A Perceptual Approach to Trimming Unstructured Lumigraphs

Yann Morvan*
Trinity College Dublin

Carol O'Sullivan†
Trinity College Dublin

Figure 1: *Exterior scene. Left: frame of DV footage. Right: a novel view rendered after trimming of the unstructured lumigraph.*

Abstract

We present a novel perceptual method to reduce the visual redundancy of unstructured lumigraphs, an image based representation designed for interactive rendering. We combine features of the unstructured lumigraph algorithm and image fidelity metrics to efficiently rank the perceptual impact of the removal of sub-regions of input views (*sub-views*). We use a greedy approach to estimate the order in which sub-views should be pruned to minimize perceptual degradation at each step. Renderings using varying numbers of sub-views can then be easily visualized with confidence that the retained sub-views are well chosen, thus facilitating the choice of how many to retain. The regions of the input views that are left are repacked into a texture atlas. Our method takes advantage of any scene geometry information available but only requires a very coarse approximation. We perform a user study to validate its behaviour, as well as investigate the impact of the choice of image fidelity metric. The three metrics considered fall in the physical, statistical and perceptual categories. The overall benefit of our method is the semi-automation of the view selection process, resulting in unstructured lumigraphs that are thriftier in texture memory use and faster to render. (*Note to reviewers: a video is available at http://isg.cs.tcd.ie/ymorvan/paper37.avi. The figure occupying the ninth page is intended to appear on a color plate.*)

CR Categories: I.3.0 [Computer Graphics]: General—Perceptually-based rendering;

Keywords: perceptually-based rendering, realistic rendering

1 Introduction

The creation of digital visual content has become a major activity in a broad range of industries. Huge numbers of person-hours are invested in related tasks, from modeling to de-noising of captured data. There has been significant research in computer graphics aimed at automating or even bypassing some of these tasks. In the field of rendering, one such direction is the image-based approach, which allows new views to be computed from available images and rudimentary geometric information. Perceptually adaptive graphics, albeit so far predominantly explored to achieve machine-hours savings, is another area of great potential.

Research in Image-Based Rendering (IBR) has mainly focused on achieving interactive frame rates at satisfactory visual quality while exploring the tradeoff between how finely the light field has to be sampled and how much geometric information to use. From an artist's point of view, all the proposed methods are difficult to leverage: on one hand, depth-map, dense correspondence or optic flow based techniques [Heigl et al. 1999] rely either on the availability of an accurate geometric model or on computer vision methods such as stereo matching or range scanners. These perform poorly on scenes that contain complex occlusion boundaries, large surfaces of uniform irradiance or highly reflective materials. On the other hand, methods that put the emphasis on a dense sampling of the light field are often based on a rigid parameterization that requires expensive and impractical gantries to constrain camera positions. In [Gortler et al. 1996], views can be acquired in an unstructured way, for example using a hand-held camera, but a lossy rebinning

*e-mail: morvany@cs.tcd.ie
†e-mail: osullica@cs.tcd.ie

pre-process is then needed.

Buehler *et al* [2001] propose their unstructured lumigraph rendering algorithm as a flexible method that bridges both ends of the light field sampling *vs* geometry compromise. It behaves similarly to light field rendering when given many views and a single plane, and like view dependent texture mapping [Debevec et al. 1998] when given fewer views but more detailed geometry. A major drawback of unstructured IBR techniques is that they cannot exploit any parameterization properties to guide compression, making it difficult to compress inter-view redundancy. Selecting a set of views that minimizes redundancy is therefore crucial.

Perceptually adaptive rendering is particularly well suited to global illumination, see the works of [Myszkowski et al. 1999], [Yee et al. 2001] and [Stokes et al. 2004] among others. One reason is that perceptual metrics, be they based on complex simulations of the Human Visual System (HVS) or on implicit models of its properties, tend to be computationally expensive, so that their own cost can counterbalance the savings they achieve (there exist works that take up this challenge in the context of interactive rendering: [Dumont et al. 2003], [Walter et al. 2002], [Williams et al. 2003]). Another reason is that intermediate results of global illumination algorithms that proceed by progressive refinement can be substituted for the gold standard that most perceptual metrics need.

In the context of unstructured IBR, view selection is a pre-process, which makes the use of perceptual metrics more tractable. Furthermore, the input images themselves can be used as a gold standard. The initial set of views then needs to be dense enough to sample the scene properly. This will typically be the case when working from video sequences.

This paper proposes a framework to facilitate the authoring of thrifty unstructured lumigraphs. We make use of image fidelity metrics as a criterion to remove redundancy from the original dataset. Three metrics are considered, respectively of a physical, statistical and perceptual nature. We perform a user study to investigate which metric performs best, as well as validate that our framework yields better results than the alternatives. The scope of this work is limited to static scenes under constant illumination.

2 Related work

Hlavac *et al* [1996] were the first to study the problem of view selection for unstructured IBR, more specifically in the context of view interpolation. They demonstrated their method in the case where camera positions are limited to one degree of freedom. It consists of growing view position intervals until the quality of the interpolation over them falls under a threshold, and then keeping the views at their bounds. They point out that the computational cost of the interval growing algorithm explodes in the general case.

In [Fleishman et al. 2000], the geometry of the scene is assumed to be known and its surfaces to be Lambertian. Thus, views can be selected without *a priori* knowledge of the corresponding images. A heuristic is presented to determine a reduced set of views that ensure coverage of all the scene polygons with quality superior to some user specified threshold. Vazquez *et al* [2001] propose a similar technique inspired by information theory, using viewpoint entropy to guide the selection process.

Schirmacher *et al* [1999] use a lumigraph representation to interactively render high quality global illumination solutions. Each frame of the solution being expensive to compute, they propose an iterative method to progressively add views to the lumigraph that are predicted to most increase rendering quality. Coombe *et al* [2005] introduce a system that lets an author interactively create a surface light field online by giving him feedback on what views to capture next. Views are incorporated on the fly into the light field using an online SVD algorithm. Their method pre-supposes a reasonably accurate geometric model of the captured scene.

The framework we propose is based on assumptions different from those made by these two last works: a dense set of views, along with their camera pose information, is available, but geometric information can be sparse and/or innaccurate. It therefore aims to prune visual information that contributes less to rendering quality. As such it is related to the various light field compression techniques that have been developed. Further discussion of this topic can be found in the works of Xin and Gray [Xin and Gray 2003]. Our approach puts more emphasis on facilitating the authoring process. The main advantage of our framework is that it transparently deals with the compromise between geometric accuracy and the light field sampling rate.

3 Proposed framework

Given a perceptual measurement tool that evaluates how similar an image is to a reference, it is straightforward to define the perceptual quality score (PQS) of a subset of input views. This is calculated by applying the measurement tool to pairs consisting of each initial input view and its reconstruction by the IBR algorithm using that subset, then taking the sum. From there, we can assess the perceptual degradation caused by the removal of an input view by taking the difference in perceptual quality score of renderings computed with and without it. The higher the degradation, the more view dependent information that input view captures, and the less redundant it is. For clarity, we will call the views used for the purpose of computing perceptual quality scores *touchstone views* (TV).

We show how the unstructured lumigraph rendering algorithm makes it easy to exploit spatial coherence to speed up the computation of the perceptual degradation caused by the removal of an input view. We then describe the perceptual measurement tools that we have chosen to use and justify our choice. The details of our greedy pruning process are then provided, followed by a description of how to leverage its results for rendering. Finally, we discuss some practical features of the technique.

3.1 Context: Unstructured Lumigraph Rendering

Buehler *et al*'s [2001] Unstructured Lumigraph Rendering (ULR) algorithm is a general purpose IBR technique that takes as input a polygon mesh approximating the geometry of the scene, a set of images of it and the camera pose information corresponding to each image. The polygon mesh is dubbed a *geometric proxy*. It is important that the registration between the geometric proxy and the input views be known.

The main principle of the technique is to compute a *blending field* that depicts how the color of each pixel of the desired view is to be obtained by blending the colors of the corresponding pixels in the input images.

Buehler *et al* design a continuous function that gives a high *blending weight* to views that see a given point of the geometric proxy with good resolution from an angle close to that from which it is seen in the desired view. This function only gives a non-null weight to a small number k of best views (in practice, they choose four). To achieve interactive frame rates, the blending weights for each input view are not evaluated at each pixel but linearly interpolated from evenly located sample points: the vertices of the triangulated geometric proxy.

Thus for a given touchstone view, it is possible to determine the list of triangles over which visual changes will happen when computing the perceptual degradation caused by the removal of an input view. It is simply the list of triangles that have at least one vertex whose set of k best views contains that input view. If that list is empty for all triangles in a given touchstone view, its contribution to the overall perceptual quality does not need to be updated. Moreover, if we are able to compute the perceptual measure locally, time can be saved by computing it only over triangles where a visual change has occured.

3.2 Perceptual measure of view utility

For our framework, we considered three image fidelity metrics: the traditional Root Mean Square error (RMS), Yee and Newman's [2004] PerceptualDiff (PDIFF) , and Wang *et al*'s [2002] Structural SIMilarity index (SSIM). We had considered including Daly's Visible Difference Predictor (VDP) [Daly 1993] using Mantiuk *et al*'s most recent implementation [Mantiuk et al. 2005], but its longer computation times (between one and two orders of magnitude) made our framework untractable in its current iteration. In this work, all the metrics considered were computed only on the luminance channel, thus disregarding chrominance information. This was motivated by computation time concerns, as well as the fact that for each of the metrics considered, color handling is an orthogonal addition that we thought better to investigate at a later point.

3.2.1 SSIM

Contrary to most perceptual metrics, the SSIM does not rely on a simulation of the low level behavior of the Human Visual System (HVS). It is based on the observation that the function of the HVS is to extract structural information from visual stimuli. It therefore estimates how similar two images are by using statistical tools to quantify the structural difference between them.

We justify our inclusion of SSIM as follows:

- As Wang *et al* show, when predicting the visual fidelity of a wide range of images, their method can compare favorably to RMS, peak signal to noise ratio, and more importantly, techniques based on models that reproduce the error sensitivity of the HVS.
- Spatial frequency masking is a HVS property accounted for by these latter techniques, as opposed to SSIM, but taking it into account is not clearly desirable for our purpose. This is because changes of the viewing context, like a change of viewpoint or the presence of an occluder between the image based rendered object and the observer, will cause arbitrary modifications of the visual information surrounding a point of the object.
- The SSIM index is less computationally intensive because it takes a statistical approach as opposed to a signal processing one, which requires complex transforms to be applied.
- It has little computational overhead, an important advantage if we want to evaluate it on many small triangles and not just a few whole images, as mentioned in section 3.1.
- It is straightforward to implement as a multi-pass fragment shader.

SSIM's evaluation is based on a sliding window mechanism and computes a score for each pixel (we invite the reader to consult Wang *et al*'s [2002] paper for further details). We implement it using OpenGL's fragment shader mechanism, taking advantage of its separability into horizontal and vertical passes. The value of the SSIM index over each triangle is obtained by taking its average over the relevant pixels.

3.2.2 PDIFF

Yee and Newman's [2004] (PDIFF), which they put forward in the context of production testing, is based on Ramasubramanian *et al*'s [1999] simplified version of the VDP. Like the VDP, it accounts for three features of the HVS: amplitude non-linearity, sensitivity variation as a function of spatial frequency, and visual masking. For efficiency purposes, the original VDP's decompositions of the signal into different bands in the frequency domain and different orientations are discarded. This allows for a purely spatial approach, based on Laplacian pyramids, at the cost of a much more rudimentary modeling of the visual masking phenomenon (because interactions between signal components based on frequency and orientation similarity are not considered). The behaviour of PDIFF depends on the field of view occupied by the signal and its resolution, which depend on the target viewing conditions: cinema theatre in Yee and Newman's case, desktop monitor in our user study.

In the context of our framework, the computation of the Laplacian pyramid proved a manageable overhead in the application of PDIFF to individual triangles. Unlike SSIM, which outputs a normalized score for each pixel, PDIFF's output consist of a number of pixels where the metric predicts viewers will perceive a difference. To obtain a score over a triangle to use within our framework, we took the ratio between the number of pixels predicted indistinguishable and the total number of pixels covered by the triangle.

3.3 Greedy sub-view selection

In order to take advantage of the locality of view-dependent phenomena - be they inherent to the scene, such as reflective surfaces, or caused by geometric proxy inaccuracies - our framework does not discard whole input views but sub-regions of them: The triangle ring surrounding each vertex of the geometric proxy is projected into each input view. The resulting image areas are treated as *sub-views* (thus a sub-view is identified by the input view it belongs to, and a vertex of the geometric proxy). This choice is justified by the nature of the ULR algorithm: it operates on vertices, whose list of blending weights only affect the neighbouring triangles.

For sub-views to be consistent from one primary view to the other, it is necessary to drop the view dependent triangulation step of the origial ULR algorithm. To maintain proper sampling of the blending field (*Cf.* 3.1), we simply subdivide the initial geometric proxy evenly. The only extra cost of this approach is that it will sample the blending field unnecessarily tightly where triangles of the proxy project to a small region of the desired view. In practise, we found that this was counterbalanced by the savings incurred by scrapping the constrained Delauney triangulation originaly performed for each frame.

We then greedily prune the sub-views in order of increasing perceptual impact: at each step, the discarded sub-view is the one whose removal causes the least perceptual quality degradation, as defined in section 3.2. Brief pseudocode is given in Algorithm 3.1.

Algorithm 3.1: SUB-VIEW ORDERING()

```
Compute the initial perceptual qualities with all sub-views in use
while there remain sub-views
do { for each remaining sub-view
     do { for each triangle in each touchstone view
          that it affects
          do { do { Compute the visual degradation
                    caused by its removal. }
          Compute the average visual degradation
          for this sub-view. }
     Append the sub-view whose removal causes the
     least degradation to the list of ordered sub-views.
     Remove that sub-view from the list of remaining
     sub-views. }
```

Some book-keeping is necessary to avoid recomputing the perceptual quality score over triangles that were not affected by the last sub-view removal. An array contains the initial PQS of each triangle for each touchstone view obtained when using every input view for rendering. It is accessed whenever the perceptual degradation over a triangle needs to be recomputed. We update the following data structures after each removal:

- A1 A cache containing the PQS degradation over each triangle in each touchstone view for each primary view to be potentially next removed.
- A2 A cache containing the next PQS degradation resulting from the potential removal of each sub-view.
- L3 The list of remaining sub-views.
- L4 The list of remaining input views ordered by ULR blending weight (*Cf.* 3.1) for each vertex in each touchstone view.
- L5 The sub-list of remaining input views that will affect the rendering of each triangle in each touchstone view.
- L6 The list of triangles (grouped by the touchstone view they belong to) to whose rendering each remaining sub-view contributes.

A1 only needs to be updated when the last removed sub-view "contained" that triangle. A2 is obtained by averaging the PQS degradations of all triangles of a sub-view over all relevant touchstone views. It gets updated if the removal of the last sub-view affected the PQS degradation of a relevant triangle in any touchstone view.

L3 is self explanatory. L4 is included in order to avoid having to go through all the remaining input views each time the blending weights are computed when rendering a triangle. Here, we have to note that the ULR algorithm is designed to approximate epipolar consistency. This means that if we were to evaluate the PQS on a touchstone view, while using that very touchstone view as an input view, it would be selected as first among the k best views and given the highest weight for each vertex. Removing other input views would then cause the PQS to increase all the more if their weight was important, because the relative weight of the touchstone view itself would be increasing. Such behavior is the opposite of what we want to measure. To correct it, for each vertex of a given touchstone view of L4, we initially remove that touchstone view from the list of potential input views. This ensures that each touchstone view is not used by the ULR algorithm when rendering a triangle with the purpose of comparing it with its appearance in that specific touchstone view.

L5 is built from L4 by taking the union of the first $k+1$ input views over the vertices of each triangle. Indeed, to compute the new PQS over a triangle resulting from the potential removal of each single input view that affects it, we need to render the triangle without that view, and therefore need the $(k+1)^{th}$ input view that will fill the gap for each vertex where the removed view was present.

When a sub-view is selected for removal, other sub-views will make their way onto the list of k best views for each vertex in each touchstone view it affected. This has an impact on which touchstone views to render on the next step to evaluate the perceptual degradation caused by the potential removal of each remaining sub-view. L6 stores that information and it is therefore updated by identifying which input view "filled the gap" left by the removal of the last sub-view for each affected triangle in each touchstone view.

The output of the algorithm is a list of all initial sub-views sorted by the estimated order in which they should be removed to minimize perceptual degradation.

3.4 Texture atlas generation

The ULR algorithm uses hardware accelerated projective texture mapping and color blending to achieve interactive frame rates. To capitalize on the visual redundancy removal achieved by pruning sub-views, input views (now with holes in them) have to be repacked into a texture atlas to optimize the use of texture memory.

The texture coordinates of each triangle in each input view can be recovered from the hardware's automatic texture generation mechanism. We adapt Hale's [Hale 1998] triangle packing heuristic to our needs. It begins by rotating triangles into "mountain" shapes and then mirrors them horizontally or vertically to tightly fill image rows of decreasing height. Since we are taking triangles from many input views, without modification this method would lead to the big triangles being packed in the first textures and the small ones in the last textures, regardless of the input view they belong to. Thus, to render a view from a certain viewpoint, many more textures than necessary would need to be in texture memory. We therefore modify Hale's height sorting with a condition on the proximity of the input views' camera centers.

3.5 Touchstone views selection

The use of our framework raises the question of how to choose the touchstone views. It is an authoring choice that depends on the final purpose of the representation. In graphical applications, image based representations are particularly well suited to mid-range visual content: content that is not far enough away to be rendered as a static billboard, but yet not close enough for the user to interact with. The ULR algorithm is better suited for this than most IBR techniques because of the ease with which it can be incorporated into a typical hardware accelerated 3D engine.

In this context, parts of the captured objects will end up being occluded by closer range objects when viewed from the area that users can navigate. Apart from the obvious choice of limiting the set of touchstone views to the navigable viewing region, our method lets the author leverage these known occlusions. By rendering the occluders in each touchstone view, areas that are irrelevant to visual quality (because ultimately unseen) can be masked out, as allowed by our implementations of SSIM and PDIFF.

If the artist is given information on where users are likely to spend the most time, and which are the most likely viewing angles, he can adjust the density of touchstone views accordingly. He can either discard or enforce high frequency phenomena, such as a glare in

a window, by discarding or choosing to keep the touchstone views that exhibit it.

The choice of touchstone views is related to the behavior of the perceptual measurement tool: In our experience, the quality of the registrations obtained with commercial tracking software was good enough for the tools we chose to behave well. However, for a scene where some views are not properly registered, any perceptual metric with a registration requirement acts like a double-edged sword: If a badly registered view is used as a touchtone view, it will upset the behaviour of the framework for sub-views taken from a neighbouring viewpoint. If on the other hand the badly registered view is excluded from the set of touchstone views, its sub-views will be automatically discarded by the framework, as their removal will increase rendering quality.

Being part of the authoring process, the choice of touchstone views is very much case dependent. To evaluate the potential of our approach, we chose to place ourselves in the neutral case where all views are equally desirable. Thus, we retain one input view out of every two as touchstone view, in an even distribution.

4 Results

We acquired four video sequences of different scenes using a handheld Canon XL1 PAL DV camcorder. Camera poses were recovered using PFTrack 3.0, a commercial camera tracking solution from The Pixel Farm. Rough geometric proxies were created in a few dozen minutes with a standard polygon editing tool, using six to ten reconstructed features and world-space orientation from PFTrack as construction guides. Since our method bypasses the view-dependent image-plane triangulation of the original ULR algorithm, our proxies need to have unit depth complexity when seen from the viewing region of interest. Statistics for each unstructured lumigraph thus created are summarized in Table 1. Views of these scenes are shown in Figure 1 and 5.

4.1 User study

We designed and ran an experiment to evaluate the results of our framework.

We first wished to measure how much better our view selection technique is to currently available alternatives. As discussed in the related work section, without accurate geometric knowledge of the scene and when cameras can be arbitrarily placed, current alternatives are limited to techniques that ensure a uniform coverage of the geometric proxy by the retained views. We therefore picked two test scenes where uniform coverage could easily be enforced for any number of discarded sub-views: their initial views were obtained by evenly trucking the camera respectively in front of a building facade (*Cf.* Figure 1) and in the university library (*Cf.* Figure 5). In this case, uniform coverage for any number of remaining views can be maintained by picking them evenly from the initial set. Since our technique operates at the sub-view level, for a given number sv of discarded sub-views, we estimated the number v of views to discard by dividing sv by the average number of sub-views contained in each view. Once all the sub-views contained in the evenly picked v views were discarded, we made up the difference in the following fashion: remaining views were browsed in sequence, removing (or recovering if more sub-views than needed had been discarded) a single sub-view from each. To ensure the even distribution of removed (recovered) sub-views, we used a vertex coloring of the geometric proxy and picked sub-views of the same color, moving on to the next color when one was exhausted. This heuristic, including the graph coloring, took less than a minute to apply to each test scene. We will call it *regular* in the remainder of the paper.

Next we wanted to know how different the results of our framework would be depending on the metric used, PDIFF, SSIM or RMS.

The goal of the experiment was therefore to measure, for each of the four sub-view discarding strategies (three metrics plus *regular*), at which number of discarded sub-views users started to perceive visual degradation compared to the original dataset. We chose a double-random staircase experiment design. In a staircase experiment, users are presented with a sequence of stimuli at different levels of intensity. They are asked to perform a two-alternative forced choice at each step. The sequence of stimulus levels is influenced by the behaviour of the user: the level is regularly updated in one direction while the user keeps making the same choice, and direction changes when the current choice contradicts the last one (this is called a reversal). Ascending staircases start with the lowest stimulus level and their initial stimulus update direction is up. It is the reverse for descending staircases. Double-random staircase designs randomly interleave one staircase of each kind.

In our case, the stimuli presented are pairs of short (two seconds) rendered video sequences of the test scene, one of which uses the original dataset, shown one after the other. The number of sub-views discarded before rendering the second video corresponds to the stimulus strength. The forced choice consists of deciding whether the videos are of the same or different quality.

Asking participants to compare the quality of two full size PAL videos of short duration is problematic because it is difficult to control where they focus their attention. For this reason, we produced four sets of smaller videos showing two sub-regions of each test scene. For the *Exterior* scene, the sub-regions consisted of the windows plus the tree branch (*window*), and the door plus the tree trunk (*door*). For the *Library* scene, they consisted of a section of the upper glass barrier (*glass*), and a section of the upper book stacks (*stacks*). The novel camera paths used to produce the videos were created ad-hoc. The same path was used for both sub-regions of a test scene. The experiment was setup so that the displayed video for each sub-region subtended roughly between 4^0 and 5^0.

The full experiment thus consisted of 32 staircases: 4 sub-view discarding strategies $\times$ 4 test sub-regions $\times$ 2 staircases (one ascending, one descending), all randomly interleaved to counteract learning and expectation effects. Each staircase was limited to 8 reversals. We chose 3200 sub-views as the initial increment (decrement) value of the stimulus level, which was lowered to 1600 upon the third reversal and 800 upong the sixth. A few pilot runs of the experiments yielded an average duration of over an hour. We judged this too long for the participants to maintain concentration, thus we split the experiments in two sessions of under 40 minutes, asking participants to come back later during the day for the second session (break times varied from 40 minutes to 2 hours). Each session consisted of the staircases corresponding to one sub-region of each scene. Which session each participant sat through first was randomized. 16 participants took part within a controlled setup (same computer, display device and viewing conditions). All were from the computer science department (3 staff, 13 students), seven were women, all had normal or corrected to normal eyesight.

By fitting a psychometric function to their responses, we estimated the Point of Subjective Equality (PSE) i.e. the number of discarded sub-views at which participants had an even chance of reporting some visual degradation. Results are summarized in Figure 2. Results for the *door* sub-region were discarded because staircases failed to converge in most cases. We attribute that fact to the presence of a slight popping artefact in the video obtained with the original dataset, which could have confused participants.

We performed a two-factor analysis of variance (ANOVA) with replication on the results. It shows a significant main effect of

	Views	Vertices	Faces	Sub-views
Exterior	143	268	491	18638
Library	91	244	434	15625
Mezzanine	68	444	808	14508
Objects	175	119	214	20674

Table 1: *Test scenes statistics (note that we do not count sub-views that are not blendable for lack of visibility of the whole triangle ring)*

the view discarding strategy factor ($F(3, 180) = 15.48, p \approx 0$), a significant main effect of the sub-region factor ($F(3, 180) = 71.24, p \approx 0$), and a significant interaction between the two factors ($F(6, 180) = 36.48, p \approx 0$). Concerning the sub-region factor, it is natural that a given metric yields different numbers of discarded sub-views at the PSE on different sub-regions, as those exhibit varying degrees of view-dependency. The strong significance of the interaction effect means that the relative performances with respect to each other of the four sub-view discarding strategies depend on the content.

Post-hoc analysis was then performed using a standard Newman-Keuls test for pairwise comparisons among means. Regarding the discarding strategy factor, there were only two cases where two strategies' outcomes were not significantly different. One was between SSIM and *regular* on the *window* sub-region ($p = 0.057$). This can be attributed to the much higher variance in the PSEs resulting from the *regular* discarding strategy: participants disagreed more with each other as to when degradations started appearing with this strategy than they did with our framework (using any metric). The other was between SSIM and PDIFF on the *stacks* sub-region ($p = 0.052$), providing the exception to the rule that each metric used in our framework yielded statistically different results.

A reading of the chart in Figure 2 shows that both SSIM and PDIFF yield results that are consistently better than those of the *regular* discarding strategy, with a slight overall advantage for PDIFF. RMS's performance is very erratic: it is by far the worst strategy for both the *glass* and *stacks* sub-regions, yet strongly outperforms the competition for the *window* sub-region. This could be explained by the fact that RMS is a very one dimensional metric compared to PDIFF or SSIM. In particular, its extremely local focus (pixel difference) makes it very sensitive to noise and blur. This would explain why it performs well on the *window* sub-region, which contains complex patterns of intertwined small branches in the forefront, whose inaccurate fit with the geometric proxy yields strong blurring as sub-views are discarded.

In Figure 3 we plot the aggregated psychometric function over all participants for each sub-region. Those plots let us compare the discarding strategies at different levels of probability that participants will spot visual degradations. The steepness of each curve at its point of inflection reflects how consistent participants were with themselves in reporting when they started noticing degradation: the steeper, the more consistent and the more predictive the psychometric function. In this respect, the *regular* discarding strategy caused the most confusion in participants, while PDIFF yielded the best results. Interestingly, this means that the lower the desired probability of detection, the better PDIFF compares to SSIM. Thus, in the case of the *stacks* sub-region, where SSIM performed better at the PSE, PDIFF overtakes it for detection probabilities below 37%. In the context of visual content authoring, those are the probability levels that matter.

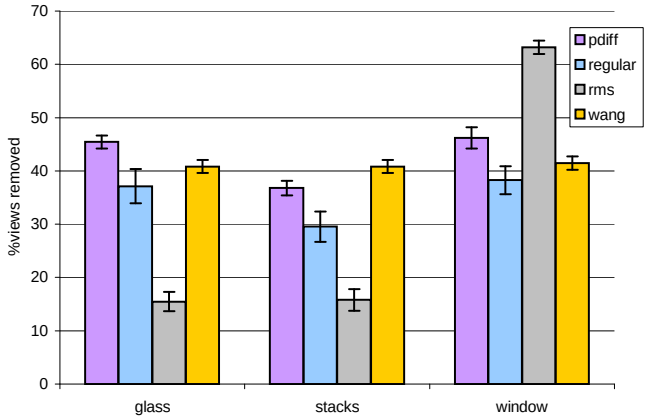

Figure 2: *Proportion of sub-views discarded for each strategy for each test sub-region at the* PSE *(standard errors are shown).*

	Exterior	Library	Mezzanine	Objects
RMS	≈ 5 h	≈ 5 h	≈ 6 h	≈ 5 h
SSIM	≈ 6 h	≈ 5 h	≈ 7 h	≈ 6 h
PDIFF	≈ 8 h	≈ 7 h	≈ 9 h	≈ 8 h

Table 2: *Approximate time taken to order all sub-views by increasing perceptual importance (hours).*

4.2 Impact on rendering performance

The sub-views of each unstructured lumigraph were perceptually ordered by our greedy pruning process with the PDIFF metric. Processing times are given in Table 2. We then navigated the unstructured lumigraphs while changing the number of retained sub-views within the ordered list, allowing us to determine the lowest number before degradations became noticeable.

Resulting texture memory footprints and framerates are summarized in Table 4. They vary significantly because of the wide differences between the properties of the test scenes. The Mezzanine scene's geometric proxy is more detailed and fits the actual geometry better. The Library contains glass surfaces that introduce strong view-dependence. In the Exterior scene, the geometric proxy matches the branches of the tree very poorly. This illustrates the importance of a human operator to make a judgement call on the data/quality compromise - a call that our technique facilitates. Indeed, our use of fidelity metrics lets us compare the importance of sub-views relative to each other, but does not provide an objective assessment of the quality of the dataset at a given point during the pruning process. The high texture memory savings obtained for the Objects scene is partly explained by the fact that its geometric proxy does not fill the whole view frustum in most views: unused areas are not packed in the texture atlas.

Two interesting quantities can be used to measure the success of our approach. First is the texture memory ratio between the original lumigraph and its trimmed version. Second, the ratio of blendable views per vertex. Indeed, the bottleneck of the ULR technique lies in the computation of the blending weights, as the algorithm has to order each candidate blendable view using a penalty function, before renormalizing the weights over the k best views. The ratio of blendable views per vertex is therefore a good hint at the speed-up that any ULR implementation can expect thanks to our technique. The framerates quoted in Table 4 illustrate this point: as expected, the rendering speed-up correlates with the number of vertices of the geometric proxy. Table 5 contains both ratios.

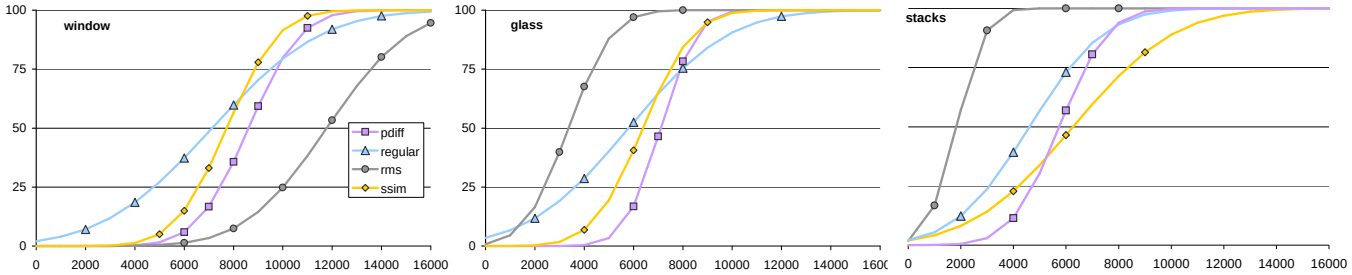

Figure 3: *Psychometric functions aggregated over the* 16 *participants for each sub-region. The Y axis represents the probability of a participant reporting some visual degradation. The X axis shows the number of sub-views discarded.*

Exterior	Library	Mezzanine	Objects
11000	11500	3000	10000
59 %	73.6 %	20.7 %	48.4 %

Table 3: *Number of retained sub-views for each test scene.*

	Size (Megabytes)		Frames per second	
	Original	Trimmed	Original	Trimmed
Exterior	164.3	90.9	9.5	11.5
Library	104.5	85.7	10.5	11.5
Mezzanine	78.1	10.5	7	11
Objects	201	29.25	NA	12

Table 4: *Results. "Size" stands for total texture memory footprint. Frames per second are measured on a 3GHz Pentium 4 computer equiped with an ATI Radeon 9800 XT graphics card. The NA rating means that there was no sustainable framerate due to texture swapping.*

Figure 4: *An input view of the Exterior scence after processing: the geometric proxy is drawn in white. Triangles in black belong to vertex rings that contain a triangle that is not fully visible in that view, which makes them unusable in the first place. Triangles in red belong to sub-views that were discarded by our technique.*

Figure 4 illustrates the behaviour of our technique on an input view. In this test scene, the proxy is fitting the ground and the wall, but is jutting towards the viewer at the tree's trunk and branches. Our technique correctly discards sub-views (in red) where little view dependence is present, i.e. where the proxy is modeling the scene properly and the surface there does not exhibit view dependent phenomena. Comparisons of renderings of the three other scenes before and after discarding sub-views are shown in Figure 5.

5 Discussion and future work

Our method has shown great potential on our test scenes, resulting in significantly better results than regular discarding when using either PDIFF or SSIM, and this consistently over sub-regions of varying nature. Once it has ordered the initial sub-views by increasing perceptual importance, an artist can quickly explore renderings of the scene using varying numbers of remaining sub-views until he is satisfied with the compromise between visual quality and resource consumption (memory as well as rendering time).

When it comes to comparing the performance of different metrics within our framework, Yee and Newman's PerceptualDiff appears to win. As acknowledged by Wang *et al* in their citation of the research conducted within the Modelfest framework and by the Video Quality Expert Group, debate is strong in the field of image fidelity metrics. It would perhaps be worthwhile to investigate ways to predict which metric is most appropriate depending on the type of content, both generally and locally.

	Exterior	Library	Mezzanine	Objects
Texture	55.3 %	82 %	13.5 %	14.5 %
Views/vertex	55.5 %	71 %	22.5 %	48.4 %

Table 5: *Results expressed in terms of texture memory usage ratio and number of candidate views per vertex ratio.*

As mentioned in the introduction, the closest techniques with which to compare our work deal with interactive rendering from compressed lightfields. Be they based on vector quantization, DCT or wavelets, they typically achieve higher memory savings than what we obtain, thanks to their much finer granularity. The most dramatic results are however obtained with stronger requirements about the geometric information than we make. From the point of view of rendering performance, typical numbers quoted in the literature hover around 8 frames per second [Xin and Gray 2003]. Since we chose to implement our framework in Haskell for the prototyping ease it provides, the performance figures we obtain are probably not representative of a heavily optimized ULR implementation. However, at roughly 7 to 15 fps, our approach already appears to be competitive. There are less easily quantifiable factors to consider in the comparison. One is the flexibility of the ULR representation, which results in better authoring convenience: the artist has more freedom to choose a geometry *vs.* sampling compromise, and he is not bound by sampling regularity, which light field techniques tend

to enforce rigidly. Another is its higher suitability for inclusion in a 3D engine. A formal comparison of the techniques would be very worthwhile, but it is a challenge because of the slight variations in requirements and features that make the choice of test scenarios that are suitable across the board difficult.

In its current implementation, the main drawback of our method is its computation time, considering that the number of input views can grow considerably with the area of the viewing region that the artist wishes to cover. To combat this, a statistical approach could be used at each removal step to avoid considering all remaining subviews, and only a sub-set of them. It is also possible to considerably reduce computation time by playing with the number of touchstone views considered when applying the perceptual metric. This is an area of future experimentation as the choice of touchstone views, or possibly their weighting by importance, would be a natural way for an artist to tune view selection to areas of particular importance in the viewing region from which the scene is intended to be seen.

Another limitation is our use of a view-independent geometric proxy, which aggravates a limitation of the ULR algorithm mentioned in Figure 4, namely that views that do not cover a triangle entirely cannot be used to texture it. This limitation can however be worked around during authoring by either taking wider angle pictures of the desired scene or panning the camera. Our framework ensures that the useless visual information around the edges will be culled in the final packing.

References

BUEHLER, C., BOSSE, M., MCMILLAN, L., GORTLER, S., AND COHEN, M. 2001. Unstructured lumigraph rendering. In *Computer Graphics (SIGGRAPH 01)*, 425–432.

COOMBE, G., HANTAK, C., LASTRA, A., AND GRZESZCZUK, R. 2005. Online construction of surface light fields. In *Rendering Techniques*, 83–90.

DALY, S. 1993. The visible differences predictor: an algorithm for the assessment of image fidelity. In *Digital images and human vision*, 179–206.

DEBEVEC, P., BORSHUKOV, G., AND YU, Y. 1998. Efficient view-dependent image-based rendering with projective texture mapping. In *Rendering Techniques 98 Proc. Eurographics Workshop on Rendering*.

DUMONT, R., PELLACINI, F., AND FERWERDA, J. A. 2003. Perceptually-driven decision theory for interactive realistic rendering. *ACM Transactions on Graphics 22*, 2, 152–181.

FLEISHMAN, S., COHEN-OR, D., AND LISCHINSKI, D. 2000. Automatic camera placement for image-based modeling. *Computer Graphics Forum 19*, 2, 101–110.

GORTLER, S., GRZESZCZUK, R., SZELISKI, R., AND COHEN, M. 1996. The lumigraph. In *Computer Graphics Proceedings, Annual Conferences Series, SIGGRAPH'96*, 43–54.

HALE, J. G. 1998. *Texture re-mapping for decimated polygonal meshes*. Edinburgh University.

HEIGL, B., KOCH, R., POLLEFEYS, M., DENZLER, J., AND GOOL, L. J. V. 1999. Plenoptic modeling and rendering from image sequences taken by hand-held camera. In *DAGM-Symposium*, 94–101.

HLAVAC, V., LEONARDIS, A., AND WERNER, T. 1996. Automatic selection of reference views for image-based scene representations. In *Proc. ECCV*, 526–535.

MANTIUK, R., DALY, S., MYSZKOWSKI, K., AND SEIDEL, H.-P. 2005. Predicting visible differences in high dynamic range images - model and its calibration. In *Human Vision and Electronic Imaging X, IS&T/SPIE's 17th Annual Symposium on Electronic Imaging (2005)*, B. E. Rogowitz, T. N. Pappas, and S. J. Daly, Eds., vol. 5666, 204–214.

MYSZKOWSKI, K., ROKITA, P., AND TAWARA, T. 1999. Perceptually-informed accelerated rendering of high quality walkthrough sequences. In *Proceedings of the Tenth Eurographics Workshop on Rendering*, 5–18.

RAMASUBRAMANIAN, M., PATTANAIK, S. N., AND GREENBERG, D. P. 1999. A perceptually based physical error metric for realistic image synthesis. In *Proceedings of ACM SIGGRAPH 1999*, ACM Press / ACM SIGGRAPH, 73–82.

SCHIRMACHER, H., HEIDRICH, W., AND SEIDEL, H.-P. 1999. Adaptive acquisition of lumigraphs from synthetic scenes. In *Computer Graphics Forum (Eurographics '99)*, The Eurographics Association and Blackwell Publishers, P. Brunet and R. Scopigno, Eds., vol. 18(3), 151–160.

STOKES, W. A., FERWERDA, J. A., WALTER, B., AND GREENBERG, D. P. 2004. Perceptual illumination components: A new approach to efficient, high quality global illumination rendering. In *ACM SIGGRAPH conference proceedings*, ACM Press.

VAZQUEZ, P., FEIXAS, M., SBERT, M., AND HEIDRICH, W. 2001. Viewpoint selection using viewpoint entropy. In *Proceedings of the Vision Modeling and Visualization Conference (VMV01)*, IOS Press, Amsterdam, 273–280.

WALTER, B., GREENBERG, D. P., AND PATTANAIK, S. N. 2002. Using perceptual texture masking for efficient image synthesis. In *Eurographics Computer Graphics Forum*, vol. 21.

WANG, Z., BOVIK, A. C., AND LU, L. 2002. Why is image quality assessment so difficult? In *Proceedings of the IEEE International Conference on Acoustics, Speech, & Signal Processing*, vol. 4, 3313–3316.

WILLIAMS, N., LUEBKE, D., COHEN, J., KELLEY, M., AND SCHUBERT, B. 2003. Perceptually guided simplification of lit, textured meshes. In *Proceedings of the 2003 ACM SIGGRAPH Symposium on Interactive 3D Graphics*.

XIN, T., AND GRAY, R. 2003. Interactive rendering from compressed light fields. *IEEE Transactions on Circuits and Systems for Video Technology 13*, 11, 1080–1091.

YEE, Y. H., AND NEWMAN, A. 2004. A perceptual metric for production testing. In *SIGGRAPH '04: ACM SIGGRAPH 2004 Sketches*, ACM Press, New York, NY, USA, 121.

YEE, H., PATTANAIK, S., AND GREENBERG, D. P. 2001. Spatiotemporal sensitivity and visual attention for efficient rendering of dynamic environments. *ACM Trans. Graph. 20*, 1, 39–65.

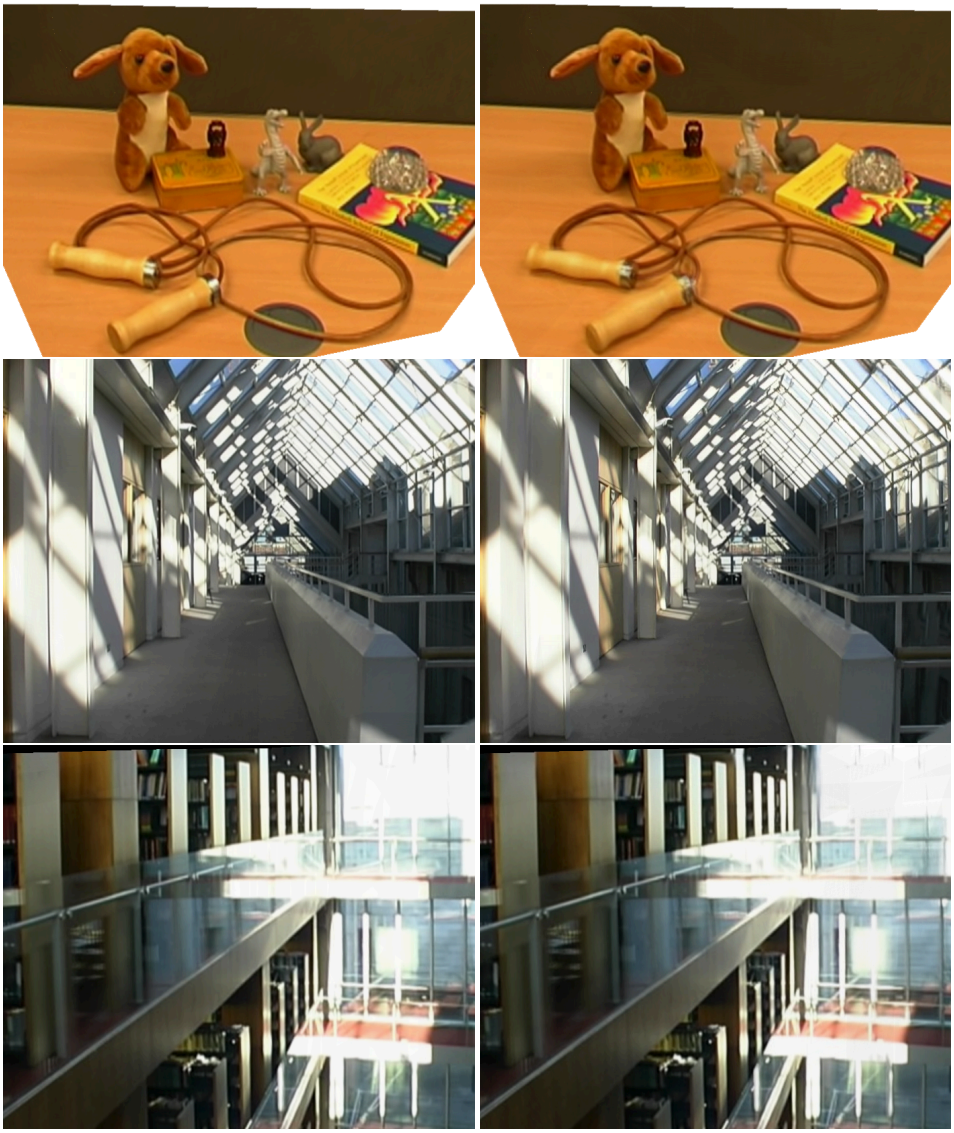

Figure 5: *Novel views of some test scenes, from top to bottom: Objects, Mezzanine and Library. On the left: original dataset. On the right: trimmed dataset.*