Bandwidth Reduction for Remote Navigation Systems through View Prediction and Progressive Transmission

Pere-Pau Vázquez [a,*,1] Mateu Sbert [b]

[a]*Dept. LSI - Universitat Politècnica de Catalunya*

[b]*IIiA, Universitat de Girona*

Abstract

In this paper we explore a set of techniques to reduce the bandwidth required by remote navigation systems. These systems, such as exploration of virtual 3D worlds or remote surgery, usually require higher bandwidth than the Internet connection commonly available at home.

Our system consists of a client PC equipped with a graphics card, and a remote high-end server. The server hosts the remote environment and does the actual rendering of the scenes for several clients, and the new image is passed to them. This scheme is suitable when the data has a copyright or when its size may exceed the rendering capabilities of the client. The general scheme is the following: each time the position changes, the new view is predicted by both the client and the server and the difference information between the predicted view and the correct one is sent to the client. To reduce bandwidth two kind of improvements can be made: improve the prediction method, and improve the transmission system. We present here two groups of techniques: First, a set of lossless methods which achieve reductions of up to a 9:1 ratio. These are a combination of a two-level forward warping, that takes advantage of spatial coherence, and a masking method, which allows to transmit only the information that really needs to be updated. Second, a set of lossy methods suitable for very low bandwidth environments which involve both progressive transmission and image reuse. They consider relevant parameters such as the number of pixels, the amount of information they provide, and their colour deviation in order to create a strategy for prioritizing the information transmission. This system allows to improve up to an additional 4:1 ratio. The quality of the generated images is very high, and often indistinguishable from the correct ones. These techniques can be applied to head-mounted displays or any remote navigation software.

Key words: Bandwidth reduction; Remote navigation; Image-based rendering; Information theory.

---

1 Introduction

Recent advances in computer science have allowed dramatic breakthroughs at the beginning of the new century, such as remote computer-assisted surgery, 3D gaming, or the creation of compelling special effects in films. However, there are still some problems to be solved to obtain attractive applications such as 3D remote exploration or gaming. One of the most important problems comes from the limited bandwidth of common internet connections. Many real time rendering systems require the transmission of images at 15-30 frames per second to guarantee a continuous sensation of movement. Moreover, these frame rates can only be obtained with high-end systems, despite the latest chipset releases of graphics hardware.

In this paper we present some techniques that can help to reduce the necessary bandwidth suitable for low bandwidth scenarios. In a first step, we improve the client-server navigation system by exploiting spatial coherence through the use of Image-Based Rendering techniques and splatting. The server encodes the difference pixels in a bitmap, and transmits the bitmap and the difference pixels, instead of sending the whole difference image. With this method the bandwidth needed to send the information through the network reduces up to 9:1 ratio. In a second step, we focus on very low bandwidth environments. We take advantage of temporal coherence by reusing previously computed frames. Moreover, a progressive transmission scheme is designed in order to prioritize the correction of the more noticeable artifacts in the predicted image. This allows an extra 4:1 reduction of bandwidth with almost no loss in image quality. The increase in computing cost can be perfectly assumed by the client processor, as the only required operation is a forward warping plus an image update. These techniques can be applied to head-mounted displays or any remote navigation software, and can be used either to reduce bandwidth or to compensate network latencies.

The remainder of the paper is organized as follows: in Section 2 we review the previous work, in Section 3 we present our techniques for lossless bandwidth reduction, in Section 4 we present some techniques which improve the previous results with frame reuse and progressive transmission. Finally, in Section 5 the results are discussed and we analyze possible lines of future work.

---

[*] Corresponding author.
Email address: `ppau@lsi.upc.es` (Pere-Pau Vázquez).
[1] Present Address: Dept. LSI Polytechnical University of Catalonia, Campus Sud - Ed. ETSEIB, Diagonal 647, 8ena Planta, E - 08028 Barcelona, Spain.

2 Previous Work

During the second half of the last decade, a set of new rendering methods, called Image-Based Rendering, have been developed. All of them have in common the use of images in some stage to partly or completely substitute the geometry of the scene [1–5]. The use of precomputed images simplifies rendering algorithms thus obtaining realistic effects at low cost. Although most of these methods present huge memory requirements and are therefore not suitable for dealing with very complex scenes in a low-end PC, some of these techniques can be used to accelerate rendering or to compensate for network latencies in remote navigation tasks. Mark *et al* [5] use two images generated in a server and warp them together to generate new views at interactive frame rates. Some commercial products such as QuickTime VR [4] offer panoramas over the network which are divided into pieces, so that the client can view them before they are totally downloaded, however, a large portion of the panorama has to be received before it may be viewed. Moreover, these systems only allow for camera rotation and zooming, and it is difficult to extend them to handle translations.

Our approach is a set of techniques to reduce bandwidth which involve image compression and image-based rendering. A cooperative scheme is described in [6], where the server generates a high-quality rendering and a low-quality one and subtracts them, sending the result to the client. The client generates the low-quality image and adds the one sent by the server. This way the author obtains better compression and avoids the low quality of JPEG scheme in edges and smooth shading. On the other hand, this method requires that the geometry is also sent to the client in order to allow it to generate the low-quality image. Biermann *et al* [7] use a similar collaborative scheme. To reduce bandwidth, the client performs a forward warping to predict the new view, the server does the same and also computes the correct view. Then, the predicted view is subtracted from the exact one and the difference view is compressed. This way they achieve reduction ratios of 55% on small movements of the camera. Cohen-Or *et al* [8] use a compression scheme which reduces up to one order of magnitude the bandwidth, compared to a MPEG postrendered sequence with equivalent quality. However, in order to exploit spatial coherence to improve compression ratios, their compression scheme requires that the client has access to the geometric data (this also happens in [6]), which is not possible in many scenarios due to copyright aspects.

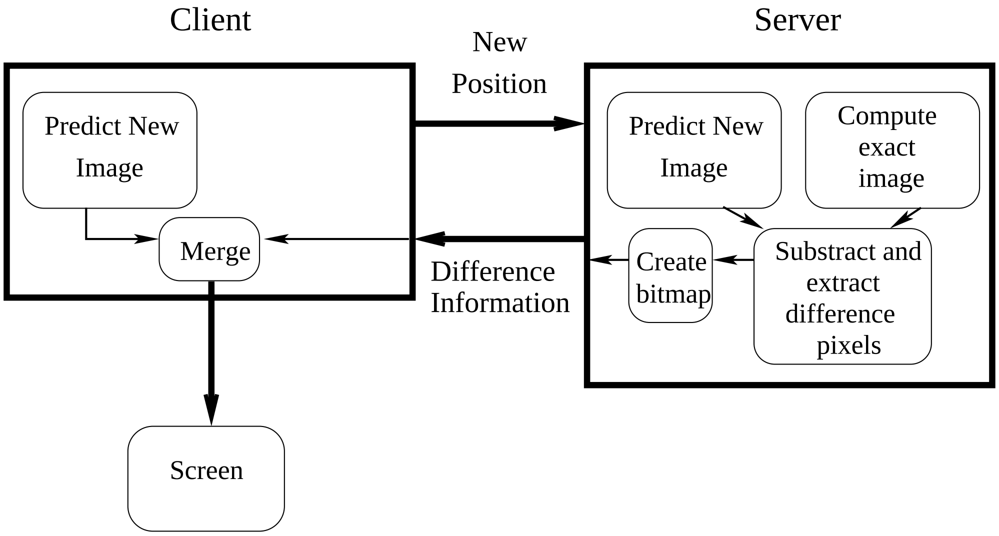


Fig. 1. The client-server remote navigation system. Instead of sending every frame through the network the server only sends the difference pixels, together with a bitmap which indicates the pixels to be modified. The client predicts the new view using a two-level forward warping.

3 Lossless Bandwidth Reduction Techniques

In this Section we present two bandwidth reduction techniques, a two-level forward warping and a compression method which avoids sending information corresponding to correctly predicted pixels to the client. We use a similar architecture to the one presented in [6], but improving the tasks on the client and on the server side. Figure 1 depicts this architecture and the tasks. The client predicting system is enhanced by adding a simple and low cost two-level of detail forward warping. The server side reduces bandwidth by only transmitting the pixels to be updated, not the whole image (with the aid of a bitmap that encodes the pixels that have to be changed). This way we achieve a reduction of bandwidth of 5 to 1 over the previous method in [7] for rotations of up to 30 degrees and a reduction of up to 9 to 1 for translations.

3.1 Two-Level Forward Warping

In a similar way to Biermann *et al* [7] we use a forward warping to predict the next image. However, after a rotation or translation movement, holes might appear (see Figure 2a). To avoid this, we will exploit spatial coherence. Some possible ways to fill holes are to create a mip-map of the resulting image (after warping, and using only the projected points, not the void regions) and fill in the gaps, or to compute correct splat sizes for each point. On the other hand, these techniques can be quite costly and we want a very cheap method. Our system performs a two-level forward warping of the valid pixels in previous image, as shown in Figure 2*a* . The first one consists into a reprojection of each point using the OpenGL [9] primitive *GL_POINT* assigning to the points a size (*GL_SIZE*) of four. Then, the depth buffer is cleared and the points reprojected again with a size of one. As some of the present gaps should have the same colour as closer pixels, this method reduces the visually unpleasant holes, and the number of correctly predicted pixels increase by a factor of up to a 40% percent, and an average of a 20% percent (bandwidth reduction of 1.25:1 ratio) in rotations. We remark that this method is not actually new in the sense that it can be considered a particular case of splatting, but it is cheap and only uses graphics hardware commonly available in most personal computers. Figure 2*b* depicts the difference in number of wrong pixels for simple and two-level warping strategies for different rotations. Note that the number of wrong pixels in the second case grows slower and the percentage of bandwidth reduction is almost constant in rotations of up to 30 degrees. We have also tested our method for translation transformations, in this case, as the pixels are reusable longer, the amount of correct pixels grows up to the 68% and an average of 60% (which represents a bandwidth reduction of 2.5 to 1). These results are depicted in Figure 2*c*. In Figure 3 we can see an example of rotation (ten degrees) and an example of translation (five steps in world coordinates).Figures 3*a* and *e* have been created by only warping the pixels of the previous image, while 3*b* and 3*f* have been created with our two-level forward warping, the amount of difference pixels has been reduced in a 20 percent. Moreover, images 3*b* or 3*f* can be used if latency is high as they do not present as many noticeable artifacts as 3*a* or 3*e* do. Figures 3*c* and 3*d* and 3*g* and 3*h* show the difference images which have to be sent through the network. Note that the amount of reused pixels is higher in translations than in rotations and thus our two-level warping performs better than simply warping known points.

3.2 View Compression

So far, we have seen how to reduce the amount of information to be sent through the network taking advantage of the spatial coherence. The previous method in [7] sent the total difference image, although compressed, and achieved reductions of up to 55% with small movements of the camera. However, with this method, some pixels that do not contribute to the final image are still sent through the network and thus better compression ratios might be obtained. For example, there is no need to transmit those pixels that are not different; we can wipe them out and only send the different ones. These pixels can also be compressed. With this method we can obtain up to 8:1 reduction ratios for small movements and an average of 5-6:1 reduction over the JPEG compressed difference image. In order to do it we have to inform the client

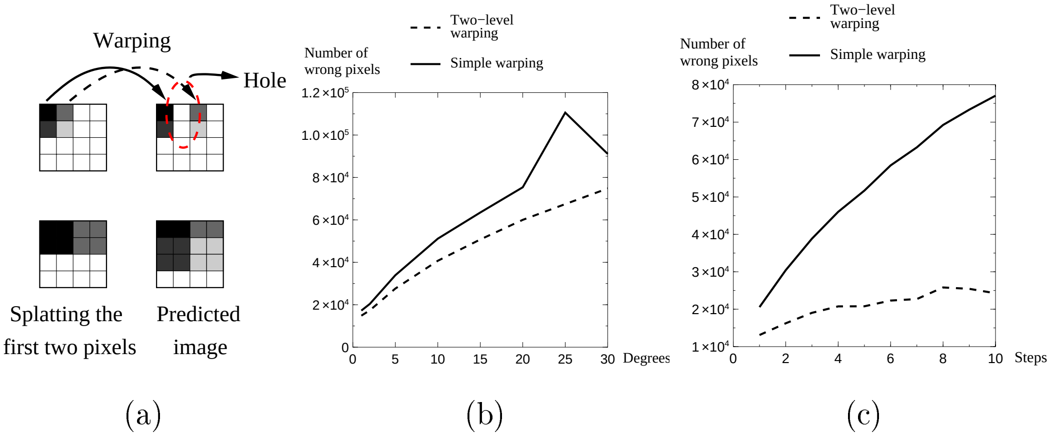


Fig. 2. In *(a)* we show our splatting strategy. Simple warping can produce holes (top row), so we splat known pixels (bottom row) to reduce the holes taking advantage of spatial coherence. The bottom row of this figure shows how the predicted image is finally built. Figures (b) and (c) show the comparison of the number of wrong pixels for simple warping and two-level warping in rotations and translations respectively. Note that for translations the improvement is higher as the projected colours are valid longer.

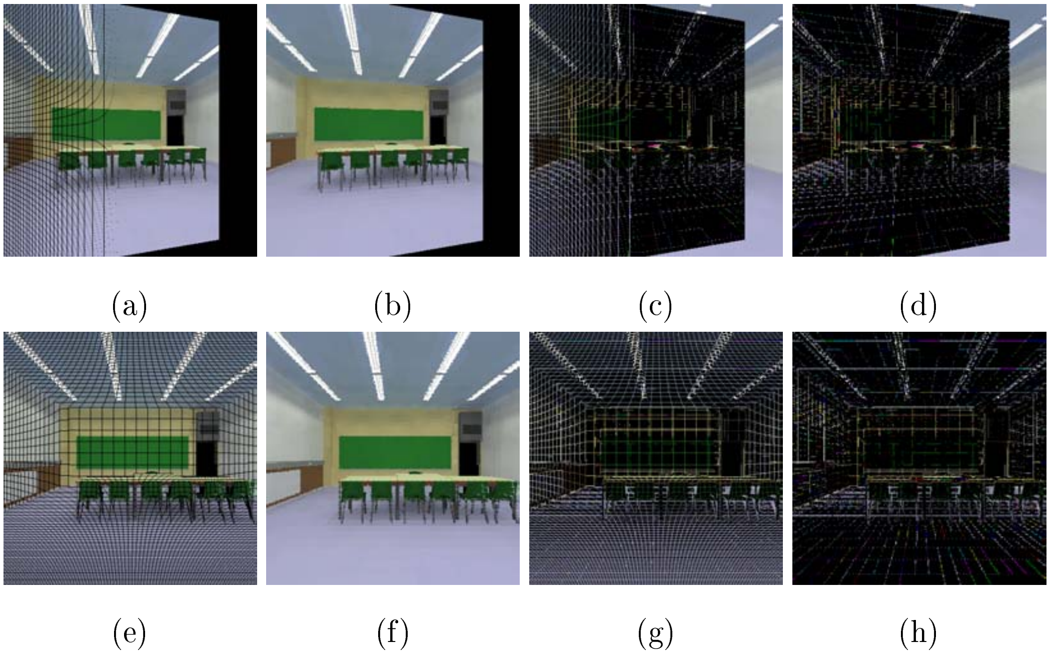


Fig. 3. Predicted images for a rotation of ten degrees (*(a)* to *(d)*) and a translation of 5 steps (*(e)* to *(h)*). Figures *(a)* and *(e)* were created using a simple forward warping while Figures *(b)* and *(f)* were created using a 2-level forward warping. The number of invalid pixels decreases a 20%. Figures *(c)* and *(d)* and *(g)* and *(h)* show the difference images respectively.

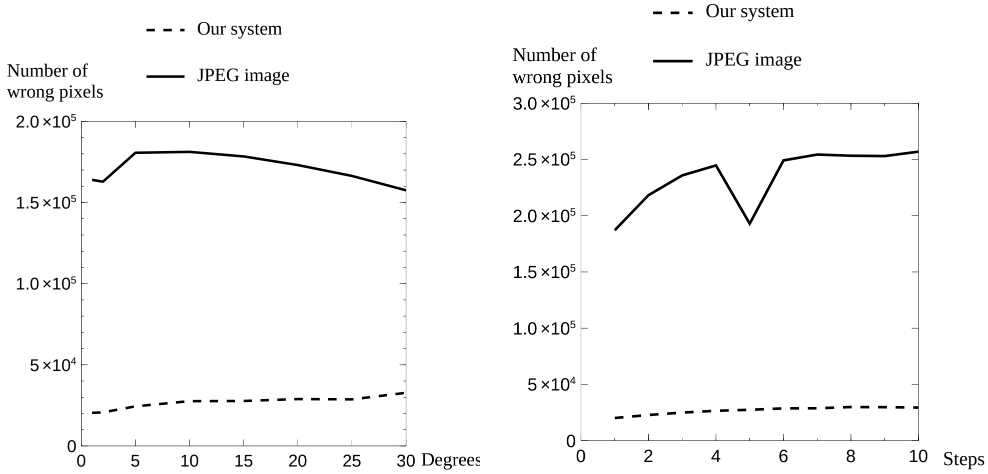
which pixels have to be updated. This can be achieved by creating a bitmap containing values of 1 for the pixels which have to be changed and zeroes elsewhere. The size of the bitmap will be 1/24th of the total image, as each colour channel is usually encoded with four bytes and a single bit is enough to determine if it has to be changed. Thus, if the difference pixels are less than 23/24 of the total (about 96%) pixels in the image, the amount of data to send (the bitmap plus the difference RGB values) will be smaller. In our tests we have found that the number of *valid pixels* is far higher than this 4 per cent for rotations of more than 30 degrees. Our method achieves good results with no loss of quality because the number of pixels that can be reused is in general higher than a 4% and thus, our masking method saves bandwidth. In Section 4 we will also explore some lossy schemes. Apart from that, we could further reduce the amount of information to be sent by also using a lossy compressing method [10] to reduce the resulting data.

3.3 Results

We have tested our method with different scenes and different movements of the camera, rotations and translations. The comparison between the size of information with our method and JPEG compressed difference images is depicted in Figure 4*a*. We can see that for the classroom scene, we achieve bandwidth reductions of up to 8:1 for small rotations (up to ten or fifteen degrees), and the average reduction stays high for wide movements around: 5:1 for rotations of 30 degrees. We have also tested translations. As we have already mentioned, translations show a better behavior due to the fact that the pixels can be reused longer. With our method we obtain improvements of up to 89.5% (ratio of 9.58:1) and an average of 85% of savings. In Figure 4*b* we show the reduction ratio for movements of 1 to 10 steps in the viewing direction.

4 Lossy Bandwidth Reduction Techniques

In this Section we present a set of methods for further bandwidth reduction which can lead to a small penalty in image quality. Of course the image quality (which is progressively updated) will depend on the strategy chosen. Thus, some of the strategies lead to undetectable artifacts. The first improvement consists in enhancing view prediction through the use of previous frame as the background of the image. This dramatically improves the quality of rotation movements, for instance, as the holes are reduced a high amount, and the possible introduced errors are less visible. Second, the predicted view is progressively improved by the transmission of the correct pixels by the server.



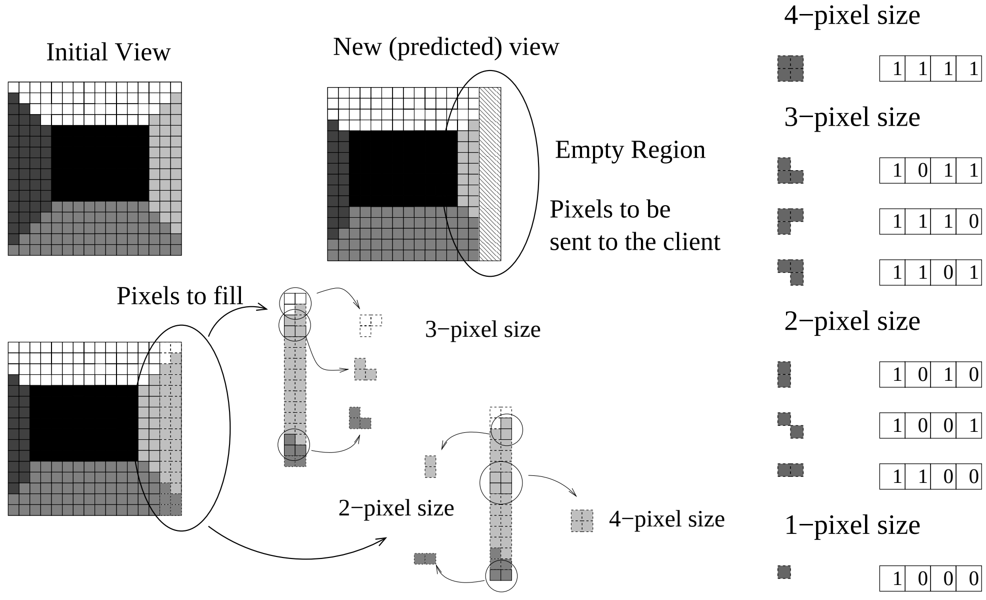
(a) (b)

Fig. 4. Comparison of results of our method against a JPEG compressed image of the difference image. With our method we achieve an improvement of up to a 80% on rotations *(a)*, that is, theoretically we can send 5 times more information with the same bandwidth. For translations *(b)* improvements are still better, up to a 85% of savings.

At its turn, the server can order the RGB values to be transmitted according to different strategies. The core of our strategy is to reuse the transmitted information as much as possible. That is, when the server is computing the difference between the predicted view and the correct one, it also checks if a pixel that has to be corrected in the final image has some neighbors that also have to be corrected and have the same colour than the currently analyzed. If this happens, instead of sending several times the same RGB value, we encode in a file the number of neighboring pixels with the same colour that have to be updated, and only send the RGB once. The regions we will use are of one to four pixels (see Figure 5*a*). We chose a size of four because it can be easily encoded in four bits (see Figure 5*b*). Then, the server sends to the client the codified array (gzip compressed) and progressively sends the information corresponding to the pixels that have to be updated. Now, the bitmap changes to a bigger file (now called splat file), but the savings in bandwidth are worth it. The information can then be sent following several strategies. The ones we have tested are: number of pixels updated by each RGB value, amount of information provided by each RGB value, and the amount of information weighted by the colour deviation of each RGB value.

4.1 Number of pixels

As we have mentioned, we can save some bandwidth if some of the RGB values sent are used to update more than one pixel position. In order to save



(a) (b)

Fig. 5. In Figure *(a)* we can see the analysis of the possible splats after a rotation. Some of the RGB values can be saved if we mark them with a certain *splat* shape. Figure *(b)* shows how to code the pixel splat shape on a byte. Only 4 bits are necessary, thus, two pixel splat shapes can be stored in a byte. Note that the combinations where the first bit is 0 are not valid as it is necessary that the actual RGB needs to be changed to study its possible splatting.

some more bandwidth, we can design a progressive scheme: the information is broken into pieces and sent in different moments, when network bandwidth is available. In order to better improve as soon as possible the resulting image, we can follow different strategies. Intuitively, a RGB value which updates a higher number of positions in the resulting image is better than one that only corrects one position. In this sense a RGB value that may correct more than one position will contribute to the final image proportionally to the number of pixels it will cover. Hence, a RGB value which can update up to four positions in the final image is more important and thus has to be updated first than a value which only updates one pixel in the final image. This can be used to prioritize the transmission. A first approach will be then to order the values to be transmitted according to the number of pixels that they will update in the final image. With this method, first the values which will update four positions in the final image are sent, then, the ones which update three, two and one respectively. This simple scheme of transmission is simple to encode and manage. The splat sizes can be extracted by the client from the splat file. This file must be sent only once as it is valid for all the pixels to update. Thus, the client knows the ordering the pixels. If the transmission is not possible, either due to problems on the network or because the client is moving the camera position too fast, the remaining data is discarded. This strategy is

(a) (b)

Fig. 6. Predicted images for a rotation of three degrees. The client predicted image 6*(a)* has been updated with the values which modify four positions in the final image 6*(b)*. The improvements have been obtained by only sending half the amount of information needed to improve the total image with the previous methods.

suitable thanks to its low cost if the client is a small or very loaded computer. Figure 6 shows the improvements obtained over the predicted view by only sending the RGB values corresponding to the pixels of splat size of four (they update four positions of the client image). With this method we achieve a bandwidth reduction of 2:1 factor if we only use these RGB values.

4.2 Information Theory

As we stated before, the amount of information contained in an RGB value is closely proportional to the number of pixels it covers in the final image. However, it is obvious that a pixel placed in the center of an image is more important that a pixel near the corners. It is desirable then, to update the central regions of the image earlier than the corners. To evaluate the difference in information carried by a certain RGB value, we use previous results of information theory tools applied to viewpoint selection. Vázquez *et al* [11] have presented a new measure, dubbed *viewpoint entropy* that can be understood as the amount of information seen from a point from a scene (see also [12]). It is based on the Shannon entropy [13], which gives the average information (or the uncertainty) of a random variable. Given a scene S if we suppose the information we deal is the projected area of each face, viewpoint entropy from a point p can be calculated by projecting all the faces of the scene onto the viewing plane and summing the contributions of each face and applying formula 1:

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

where N_f is the number of faces of the scene, A_i is the solid angle of 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 our case, the information we deal with is broken into pixels, as these are the units we will transmit. In our formula, the amount of information can be computed by substituting faces by pixels. Now we want to measure the amount of information provided by a part of an image, the region we want to correct. For a single pixel, this can be carried out by calculating its solid angle (Ω), and then applying $-\Omega * \log \Omega$. This will measure the amount of information covered by a certain pixel of the image. Thus, a pixel in the center of an image will provide higher amount of information than a pixel on a corner. If we can update more than one single position using only an RGB value, we have to sum up the total solid angle subtended by the pixels we update and apply the same formula. This method is better than the previous one because we are first sending the data which provides a higher amount of information.The computation of the information provided by every RGB value can be performed once the splat file has arrived. The client stores in an array the solid angles of the pixels of the image, and once the splat file has been received, the amount of information carried by each value is calculated by indexing the position of the pixels to be updated (this is encoded in the splat file) in the solid angles array, and then the information is computed as $-\Omega * \log \Omega$. This value will be the amount of information carried by an RGB value. In Figure 7*b* we can see an example of this method. The central parts of the image are updated earlier than with the previous method (7*c*).

4.3 Information Theory plus Colour Weighting

So far, we have only used the geometry of the scene. However, a wrong pixel with a high colour deviation from the correct one can attract the user attention and therefore introduce a more visible artifact than a small colour deviation for a larger region (see the bright colours incorrectly placed on the left side of image Figure 7*b*). In order to avoid this, we are going to take into account not only the information of the region updated, but also the difference between the final colour and the already existing colour in the predicted image. On the other hand, this improvement requires some extra changes to our method. The main reason is that the colour deviation is not known by the client side, and thus, the ordering of the pixels can not be known in advance, so this information has to be sent through the network. In spite of that, the quality

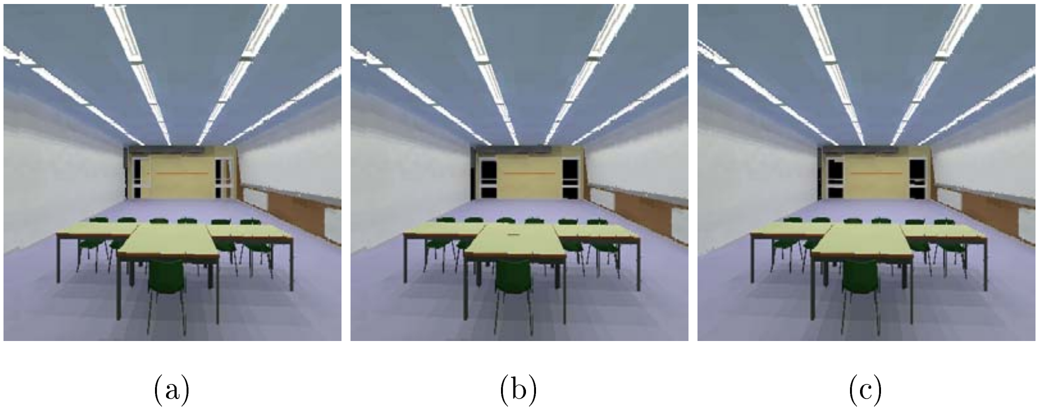


Fig. 7. Predicted images for a rotation of three degrees. Figure *(a)* shows the initial predicted image and Figure *(b)* shows the update of the client image by sending the same amount of information than for Figure 6 but the priority strategy is based on the information carried by the RGB values. Figure *(c)* shows the image updating using as prioritizing function the number of pixels. Note that the central parts of the image are earlier updated in *(b)* than in *(c)*.

of the final image is so good that it needs no more progressive update, and, consequently, we will save some more bandwidth, as the pixels which do change the client image by a small amount (let's say a 2%) can be discarded. The priority formula is then:

$$P = -SA * \log SA * \Delta L, \qquad (2)$$

where ΔL is the distance between the predicted colour and the correct one. We have chosen as distance $\Delta L = (|R_s - R_c| + |G_s - G_c| + |B_s - B_c|)/3$, where subscripts s and t attend for *estimated* and *correct* colour respectively. For a more exact computation RGB values may be translated to the corresponding CIE coordinates which encodes a colour in terms of perceptual issues. Now the server task is to predict a new view as the client would do, and subtract the correct one, but now, the different pixels with a low contribution to the final image are discarded. Then, the rest of the pixels are coded into the splat map, and the information sent consists of this map together with the RGB values of the modified colours. This leads to prioritize the correction of more noticeable artifacts. Moreover, the pixels which are updated first are the ones close to the center of the screen (their entropy is higher). We eliminate the RGB values whose update will only correct the colour in the final image by less than a 2%, which is almost imperceptible (even higher errors can be allowed without introducing visible artifacts), and we obtain a bandwidth reduction of 4:1 ratio. In Figure 8 there is a comparison between the previous method and the method which uses this colour and entropy-based prioritizing strategy. Figure 8a shows the predicted image and Figure 8b shows the predicted view using the same amount of updating RGB's than in previous systems. Figure 8c shows a grey scaled map of the difference between the computed view with

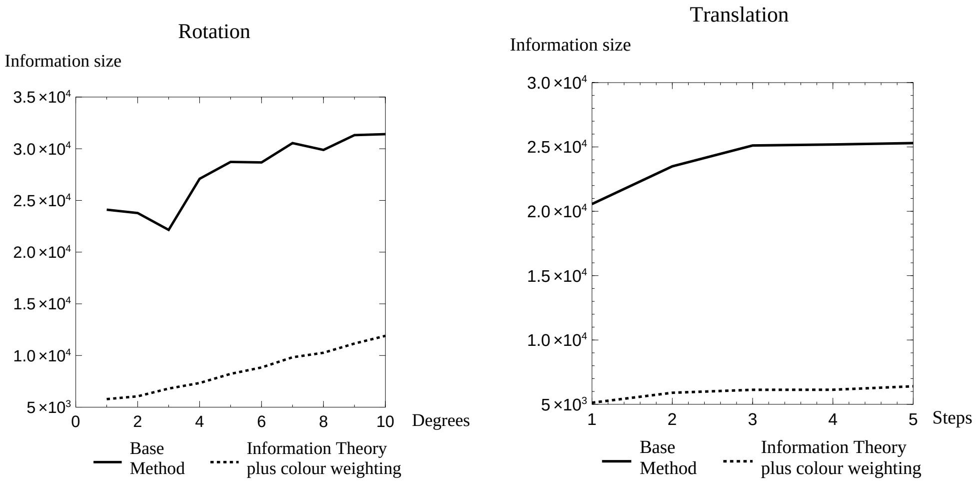
(a) (b) (c)

Fig. 8. Server image (left) and client image (center) for a rotation of three degrees. The client has been updated by sending one third of the amount of information needed for Figures 6 and 7 but the priority queue is ordered according to the information carried by each RGB value weighted by the colour deviation of the final image. The regions with visible artifacts are updated earlier. Thus, although the number of correct pixels is lower than the achieved with the first strategy, the results are notably better. Figure *(c)* shows a grey scaled image of the errors between the computed view and the correct one.

the correct one, notice that the incorrect pixels have a small (almost black) error. In this case, although roughly 12000 (of a 400 × 400 image) pixels are incorrect, the difference is hardly visible. Moreover, in this case, the total amount of information required is smaller, yielding a total reduction of up to 4:1 respect to the previous method.

4.4 Results

We have tested our new improvements for different movements of the camera. In Figure 10 we can see a comparison of the resulting images with the different strategies. Figure 10a shows the initial predicted image with the new method. Figure 10b shows the image improved by the colour values which cover four positions in the client side. In Figure 10c the image has been improved with the RGB's that provide a higher amount of information on the image. The image used to set as background correspond to a difference rotation of three degrees. If the intermediate frames were computed, that we would use is the previous one (with just one or half degree of difference), which would lead to diminish the presence of bad pixels. In Figures 10e to 10h we can see how the differences with the correct image decrease. Note that 10e shows the errors of the predicted view by the client. The strategy which uses the number of pixels (10b) reduces greatly the errors by only sending a small amount of data, while the method which uses Information Theory (10c) first reduces the errors in the center of the image. The third strategy, the one which also accounts for the difference in colour (10d) does not reduce large regions, but the final image



(a) (b)

Fig. 9. Reduction of bandwidth compared with the first strategy. Figure *(a)* shows the bandwidth size for rotations of up to ten degrees and Figure *(b)* shows the bandwidth reduction for translations of up to five steps.

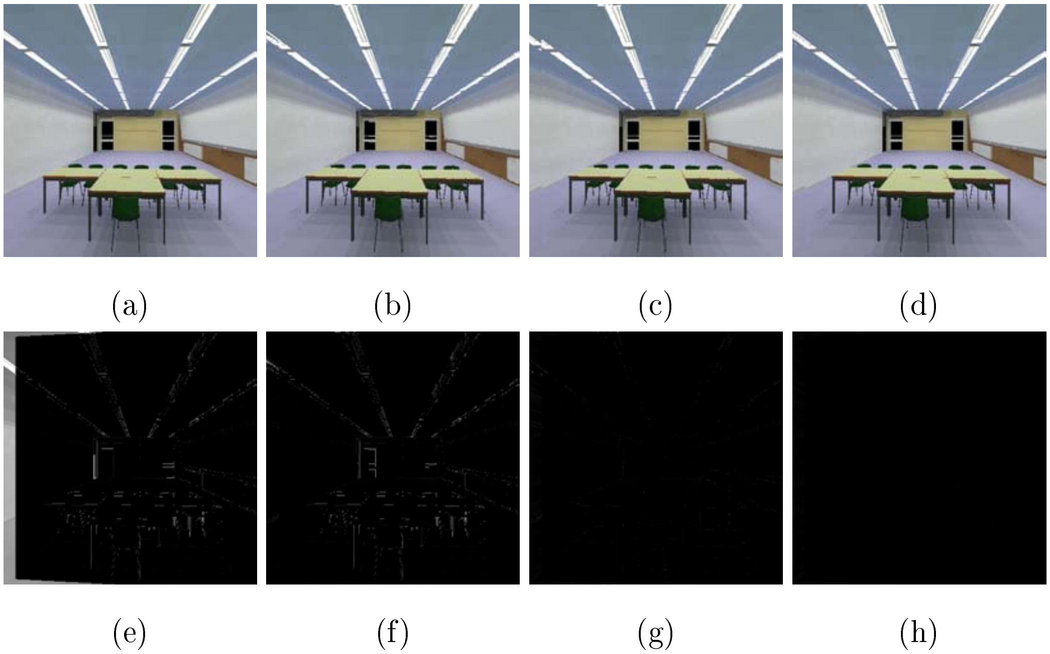
quality is superior to the other previous methods and virtually undetectable to the user (compare Figure 8*a* and 8*b*).

4.5 Depth Information

In both schemes, the one that uses two-level warping and the one which uses information theory and colour weighting, the information available at the beginning of the navigation is the initial image and the depth map. Subsequent frames are computed by reprojecting this information and updating wrong pixels. For the in-between frames, the depth information has not been transmitted. As the final quality is high, even without this information, depth values can then be sent in a lazy way. When the user stops moving or enough bandwidth is available, the depth values of current pixels are sent. In this case, it is only necessary to send the depth values of the pixels which have been updated, that is, the ones contained in the RGB coding map.

5 Conclusions and Future Work

We have presented a set of techniques for bandwidth reduction in remote navigation systems. First, we presented a lossless method which uses view prediction and bitmap coding. View prediction is carried out by a two-level forward warping in order to exploit spatial coherence. Big splats reduce the artifacts in the predicted view and therefore the information to be updated



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

(e) (f) (g) (h)

Fig. 10. Predicted images for a rotation of three degrees. Figure *(a)* shows the correct image and Figures *(b)* to *(d)* compare the different prioritizing strategies. Although in *(d)* half of the RGB values have not been corrected, the image is almost indistinguishable from the correct one. Figure *(e)* shows the errors of the initially predicted image with the correct one and Figures *(f)* to *(h)* show how the errors of the improved methods (*(b)* to *(d)*) decrease with respect to the initial predicted view.

also decreases. The information to be transmitted is reduced with the aid of a bitmap that encodes the positions of the pixels to be updated. The average ratio of data size reduction is about 5:1 for wide movements of even 30 degrees. Our second contribution is the combination of these techniques with two extra improvements focused on very low bandwidth environments. A better image prediction is achieved if we reuse the previous image as a background of the current frame. Moreover, this time the information is progressively sent from the server to the client prioritizing the data which improves the final image the most, taking into account the difference between the predicted colour and the correct one as well as the amount of information provided by the regions we correct. This scheme helps exploiting spatial coherence a further step and the reductions obtained are 2:1 over the previous method with almost no image quality degradation. In spite of the simplicity of these methods, dramatic bandwidth reduction ratios have been achieved.

In both cases the reduction ratios have been achieved only by the transmission of image pixels. Although depth values are only available for the initial frame, the results obtained are very good (for instance, the images in Figure 8 are almost indistinguishable). When the user is not moving or we have enough bandwidth, the server can send the depth information of the currently valid frame and thus update the total model. This will result in less information required for the next frames. In the future we want to explore dynamically varying environments. In this case the server should send information about the variation in the visible pixels (such as a vector field, combined with the bitmap that indicates the pixels to be moved) to obtain temporarily valid views in the client side. This information can also be sent lazily in order to obtain more savings.

Acknowledgments

This work has been partially supported TIC 2001-2416-C03-01 project of the Spanish Government, and SGR2001-00296 grant from Catalan Government.

References

[1] M. Levoy and P. Hanrahan. Light field rendering. In *Computer Graphics Proceedings (Proc. SIGGRAPH '96)*, pages 31–42, August 1996.

[2] S. J. Gortler, R. Grzeszczuk, R. Szeliski, and M. F. Cohen. The lumigraph. In *Computer Graphics Proceedings (Proc. SIGGRAPH '96)*, pages 43–54, 1996.

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

[4] S. E. Chen. Quicktime VR - an image-based approach to virtual environment navigation. In *Computer Graphics Proceedings (Proc. SIGGRAPH '95)*, pages 29–38, 1995.

[5] W. R. Mark, L. McMillan, and G. Bishop. Post-rendering 3D warping. In *Proc. of 1997 Symposium on Interactive 3D Graphics*, pages 7–16, New York, April 1997. ACM Press.

[6] M. Levoy. Polygon-assisted JPEG and MPEG compression of synthetic scenes. In *Computer Graphics Proceedings (Proc. SIGGRAPH '95)*, pages 21–28, August 1995.

[7] H. Biermann, A. Hertzmann, J. Meyer, and K. Perlin. Stateless remote environment navigation with view compression. Technical Report TR1999-784, Media Research Laboratory, New York University, New York, NY, 1999.

[8] D. Cohen-Or, Y. Mann, and S. Fleishman. Deep compression for streaming texture intensive animations. In Alyn Rockwood, editor, *Siggraph 1999, Computer Graphics Proceedings*, Annual Conference Series, pages 261–268, Los Angeles, 1999. ACM Siggraph, Addison Wesley Longman.

[9] Silicon Graphics Inc. Open GL web page, 2001. Specification document, available from http://www.opengl.org.

[10] Hee Cheol Yun, Brian K. Guenter, and Russell M. Mersereau. Lossless compression of computer-generated animation frames. *ACM Transactions on Graphics*, 16(4):359–396, October 1997. ISSN 0730-0301.

[11] 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.

[12] 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.

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