. ACM Press / ACM SIGGRAPH / Addison Wesley Longman, 2000.