Automatic View Selection Using Viewpoint Entropy and its Application to Image-Based Modelling

Pere-Pau Vázquez[1] Miquel Feixas[2] Mateu Sbert[2] and Wolfgang Heidrich[3]

[1] Department of LSI, Polytechnical University of Catalonia, Barcelona, Spain
[2] IIiA, University of Girona, Girona, Spain
[3] Department of Computer Science, University of British Columbia, Vancouver, Canada

Abstract

In the last decade a new family of methods, namely Image-Based Rendering, has appeared. These techniques rely on the use of precomputed images to totally or partially substitute the geometric representation of the scene. This allows to obtain realistic renderings even with modest resources. The main problem is the amount of data needed, mainly due to the high redundancy and the high computational cost of capture. In this paper 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 input is a 3D polyhedral model including textures and the output is a set of views that sample all visible polygons at an appropriate rate. The viewpoints should cover all visible polygons with an adequate quality, so that we sample the polygons at sufficient rate. This permits to avoid the excessive redundancy of the data existing in several other approaches. We also reduce the cost of the capturing process, as the number of actually computed reference views decreases. The localization of interesting viewpoints is performed with the aid of an information theory-based measure, dubbed viewpoint entropy*. This measure is used to determine the amount of information seen from a viewpoint. Next we develop a greedy algorithm to minimize 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.*

Categories and Subject Descriptors (according to ACM CCS): I.3.7 [Computer Graphics]: Three-Dimensional Graphics and Realism

1. Introduction

In this paper we present a new method to automatically select the camera positions that allow to minimize the amount of images used as an Image-Based 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.

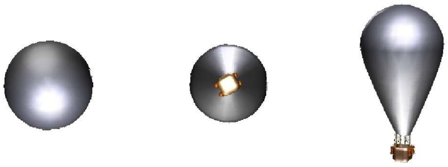


Figure 1: *Three images of a balloon, the one on the left and the one in the middle are taken from bad viewpoints, the one on the right is taken from a good viewpoint.*

To decide which images are important we will use an information theory-based[1, 2] measure called *viewpoint entropy*[3]. This measure can be used to determine the amount of information captured from a viewpoint, and therefore to select the best view of a scene (see Figure 1), given its 3D geometry. 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 organized as follows: In Section 2 we present the previous approaches to this problem, Section 3 reviews the *viewpoint entropy* measure, in Section 4 we present our method and discuss several selection strategies. In Section 5 we show how viewpoint entropy can be applied to the selection of a minimal set of reference views for Image-Based Rendering. In Section 6 we address the concrete problem of properly sampling textured polygons. In Section 7 we discuss the results, and finally, in Section 8 we conclude pointing out some lines of future research.

2. Related Work

The problem of selecting an optimal set of images for Image-Based Rendering has not attracted much research yet. Most of the techniques use a fixed set of camera positions to capture the images of the scene (a notable exception is [4]). 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 views are useful and which are not, and therefore avoid capturing data from positions which will add redundant data. This problem can be stated as trying to recover the maximum information of a scene with the minimum number of images. In this case, as we are going to use this set of images to build a Layered Depth Image, these views should show the faces at a good gazing angle to allow for appropriate rate capture. A similar problem (in most cases not considering sampling rate) has been examined by the robotics and AI communities, under the names of *sensor planning* or *next best view*.

2.1. Non Image-Based Modelling methods

The problem of finding a good viewing direction to help the user understand a scene has already been treated in Computer Graphics but not with an Image-Based Modelling purpose in mind. Kamada and Kawai[5] 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 [6]. Barral *et al.*[6] and Dorme[7] 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 permits 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. Moreover, these techniques cannot be used as they do not address the issue of appropriate covering the objects of the scene.

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 next camera position 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.*[8] 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[9], such an approach may result in views that observe surfaces at very oblique angles, which is undesirable in IBR (Image-Based Rendering), as it yields poor sampling of colour in those surfaces.

Other approaches use the silhouettes of objects. For example, Abidi[10] develops a method that employs information theory. For a given view, a silhouette is divided into segments of equal lengths. Then, an information measure based on two components, a geometric component, intended to measure the irregularity of an edge, and a contrast component, is used to determine how much of the contour of an object is well seen. The segment with the minimal entropy is chosen to select the next best view. This method assumes that more information about the scene will be captured by moving the camera to better observe the segment containing the least information. 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 silhouette in an image for an arbitrary (for example indoor) scene.

Klein and Sequeira address the problem of adequate coverage in 3D modelling from range data[11]. They build a quality function that weights both the cost of image acquisition and the visible area captured. Roberts and Marshall[12] select a minimized number of views for complete coverage of the surface of three dimensional objects. This problem is also faced by Tarbox and Gottschlich[13].

2.2. Image-Based Modelling and Rendering

There are few papers which refer to the viewpoint selection process for Image-Based Modelling. Grossman and Dally[14] use 32 orthographic projections of an object. McMillan and Bishop[15] 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[16] creates a method for sampling all visible surfaces but does not address the problem of adequate coverage. Lischinski and Rappoport[17] use 6 perpendicular depth images placed on the boundaries of a cube (LDC). Fleishman *et al.*[4] 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. This method is well-suited for the problem it addresses, however, the ordering of the cameras is guided by the amount of polygons sampled. Thus, if we had a scene with certain regions covered with a lot of very small polygons (or a viewpoint that sees a high number of polygons from a higher distance), this method could first sample parts of the scene that cover small areas instead of choosing other regions which cover larger portion of an image with less (or closer) polygons. If, in addition, we have a constrain in the number of captured views, this could leave important regions without sampling.

Hlavac *et al.*[18] and Werner *et al.*[19] 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, so it is useless for views which show different parts of the same scene. Xiang *et al.*[20] 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. Viewpoint Quality Evaluation

The purpose in our paper is to select a minimized set of views adequate for Image-Based Rendering. First, we need to choose a measure that can be used to evaluate the quality of a view. The criterion we chose is *viewpoint entropy*[3], a measure based on information theory, which can be understood as the amount of information coming from the geometry of a scene captured from a point. For the sake of completeness, we review in this section the definition and computation of viewpoint entropy. We also comment how the best view of an object can be determined. Further details can be found in a previous paper[3].

3.1. Good View Selection Criteria

In order to evaluate the quality of a single view several criteria can be chosen according to different parameters, the simpler ones are: number of faces seen and total area covered. By itself, the projected area does not tell us about the amount of detail we can see, and cannot be used for indoor scenes because the projected area is constant (unless the model contains holes or windows that let visible the background). On the other hand, even though we have a high number of faces, they could be small and thus provide little information about a scene. In Figure 2*a* and 2*b* two projections of the same scene are shown. They exhibit different number of faces and different projected area. Another example is depicted in Figure 3, where two different views of the same room are shown. Figure 3*a* shows approximately the same amount of projected area than 3*b*, but 3*b* is far more interesting and informative than 3*a*. It seems necessary to obtain a quality function that weights those two parameters. In [7] an algorithm in such direction is designed but the results are not completely satisfactory.

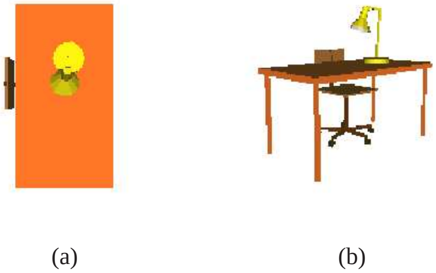


Figure 2: (a) *and* (b) *are different projections from the same scene. In* (b) *we see faces that do not appear in* (a)*, however, the area projected in* (a) *is larger than the area projected in* (b)*. It is difficult to decide which criterion is better to select a good view: the number of visible faces or the total projected area.*

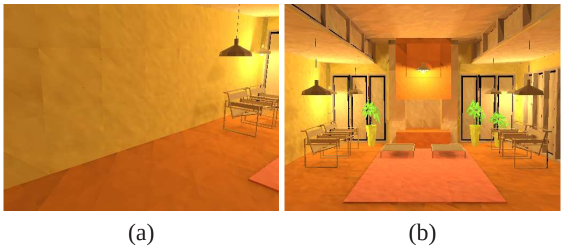


Figure 3: (a) *and* (b) *are different views of the same room (model courtesy of Karol Myzskowski). Although both views project roughly (there are some holes that make the background visible) the same amount of area because we are in a indoor scene, it is clear that image* b *is preferable to* a.

In Fleishman *et al.*[4] the good view criterion chosen is the number of faces properly captured. We will use a criterion based on *the amount of information* coming from properly covered faces. To measure the amount of information seen from a new point we borrow some tools from Information Theory[1, 2] and construct the *viewpoint entropy* measure.

3.2. Viewpoint Entropy

In order to build a good view criterion, one of the features we associate with the *quality* of a viewpoint is the amount of *geometric information* it provides us with. *Viewpoint entropy* is a new measure that allows us to obtain good viewpoints of a scene. We will see how the viewpoint entropy incorporates both the projected area and the number of faces, and can be understood as the amount of information captured by the viewpoint. In a recent work Rigau *et al.* have arrived to an equivalent measure when studying the visibility complexity of 2D scenes[21].

The *Shannon entropy*[1, 2] 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 $0 \log 0 = 0$ for continuity. 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*.

To define viewpoint entropy we use as probability distribution the relative area of the projected faces over the sphere of directions centered in the viewpoint. Thus, given a scene S and a viewpoint p, we define *viewpoint entropy* as

$$I(S, p) = -\sum_{i=0}^{N_f} \frac{A_i}{A_t} \log \frac{A_i}{A_t}, \qquad (1)$$

where N_f is the number of faces of the scene, A_i is the projected area of face i over the sphere, A_0 represents the projected area of background in open scenes, and A_t is the total area of the sphere. In a closed scene, or if the point does not *see* the background, the whole sphere is covered by the projected faces and consequently $A_0 = 0$. Hence, A_i/A_t represents the *visibility* of face i with respect to point p. It is important to remark that, with respect to the total area of face i, the projected 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. This justifies the use of projected area as the probability distribution to compute entropy.

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, in an open scene the maximum viewpoint entropy is $\log(N_f + 1)$, and in a closed scene it is equal to $\log N_f$. We define the *best* viewpoint as the one that has maximum entropy, i.e. maximum information captured. If we consider two models of a scene with different discretization, the best views might appear different. It is important to emphasize that we are capturing geometric information, and an object with a finer subdivision gives a higher entropy quantity, which is coherent with the expected result. Thus, the regions with finer subdivision will *attract* the camera's attention.

3.3. Implementation

The computation of viewpoint entropy can be done with the aid of graphics hardware using OpenGL, in a similar way to Barral *et al.*[6]. 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, and to cover all the view directions six different views are used. The background can be detected because it is set as black. However, for the concrete case where the scene is composed by a single object (or a set of objects with empty space outside a bounding sphere) and the camera is placed outside a bounding sphere of the scene, a simple view is enough for each viewpoint, provided that all the objects fall into the viewing frustum of the camera.

To compute the view with maximum viewpoint entropy of an object, we perform the following steps: the scene is rendered from a set of points placed at regular positions over a bounding sphere of the object. At each position, the item buffer is read and the viewpoint entropy is calculated. The best view will be the one with maximum viewpoint entropy. The distance of the camera and the number of viewpoints to analyze can be modified by the user. For a better approximation, a higher number of positions must be visited (unless we use an adaptive scheme).

3.4. Results

We have implemented the method described above and tested it for several objects. A deep comparison with the relative projected area can be found in Vázquez *et al.*[3]. The algorithm works at 60-65 frames per second on a P-IV processor with a NVidia GeForce 4 Go graphics card although no optimizations were made.

In Figures 4*a* and 4*b* we can see the points of maximum viewpoint entropy computed around a man and a desk. Entropy decreases with increasing distance, because the projected area of each face is smaller. On the other hand, sometimes it is not enough with a single view, as it might not provide enough information from the scene, and thus, we will need more images.

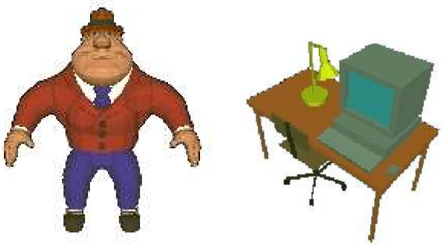


Figure 4: *The points of maximum viewpoint entropy of a man* (a), *and a desk* (b).

4. Selection of *N* Views

So far, a measure to compute the quality of a view and to find which is the best one according to geometric information has been presented. In this section we will address the problem of selecting a small set of *N* views that may be best suited for representing the scene. In complex scenes, *N* may be large, and therefore it can be set to a maximum by the user, although this does not guarantee to adequately represent the scene. Hence, two problems arise: which are the best images to choose, and how many of them are sufficient. Consequently, we will focus on finding a good distance measure in order to select views with high entropy and which are as much different to each other. Notice that the selection of an optimal set of *N* views is related to the Art Gallery problem, which is a well-known NP problem[22]. So, instead of searching an optimal set, we will find a suboptimal set with a greedy algorithm which is good for our purposes.

4.1. Computing the difference between two views

The difference on viewpoint entropy between two frames could be computed using the *Kullback-Leibler distance*[1, 2], which is defined as:

$$\delta = \sum_{i=1}^{n} p_i \log \frac{p_i}{q_i}$$

Unfortunately, this is not possible when having probability values of 0, because of the zero division, as it is our case. A solution has been proposed in order to cope with similar cases for compositional data[23], but in this case the zeroes that appear in the formula are actually very small values, thus, in our case it is not applicable.

We have tested different approaches which require the aid of a second parameter. In a first approach we encoded in a bitmap the faces that were visible from each viewpoint, then, the selection of views can be carried out according this second criterion. Some measures based on Euclidean distances (of areas or entropies) have also been built, but the one that gives better results is based on entropy recomputation. It is explained in the following section.

4.2. Entropy Recomputation

In this section we present an approach that does not try to find a good definition of distance between two views. Our method recomputes the entropy for each new view selected, but this time only taking into account the not yet visited faces. So our algorithm performs three steps:

1. For each view compute viewpoint entropy and store in an array the contribution to entropy of the visible faces from that point (zero for the non visible ones).
2. Order the views in decreasing entropy and select the first one. Accumulate the contributions to entropy of the visited faces in an array.
3. Recompute the entropy of each non selected viewpoint by only taking into account the faces which have not been visited yet.
4. Order the views in decreasing entropy and select the first one. Accumulate the contributions to entropy of the visited faces in an array. Go to step 3.

This method allows us to obtain a set of views which, for each view, shows the maximum amount of information coming from non visited faces. As will be explained later, this can be useful if we need to sample faces with appropriate rate in order to obtain an Image-Based representation[24]. Moreover, this method has the advantage that it needs no threshold if we need to sample the complete scene. An algorithm which uses this method and a criterion to determine what is a good sampling rate is presented in Section 5. Figure 5 shows the results obtained with this method for the mug model.

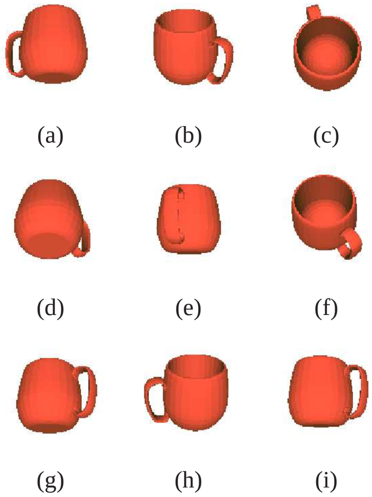


Figure 5: *Results obtained for the mug with the entropy recomputation method.*

4.3. Discussion

From the methods commented above, some of them have advantages over the others depending on the application in mind. If we need to visit all the faces of an object and it does not matter if they appear at oblique angles, we can use a simple criterion such as the encoding of the visible faces from each view. It only requires a bit per face and camera position, and it is suitable for problems that require all object or faces to appear in any view. This happens in Molecular Visualization (see Vázquez *et al.*[25]).

The method that uses entropy recomputation can be easily adapted to obtain a good Image-Based representation in the way that it is explained in the following Section[24]. In this case the memory requirements are $\#V \times \#F \times 4$ bytes where $\#V$ represents the number of camera positions and $\#F$ the number of faces of the scene.

5. Image-Based Modelling

In this Section we present our automatic camera placement system. We follow a similar scheme to the one of Fleishman *et al.*[4]. As they 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 (indicated by the semi-transparent polyhedron in Figure 6), that is initially given by the user.

5.1. Image-Based Modelling using Viewpoint Entropy

In this Section we present an algorithm to select a set of images that accurately represent the scene to be rendered. These reference views are then rendered with a realistic rendering system such as Render Park[26]. 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. These points are placed at constant distances on every face of the walking region box, as in Figure 6.

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 walking 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.

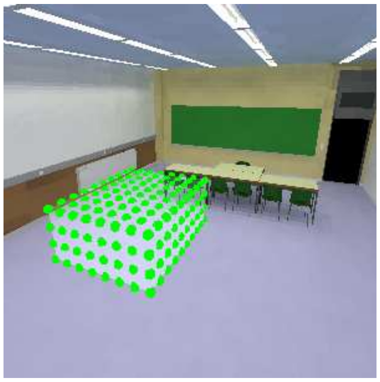


Figure 6: *The positions of the camera are selected over the semi-transparent polyhedron that bounds the walking region.*

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 $Q = (A/A_{max}) * 100$, 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. The difference between this algorithm and the steps presented in Section 4.2 is that the entropy here is recalculated taking into account only the faces which are visible at an appropriate rate, that is, they contribute to entropy only when the projected area is a high percentage (usually 90%) of the maximum. This way we ensure that the viewpoint entropy captures information of faces that can be captured at adequate rate.

5.2. Discussion

For our method to obtain the best results the scene must be tessellated. This is a normal situation in global illumination field, where the scene must be discretized in order to better calculate the distribution of light. Ideally, all the polygons

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

```
Select a set of points placed on the boundary of the walking region
for all the points do
    Store the projected area of each face from the view
    Update the array of maximum area projected for each face
end for
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 contributions to entropy in an array
end while
```

should have the same size, but this is not compulsory. Actually, the tessellation allows us to find the proper camera positions that guarantees an adequate coverage for all the polygons. If very large polygons exist in the scene, and they are partially occluded in most (or all) the reference views, the model will present holes that are not always easy to fill. As in Fleishmann *et al.*[4] we start from a discretized model to avoid such occlusions. Recall that global illumination methods discretize to smaller polygons in places where the light distribution is more difficult (such as corners) and therefore this results in more information to be treated in these regions, which is consistent with our viewpoint entropy measure. As the reconstruction process relies mainly on the *quality* of the captured geometry, this is the key issue. For a measure that evaluates the lighting information contained in an image, the interested reader can refer to Gumhold [27], Shacked and Lischinski[28], or Vázquez *et al.*[29].

Our method uses viewpoint entropy to ensure that the captured views show a high amount of information on the scene. Moreover, as we know, thanks to our previous analysis, which is the maximum visible area of each face, the selection of views guarantees that the faces will be captured nearly at the maximum sampling rates available. Previous methods[16] 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[4]. If some of the polygons 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 7. Figure 7*a* shows a view of a teapot which sees several polygons of the top of the teapot, at a good sampling rate, although the coverage of the image is smaller than in Figure 7*b*, where a lower number 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[22]. With our method we obtain a good suboptimal, which is enough.

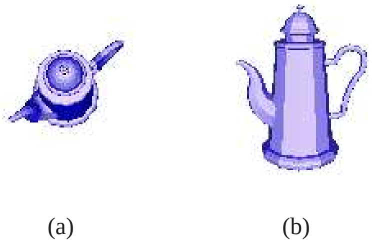


Figure 7: *Two different views of a teapot. In image (a), a higher number of small polygons are captured properly, but the amount of information they provide is lower than in picture (b), where we are clearly able to identify the object.*

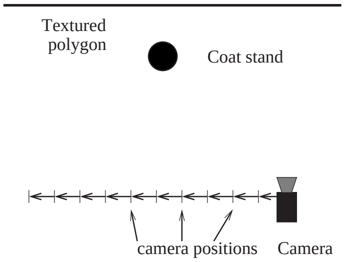


Figure 8: *The path followed by the camera in front of the textured polygon and the object.*

6. 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 discretized, 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 8), and the polygon is partially occluded in all views.

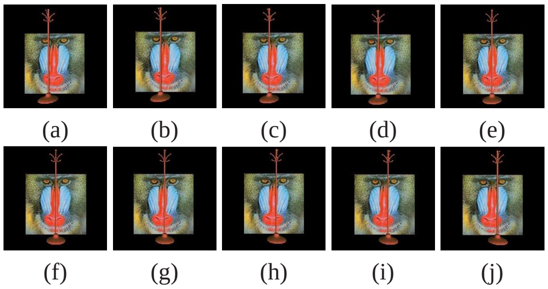


Figure 9: *The ten initial images used to sample a partially occluded textured polygon. Figure (f) shows the better projection of the polygon, and will be the one selected to sample it if we do not take into account the texture. Note that there is no view showing the whole polygon without occlusions.*

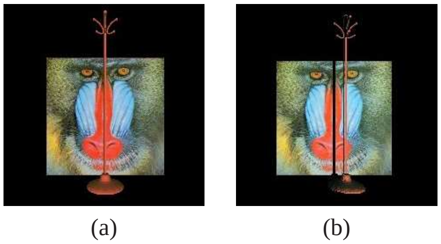


Figure 10: *In (a) we can see the view that best captures the large polygon without using the information contained in the texture. Figure (b) shows the reconstruction of the scene using this information. Note the hole appearing behind the coat stand due to poor sampling. This gap is difficult to fill as we should use the correct colours of the texture.*

Figure 9 shows the captured set of views of this scene. If we were only considering the textured polygon, Figure 9*f* (enlarged in Figure 10*b*) will be the one selected to capture the polygon. However this would produce a hole as in Figure 10*b*, which is difficult to fill correctly. Although usually such a big polygon will be discretized, note that this situation can happen with densely populated scenes. Moreover, discretizing the polygon according to the higher frequency of the texture might result in a huge number of unnecessary very small polygons.

Our purpose in this section is to design a method that avoids artifacts due to poor tessellation in textured polygons. In order to do this we use the information present in the texture to create a set of colour-coded regions that will behave as the rest of the polygons of the scene with our algorithm. We perform a three step process:

1. Segment the texture.
2. Colour code each region of the segmented texture.
3. Replace the textured polygon with this colour coded map.

We have chosen a well known robust segmentation method dubbed region growing[30]. Once the texture is segmented, each region is colour coded as if they were different polygons of the scene. Then we map the polygon with the colour coded segmented texture and treat it as if this was part of the scene. The rest of the polygons are colour coded by using colours not yet used for the texture. Then we use the algorithm described in the previous Section to ensure appropriate sampling for every region of the image. Figure 11 shows an example. In Figure 11*a* the texture has been segmented using the region growing algorithm. The texture colour coded appears in 11*b*. As a consequence, views 9*f*, 9*b* and 9*j* are selected. The result combining the information of these views appears in Figure 11*c*.

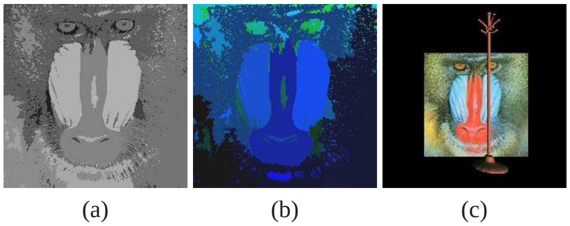


Figure 11: *Processing the texture. Figure (a) shows the texture after being segmented with a region growing algorithm. (b) shows the colour-coded segmented texture. Finally, (c) shows the results. This figure has been created using the results of our algorithm with the colour-coded texture. Note that the gap behind the coat stand does not appear now.*

7. 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 discretization 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 12. For the classroom scene 51 camera positions were selected from the initial set of 150 possible views. The process takes less than twenty seconds in a Pentium IV with 512 Mb of memory and a NVidia GeForce 4 Go graphics card. In this case, the scene contains 17200 faces from which only 11200 are visible from the walking region. If we sampled all the faces uniformly, a 35% of the capture would be useless. Figures 12*a* to 12*d* show different views of the classroom. The representation we store is a Layered Depth Image[31], which is computed with Render Park[26] 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[32]. Note that the quality of the result is high with our simple rendering algorithm.

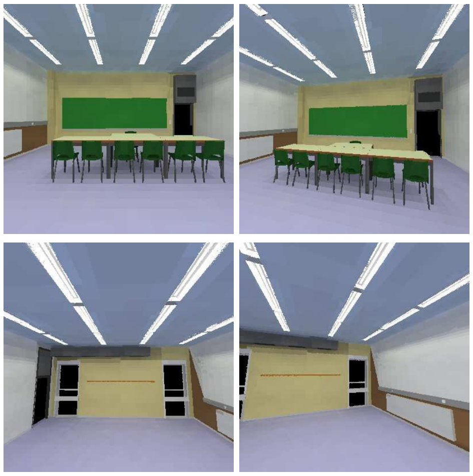


Figure 12: *Examples of the classroom. The LDIs representation was created using the views selected with our method.*

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.

Our selection method can generate a number of extra views when compared with the approach by Fleishman *et al.*[4]. This happens because their good view criterion accounts for the number of visited faces at appropriate rate while ours is the amount of information provided by the faces seen at adequate rate. However, the number of extra views we compute is not important and the amount captured faces decrease almost as fast as with the system by Fleishman *et al.*.

Figure 13 shows a comparison of the accumulated projected area selected throughout the views selection process. This demonstrates that during the selection process, the quality of the captured faces is guided better with our method than by counting the number of faces. To improve the selection method, Fleishman *et al.*[4] also add a correction: if a new view sees a face that was already captured and is now sampled better, they change the masking of the previous view and set the face to be captured with the current viewpoint. However, this solution (which is not included in Figure 13) only works when there is no limitation to the number of views to capture. If there is, view selection based on the number of faces may yield early choice of small faces that are projected under a good sampling rate but that show low amount of information of the scene. Note that the important difference in captured area appears in the first five to seven views. Using the strategy by Fleishman *et al.* this difference would softly diminish trough the computation of the rest of the views. In Figure 14 we can see the two initial views selected by both strategies. Although the number and the are of captured polygons is similar, the strategy which uses as a criterion the number of visible faces (Figures 14*a* and 14*b*) capture a slightly lower amount of area than the strategy based on entropy recomputation (Figures 14*c* and 14*d*).

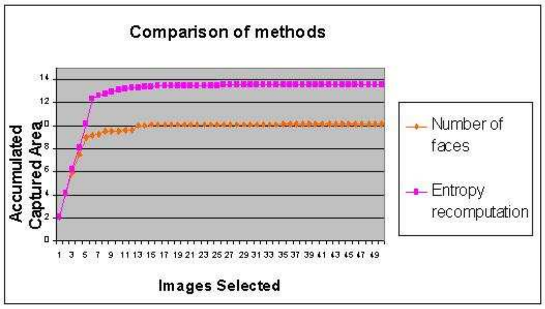
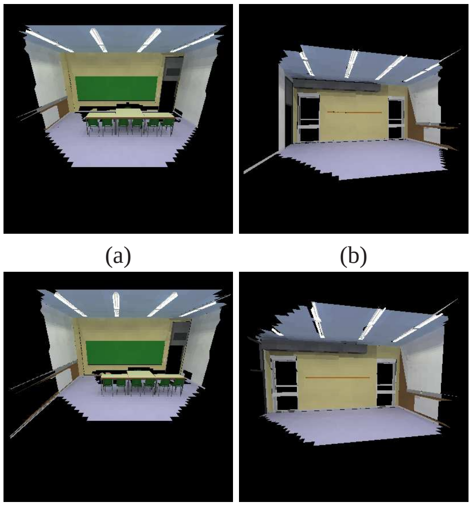


Figure 13: *Relation of the number of total accumulated projected area in the currently selected set of images.*



(a) (b)

(c) (d)

Figure 14: *The initial two images of both methods. Figures* (a) *and* (b) *show the two initial views selected by the method that uses the number of faces while Figures (c) and (d) are the first ones selected using the entropy recomputation strategy.*

8. Conclusions and Future Work

We have presented here a new method to automatically build an Image-Based model of a scene. In our case we have dealt with an indoor scene where the walking region is selected by the user. Note 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.

In order to select the reference views, 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 discretized. 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 threshold that can be set 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 on view-dependent illumination. We need to define a new measure that takes into account the amount of illumination information present in a view, and study if it is better to store it apart from the geometry (some kind of BRDF information) or if it is possible to reduce the number of images with some kind of principal components analysis, in a similar way as in surface light fields[33]. This will make our system more general to deal with any kind of materials.

Acknowledgments

The authors want to thank Karol Myszkowski and Takehiro Tawara for the model of the room and Jordi Freixenet from University of Girona for providing us with the code of region growing segmentation. Many thanks to Hartmut Schirmacher and Santi Thió for their helpful comments. This work was partially supported by TIC2001-2416-C03-01 from the Spanish government, and SGR2001-00296 from Catalan government.

References

1. R.E. Blahut. *Principles and Practice of Information Theory*. Addison-Wesley, Cambridge, MA, 1987. 1, 4, 5

2. T.M. Cover and J.A. Thomas. *Elements of Information Theory*. Wiley, 1991. 1, 4, 5

3. P.-P. Vázquez, M. Feixas, M. Sbert, and W. Heidrich. Viewpoint selection using viewpoint entropy. In T. Ertl, B. Girod, G. Greiner H. Niemann, and H.-P. Seidel, editors, *Proceedings of the Vision Modeling and Visualization Conference (VMV-01)*, pages 273–280, Stuttgart, November 21–23 2001. IOS Press, Amsterdam. 1, 3, 4

4. S. Fleishman, D. Cohen-Or, and D. Lischinski. Automatic camera placement for image-based modeling. *Computer Graphics Forum*, 19(2):101–110, Jun 2000. 2, 3, 6, 7, 9

5. 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. 2

6. 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. 2, 4

7. G. Dorme. *Study and Implementation of 3D Scenes Comprehension Techniques*. PhD thesis, Université de Limoges, 2001. In French. 2, 3

8. 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. 2

9. N.A. Massios and R.B. Fisher. A best next view selection algorithm incorporating a quality criterion. In *Proc.of the British Machine Vision Conference*, 1998. 2

10. B. Abidi. Automatic sensor placement. In *Proc. Intelligent Robots and Computer Vision: Algorithms, Techniques, Active Vision, and Materials Handling*, pages 387–398, 1995. 2

11. K. Klein and V. Sequeira. The view-cube: An efficient method of view planning for 3d modelling from range data. In *Proc. of the Workshop on the Application of Computer Vision*, pages 186–191, Palm Springs, CA, December 2000. 2

12. D.R. Roberts and A.D. Marshall. Viewpoint selection for complete surface coverage of three dimensional objects. In *Proc.of the British Machine Vision Conference*, 1998. 2

13. G.H. Tarbox and S.N. Gottschlich. Planning for complete sensor coverage in inspection. *Computer Vision and Image Understanding*, 61(1):84–111, may 1995. 2

14. J.P. Grossman and W.J. Dally. Point sample rendering. In George Drettakis and Nelson Max editors, editors, *Rendering Techniques'98*, pages 181–192. Springer-Verlag, 1998. 2

15. L. McMillan and G. Bishop. Plenoptic modeling: An image-based rendering system. *Proc. of SIGGRAPH 95*, pages 39–46, August 1995. 2

16. 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. 3, 7

17. 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. 3

18. 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). 3

19. T. Werner, T. Pajdla, V. Hlaváč, A. Leonardis, and M. Matousek. Selection of reference views for image-based representations. *Computing*, 68(2):163–180, March 2002. 3

20. J.-X. Chai, X. Tong, S.-C. Chan, and H.-Y. Shum. Plenoptic sampling. In *SIGGRAPH 2000, Computer Graphics Proceedings*, pages 307–318. ACM Press / ACM SIGGRAPH / Addison Wesley Longman, 2000. 3

21. J. Rigau, M. Feixas, and M. Sbert. Information theory point measures in a scene. Technical Report IIiA-00-08-RR, Institut d'Informàtica i Aplicacions, Universitat de Girona, Girona, Spain, 2000. 4

22. J. O'Rourke. *Art Gallery Theorems and Algorithms*. Oxford University Press, New York, 1987. 5, 7

23. J. Aitchison, C. Barceló-Vidal, J.A. Martín-Fernández, and V. Pawlowsky-Glahn. Logratio analysis and compositional distance. *Mathematical Geology*, 32(3):271–275, 2000. 5

24. P.-P. Vázquez, M. Feixas, M. Sbert, and W. Heidrich. Image-based modeling using viewpoint entropy. In J. Vince and R.A. Earnshaw, editors, *Advances in Modelling, Animation and Rendering (Proc. Computer Graphics International)*, pages 267–279. Springer, 2002. ISBN-1-85233-654-4. 5, 6

25. P.-P. Vázquez, M. Feixas, M. Sbert, and A. Llobet. Viewpoint entropy: A new tool for obtaining good views for molecules. In D.Ebert, P.Brunet, and I.Navazo, editors, *Data Visualisation 2002 (Eurographics/IEEE TCVG Symposium Proceedings)*. Eurographics/IEEE, May 27-29 2002. 6

26. P. Bekaert, F. Suykens de Laet, P. Peers, and V. Masselus. Renderpark: A test-bed system for global illumination. available under http://www.renderpark.be. 6, 8

27. S. Gumhold. Maximum entropy light source placement. In *Proceedings of the Visualization 2002 Conference*, pages 275–282. IEEE Computer Society Press, October 2002. 7

28. R. Shacked. Automatic lighting design using a perceptual quality metric. Master's thesis, School of Computer Science and Engineering, The Hebrew University of Jerusalem, Jerusalem, Israel, February 2001. Available from http://www.cs.huji.ac.il/ danix/ldesign/ShackedThesis.pdf. 7

29. P.-P. Vázquez and M. Sbert. Perception-based illumination information measurement and light source placement. *Lecture Notes in Computer Science*, 2669:306–316, May 2003. 7

30. R.M. Haralick and L.G. Shapiro. *Computer and Robot Vision, vol. 1*. Addison-Welsey: Reading, MA, 1992. 8

31. J.W. Shade, S.J. Gortler, L.-W. He, and R. Szeliski. Layered depth images. In M.F. Cohen, editor, *SIGGRAPH 98 Conference Proceedings*, Annual Conference Series, pages 231–242. ACM SIGGRAPH, Addison Wesley, July 1998. ISBN 0-89791-999-8. 8

32. S. Rusinkiewicz and M. 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. 8

33. 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 Kurt Akeley, editor, *SIGGRAPH 2000, Computer Graphics Proceedings*, Annual Conference Series, pages 287–296. ACM Press / ACM SIGGRAPH / Addison Wesley Longman, 2000. 10