Compression of Light Field Images using Projective 2-D Warping method and Block matching

A project Report for EE 398A
Anand Kamat Tarcar
Electrical Engineering
Stanford University, CA
(anandkt@stanford.edu)

Abstract:
Light Fields are 2-D array of images of a 3D object taken from various viewpoints around the object. They constitute massive amounts of data if accurate representation of light field of an object is expected. Hence Compression is very much essential in case of light fields. In this paper we suggest 2-D projective warping method to predict a view from a previous view and use it in collaboration with motion compensated block matching technique to compress such datasets. The performance observed is better than what we would have if we had only used motion compensation.

Keywords: Light Fields, Motion Compensation, Projective Warping

I. INTRODUCTION

Light fields capture the essence of amount of light reflected from an object in all directions. The best way to obtain this is by taking images of the object of concern from all possible directions. Sometimes, the users might be interested in only certain views of the object in which case the images might be taken from several equally spaced points in a plane right in front of the object. This leads to a 2-D array of 2-D images of the object. So, overall the light fields represent a 4-D signal. Accurate representation of light fields requires images of the order of resolution of each image which can amount to tens of thousands of images. Hence compression of this Gigabytes of data becomes necessary. Example light fields are shown in Fig.1 (Crystal) and Fig.2 (Lego) which we use in our simulations. The classic paper [1] gives us an idea of what light fields are, how they can be represented and captured or obtained. This will not be discussed in this paper.

However, since these images are taken from predetermined locations with camera positions equidistant from one another and specific pre-known angles these consecutive views are related to one another in a deeper way. The consecutive views are very similar to one another and hence if we exploit this redundancy between views then large compression gains can be obtained.

Fig. 1 Crystal Light Field

In this paper we propose to predict each consecutive image from previous view using 2-D warping method explained shortly and code the
residual only along with side information required to decode the views.

II. RELATED WORK

A lot of work has been done in the past in the topic of compression of light fields. Shape adaptation was used in [2] where only the relevant portion related to shape of the object is encoded and sent while neglecting the background. Another major technique that has been extensively studied is Disparity compensation. In [3] a recursive algorithm for compressing light fields using disparity maps is proposed. Using four corner views of the array of images as reference the other images are predicted recursively using disparity maps and residuals encoded and sent. Compression ratio as high as 1000:1 was obtained at acceptable image quality. Among other modifications made to disparity compensation includes using it with lifting structure of Wavelet Transforms and shape adaptation as in [4], [5]. All of them use the same basic principle. Predict views from one another and encode the residual. We will use the same approach using a 2-D Projective warping approach.

III. MOTIVATION

It is very clear that each consecutive view is some sort of a 2-D warped version of the previous view. If we know the camera parameters for each view i.e. we have the geometry model then prediction of each view from the other becomes a simple affair. But in our case since we do not have any kind of camera parameters at our disposal we decided to come up with a technique in which we can project the previous image using 2-D warping which uses feature mapping between the two views to obtain a projection matrix. The projected view gives us a very good prediction of what the next view is supposed to look like. Hence, then by simple application of motion compensation we can encode the residual easily and send it to the decoder with motion vectors and projection matrix.

IV. ALGORITHM AND MODEL

IV.A. Model:
The 2-D warping algorithm predicts consecutive view from a previous view by using the projection matrix. This projection matrix gives the relation between the pair of views and is uniquely defined for each pair of views. It is a 3x3 matrix which when applied to the previous view gives us an estimate of how the next projected image will look like. Since each pixel in previous image is likely to be projected similarly in the next view this matrix is same for a pair of views and need be transmitted only once. But it is different for different pairs of views. This matrix needs to be quantized, encoded and transmitted along with the residual image described next as side information for successful reconstruction of views at the decoder.

![System Model](image)

Fig. 3 System Model

Since we want our encoder and decoder to be in Synchronization with each with regards to decoding we use reconstructed image at each step
to project the next view of the sequence. Once we have the reconstructed image from previous step and its projection we can use block matching algorithm as in motion compensation to get the actual prediction and hence the residual image. The block diagram of the system model is shown in Fig.3.

Now, why do we go for motion compensation? When we project the image using 2-D warping we notice that some artifacts are introduced at edges of the projected image. This is because projected view is predicted using previous view which lacks some information which is introduced in the next view as camera moves along the 2-D plane clicking different views. Some other artifacts and blurring may also be introduced in the projected image. Hence we use the previous reconstructed image in that case as another option to predict the next view. The views are then broken into equal sized blocks and motion compensation is applied using the previous reconstructed view and the projected predicted view of the previous reconstructed view. The best blocks from both views are found by block matching. The block from the current view is moved around a 2-D search area in the reconstructed and reconstructed projected previous view. By applying the Lagrangian cost function the best block among the two options mentioned is obtained and the difference gives us the residual block. Then Discrete Cosine Transform (DCT) is then applied to the residual image thus obtained which is then scalar quantized and entropy coded. This is transmitted to the decoder along with 2-D motion vector which gives information regarding movement in the horizontal and vertical direction in the previous view of the current block in the current view. The side information also includes the one bit needed to provide information related to which view was used for prediction of that particular block: whether it was the previous reconstructed view or the projection of the previous reconstructed view. The decoder also needs the projection matrix for each pair of views for reconstruction; this is also transmitted as side information. Since this is just a 3x3 matrix and does not have fixed statistics we choose to encode it using fixed length coding.

Once the decoder has the all the side information it needs it can simply de-quantize the residual, perform inverse DCT and obtain the actual reconstruction using motion vector and previous reconstructed view or its projection.

One important point to note is that the first view is always intra coded as it cannot be predicted and then decoded from other views. Also view sequencing plays an important role in the way the algorithm works which is discussed shortly.

IV.B Assumptions:
We have made a few assumptions while executing our project. Firstly, we assume that the motion vector components that we encode have a Laplacian distribution and a displacement of 10 in horizontal or vertical direction can be encoded using 16 bits. Assuming Laplacian distribution for motion vectors is a fair deal as it has been done in the [6] where variable block size motion estimation has been implemented for H.264.

Secondly, we encode the projection matrix with fixed length coding of 16 bits since it does not have specific statistics. Also it does not really add a great deal to the overall bits per pixel since it is just a 3x3 matrix for each pair of views.

IV.C View Sequencing:
View Sequencing is an important part of the overall scheme. We need the consecutive views to have minimum differences among them so we choose to read the 2-D array of images in a zig-zag pattern as shown in the Fig.4. That way the difference in camera movement between views is kept minimum and hence prediction is easier giving smaller residuals and better results.

Fig. 4 View Sequencing
IV.D. Obtaining Projected Image:
To obtain the Projection matrix relating two views we first need to get some corresponding matching points between the two views to capture the essence of how the pixels from previous view are projected in the next view in the sequence. This is done by feature matching algorithm called Harris corner detection, which gives us a number of corner points in each of the views in a pair. From these sets only the pairs of points which match each other in both the views with high correlation are chosen. Using these points as reference, we apply Random Sample Consensus algorithm to obtain the best fit projection matrix model for these points. It is an iterative method to estimate parameters of a mathematical model from a set of observed data which contains outliers. Using this matrix a normalized 2-D homogenous transform is applied to the previous view and nearest neighbor interpolation performed to obtain the projected view. Fig.5 shows transformation of reconstructed previous view into projection view and how we obtain residual image.

IV.E. Algorithm and Implementation:
Step 1: Read images in a sequential manner mentioned above.
Step 2: If it is the first image encode it using intra coding, apply DCT, Quantize it.
Step 3: If not the first image, use the 2-D warping technique to obtain projection of the previous reconstructed view as a prediction for the current view and the projection matrix. Include the rate of encoding the quantized matrix in the total rate.
Step 4: Consider the current view block-wise. For each block perform block matching in previous reconstructed view and its projection prediction.
**Step 5:** Select the best block based on the Lagrangian cost function.

\[ C_1 = D_1 + \lambda R_1 \]
\[ C_2 = D_2 + \lambda R_2 \]

Where \( C_1, D_1, R_1 \) are the total cost, distortion compared to block in current view and rate required to encode the residual block and motion vector for that block in the reconstructed previous view. Similarly, the second equation is representative of projection of previous reconstructed view. \( \lambda \) is related by the following relation in default case.

\[ \lambda = 0.2 Q^2 \]

where \( Q \) is the quantization step size.

**Step 6:** Select the best overall block from above based on minimum cost criteria and include additional rate for side information related to option we choose.

**Step 7:** Apply 2-D DCT to the residual and Quantize and transmit it to the decoder. Also perform inverse operations like de-quantizing and IDCT and motion compensation using motion vector side information to obtain reconstructed frame to use for next encoding stage.

**Step 8:** Calculate total distortion as mean squared error between the original view and the reconstructed view obtained. Total rate is the rate of encoding residual frame and side information as mentioned above. The Peak Signal to Noise Ratio (PSNR) is calculated from distortion using following relation:

\[ \text{PSNR} = 10 \log_{10} \frac{255^2}{d} \]

where \( d \) is the average distortion per pixel.

**Important Note:** A training module is implemented before actual encoding happens. In this i) All the images are intra coded and a probability density function of different DCT coefficients is obtained. ii) All images except the first one are coded using block matching with 2-D warping, only this time the Lagrangian cost does not include the rate for encoding the block but only the side information. This gives us the probability density of residual DCT coefficients. These pdf’s are used for encoding stage.

**V. RESULTS**

**V.A Parameters we used for our implementation:**
Since the algorithm is computationally and time exhaustive we decided to test our system for a small data set of 4x4 grid of 256x256 resolution images. The Light Field data sets we chose for our project are Crystal Light Field and Lego Light Field. These can be seen in Fig.1 and Fig.2 above. Only the luminance component of the light field data sets was compressed. The original images were of 1024x1024 resolution. These were downsampled and interpolated to avoid aliasing using bicubic interpolation to obtain the smaller resolution images. The default values of block size used was 8x8. For obtaining rate PSNR curves the Quantization step size was varied as \( 2^i \), where \( i = 3,4,5,6,7,8 \).

**V.B Rate-Distortion Behavior:**
Here we compare the performance of 2-D projective warping method with block matching and Motion Compensation with integer pixel accuracy method for the two data sets. It is clear from Fig.6 and Fig.7 that the projective warping method gives higher PSNR and lower rate. We can explain this by saying that the projective method is a better predictor for the next consecutive view and hence gives more efficient residuals resulting in lower distortion and smaller rate to encode the residual as compared to Motion Compensation in whose case we expect longer motion vectors to compensate for camera movement and more distorted residuals.

![Fig. 6 Rate PSNR curve for Crystal](image-url)
Also, we see that Crystal Light field has lower PSNR rate curve compared to Lego. This can be explained because Lego has kind of a flat background wall while Crystal has sharp background. Hence, as camera moves taking views the movement does not affect the flat background much causing smaller residuals and hence better performance compared to Crystal where camera movement affects the sharp background more.

V.C. Compression Ratio Performance:
For these results we use a default block size of 8. The compression ratios we observe for Crystal light field are as high at 60 at high quantization step size of $2^8$, while those for Lego light fields are as high as 100. It is clear from the Fig.8 and Fig.9 that motion compensation gives lower compression ratios than projective 2-D warping method. The reasoning for that is again that we get a better prediction from the projection than using the previous reconstructed view only and hence smaller residuals and motion vectors to code. This leads to a better data compression.

V.D. Rate-Distortion behavior for different block sizes:
We use three different block sizes: 8x8, 16x16, 32x32. From both Fig.10 and Fig.11 for the two data sets we observe that for 2-D warping method using smaller block size gives better results than using larger block sizes. Smaller block sizes mean better estimation of current view using previous reconstructed frame and its projection. This means lower distortion and hence better PSNR.

VI. SUMMARY AND CONCLUSION
Overall, we can summarize that light fields which amount to gigabytes of data can be compressed using redundancy between the views as much as possible. Motion compensation tries to do that but the projection that we obtain using the 2-D warping method turns out to be a better predictor.
of consecutive views and hence with use of block matching like that in motion compensation we can get much better results than the case where we only use motion compensation. In our method we exploit the idea that consecutive views are related to one another due to the specific way the camera moves and can be predicted from each other using projection matrix. Also, the way the views are sequenced also plays a critical role in getting good compression. Our aim is to minimize distortion as well as rate which we achieve with our method better than that with motion compensation.

VII. FUTURE WORK

In our project simulations we used small data sets by sub-sampling the light fields, yet we get very good compression ratios at acceptable image qualities. Accordingly, use of larger data could give much higher ratios. This could be verified. Also right now we are only considering algorithms upto integer pixel accuracy. Sub-pel algorithms with interpolation of blocks to get residuals could be developed and performance evaluated. I suspect it will give much better results than current version. Also, extra modes like copy mode could be introduced.

ACKNOWLEDGEMENTS

I would like to thank Professor Girod and Mina Makar for their help and advice throughout the project. Thanks to Professor Peter Kovesi and Professor Levoy’s group for access to open source Matlab library and Light field data sets. Special thanks to Chuo-Ling Chang for his DAPBT code (which we finally did not use in our simulation) and Huizhong Chen for his help.

REFERENCES


APPENDIX

Division of Work

<table>
<thead>
<tr>
<th>Task</th>
<th>Anand</th>
<th>Shinjini</th>
</tr>
</thead>
<tbody>
<tr>
<td>Choice of data sets</td>
<td>50%</td>
<td>50%</td>
</tr>
<tr>
<td>View Sequencing</td>
<td>100%</td>
<td>-</td>
</tr>
<tr>
<td>Projection Function*</td>
<td>20%</td>
<td>40%</td>
</tr>
<tr>
<td>Motion Compensation</td>
<td>100%</td>
<td>-</td>
</tr>
<tr>
<td>and Mode Selection</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Block matching</td>
<td>100%</td>
<td>-</td>
</tr>
<tr>
<td>Simulation and all</td>
<td>100%</td>
<td>-</td>
</tr>
<tr>
<td>results</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Presentation</td>
<td>40%</td>
<td>60%</td>
</tr>
<tr>
<td>Report</td>
<td>100%</td>
<td>-</td>
</tr>
</tbody>
</table>

*We had help from Prof. Peter Kovesi’s Matlab open source function Library