FAST LCD MOTION DEBLURRING BY DECIMATION AND OPTIMIZATION

Stanley H. Chan and Truong Q. Nguyen

ECE Dept, UCSD, La Jolla, CA 92093-0407
http://videoprocessing.ucsd.edu

ABSTRACT

The LCD deblurring problem is considered as a simple bounded quadratic programming problem and is solved using conjugate gradient with early stopping criteria to avoid excessive search. A decimation and interlace interpolation method is introduced to reduce the computing time. Solutions are competitive to those the generated by conventional Lucy Richardson algorithm, but using much shorter amount of time. The method can be extended to higher decimation factors. Visual subjective tests are conducted to justify our proposed method.

***Index Terms*—** LCD, motion blur, inverse problem, optimization, decimation.

1. INTRODUCTION

Liquid Crystal Display (LCD) differs from traditional cathode ray tube (CRT) display by its sample and hold behavior versus the CRT's impulse driven behavior. Such sample and hold behaviors cause motion blur in video. In [1], Har-Nor et al demonstrated a novel signal processing strategy to preprocess the video signal such that the video sequence is oversharpened. When it is subjected to the motion blur, the output would be close to the target video. The main algorithm that Har-Nor et al applied is standard Lucy Richardson (LR) algorithm [2] (using MATLAB routine `deconvlucy`). Though the results are satisfactory, the computing time is long. In this paper, we propose a new approach that reduces the computational time significantly while retaining visual performance.

We consider the problem as an optimization problem. First, as in most classical deblurring algorithms we are going to assume linear shift invariant image system. Thus for an $m \times n$ image x, and a point spread function (PSF) h, the output would be given by $y = h \star x$, where $\star$ denotes a convolution operator. In linear algebra words, if we stack columns of x as an $mn \times 1$ vector, and let A be the convolution matrix, then given the target $b \in \mathbb{R}^{mn\times 1}$ we would like to solve the following optimization problem:

$$\underset{x\in\mathbb{R}^{mn}}{\text{minimize}} \quad \|Ax - b\|_2^2 \tag{1}$$

$$\text{subject to} \quad 0 \le x \le 1. \tag{2}$$

The simple bound constraint $0 \le x \le 1$ is posed to limit the magnitude of the pixels to not exceeding 1 and 0.

This quadratic programming problem is difficult because it has a huge number of variables (medium sized image with 1000×1000 pixels will have 1 million variables). Therefore a good algorithm for this problem
1. should be fast;
2. does not need to store matrices A, and A^T;
3. can give visually satisfactory results (not only PSNR).
Based on these criteria, we propose to down sample the images and then apply conjugate gradient algorithm for the constrained optimization. The solution will be reconstructed using an interlace interpolation method. The image quality by the proposed method is as good as the one generated by LR, but using significantly shorter time.

The organization of this paper is as follows. In section 2 we will describe the main blocks of our algorithm. In section 3 we will extend the factor to arbitrary integers. Some experimental results are shown in section 4, and a concluding remark will be given in Section 5.

2. INTERLACE INTERPOLATION

2.1. Overall system

Human eyes pay much more attention to horizontal motions than vertical motions. Therefore, in this paper we assume that the PSF is horizontal (if not, we can approximate the PSF with a separable PSF and take its x-component). Then, since the rows are independently convolved, we can treat one row at a time. Also, since adjacent frames of a video are highly correlated, it is possible to consider only a few rows of the current frame, and reconstruct the remaining from its adjacent frames.

Fig. 1 shows the block diagram to compute one frame. For the k^{th} (k is even) frame I_k, the image is first down sampled and only the even rows are taken. Denote it by $(I_k)_{even}$. Then $(I_k)_{even}$ is passed into conjugate gradient algorithm to

This project is supported by AMD and UC Discovery.

$$\begin{tikzcd}
 & & & (\widehat{I}_k)_{even} \arrow[d] & (\widehat{I}_{k-1})_{odd} \arrow[dl] & \\
I_{k+1} \arrow[r] & \downarrow 2 \arrow[r, "(I_{k+1})_{odd}"] & CG \arrow[r, "(\widehat{I}_{k+1})_{odd}"] & \text{Reconstruction} \arrow[r] & (\widehat{I}_k)_{full} &
\end{tikzcd}$$

Fig. 1. Block diagram of our proposed fast inverse computation. We inverse compute the desired image for $(I_{k+1})_{odd}$, and use previous information $(I_{k-1})_{odd}$ and $(I_k)_{even}$ to reconstruct the current frame $(I_k)_{full}$. When proceeding to the next frame, we swap the odd and even indices.

obtain the inverse solution $(\widehat{I}_k)_{even}$. The subscript $even$ is used to emphasize that $(\widehat{I}_k)_{even}$ has only half number of rows as $\widehat{I}_k$ (to be determined). If k is odd, the roles of *even* and *odd* in the above will be swaped.

Suppose the current solution to be reconstructed is I_k. From previous iterations the full size solution $(\widehat{I}_{k-1})_{full}$ can be obtained. Since previously when $(\widehat{I}_{k-1})_{full}$ was reconstructed the k^{th} inverse solution $(\widehat{I}_k)_{even}$ had already been computed, in the k^{th} frame reconstruction there is not need to re-compute it again. However, the $k+1^{th}$ inverse solution to reconstruct $(\widehat{I}_k)_{full}$ is needed. Therefore, we apply conjugate gradient to the decimated image $(I_{k+1})_{odd}$ and obtain the inverse solution $(\widehat{I}_{k+1})_{odd}$. Then the three decimated inverse solutions $\{(\widehat{I}_{k-1})_{odd}, (\widehat{I}_k)_{even}, (\widehat{I}_{k+1})_{odd}\}$ are passed to the reconstruction scheme and the k^{th} reconstructed image can be obtained.

2.2. Interlace Interpolation

The interlace interpolation method is shown in Fig. 2. Having the inverse solution $(\widehat{I}_{k-1})_{odd}$, $(\widehat{I}_k)_{even}$, and $(\widehat{I}_{k+1})_{odd}$, we would like to fill in the unknown rows of $(\widehat{I}_k)_{even}$. First assume that the frames are motion compensated (using linear motion compensation) and aligned on the same pixel. Consider the corresponding pixels of the $k-1^{th}$ frame and the $k+1^{th}$ frame, the upper and lower pixel of the k^{th} frame. The solution can be estimated as

$$\begin{aligned}
&(\widehat{I}_k)_{full}(m,n) \\
&= 0.4(\widehat{I}_{k-1})_{odd}(m,n) + 0.4(\widehat{I}_{k+1})_{odd}(m,n) \\
&\quad + 0.1(\widehat{I}_k)_{even}(m,n+1) + 0.1(\widehat{I}_k)_{even}(m,n-1).
\end{aligned}$$

Note that this is a weighted sum of the four corresponding pixels, where the weights are empirically justified.

2.3. Conjugate Gradient

The main algorithm in this paper is conjugate gradient. It can be summarized as [3]:

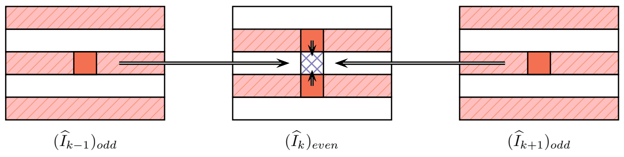


Fig. 2. Proposed linear interpolation method. To distinguish the known pixels from the unknown pixels we shaded the known ones. We take weighted mean of the corresponding pixels from previous frame $(\widehat{I}_{k-1})_{odd}(m,n)$, next frame $(\widehat{I}_{k+1})_{odd}(m,n)$, upper pixel $\widehat{I}_{k\,even}(m,n+1)$ and lower pixel $\widehat{I}_{k\,even}(m,n-1)$. Here we assume that the previous and the next frames are motion compensated.

```
Initial Condition
Given $x_0$, set
$r_0 \leftarrow Ax_0 - b$, $p_0 \leftarrow -r_0$, $k \leftarrow 0$

While $r_k \neq 0$
    $\alpha_k \leftarrow \frac{r_k^T r_k}{p_k^T A p_k}$
    $x_{k+1} \leftarrow x_k + \alpha_k p_k$
    $x = \begin{cases} 1 & x > 1 \\ 0 & x < 0 \end{cases}$
    $r_{k+1} \leftarrow r_k + \alpha_k A p_k$
    $\beta_{k+1} \leftarrow \frac{r_{k+1}^T r_{k+1}}{r_k^T r_k}$
    $p_{k+1} \leftarrow -r_{k+1} + \beta_{k+1} p_k$
    $k \leftarrow k+1$
End
```

There are a few implementation issues that we experienced:

The matrix vector products Ax can be obtained by Fast Fourier Transform based convolution: Given any PSF h and an image x, we first find their Fourier Transforms $\tilde{h} = \mathfrak{F}[h]$ and $\tilde{x} = \mathfrak{F}[x]$ so that the output can be found as $Ax = \mathfrak{F}^{-1}[\tilde{h} \cdot \tilde{x}]$, where $\cdot$ is element-wise multiplication. In MATLAB this can be done by calling

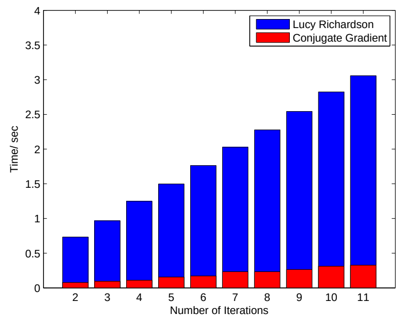
```
Ax = imfilter(x, h, 'replicate');
```

The second remark is the stopping criteria. In this problem, we found that the marginal benefit of each additional iteration after 10 iterations is small. Thus an early stop is sufficient for us to get a satisfactory solution. Typically, we stop the iteration when either

`max_itr >= 10`, *or*, `obj_val <= 1e-4`.

The third remark concerns about the initialization. Since there are strong correlations between adjacent frames, the solution of the previous frame can be served as an initial guess for the next frame. In experience this strategy can speed up the computation.

To compare performance of standard Lucy Richardson and our proposed conjugate gradient method we plot two graphs in Fig. 3. Fig. 3a shows the computing time of the two algorithms. As shown, the conjugate gradient is much faster than `deconvlucy`. Fig. 3b shows that conjugate gradient improves PSNR when number of iterations increases. At 10 iterations (which is our stopping criteria) CG algorithm is almost as good as LR algorithm.



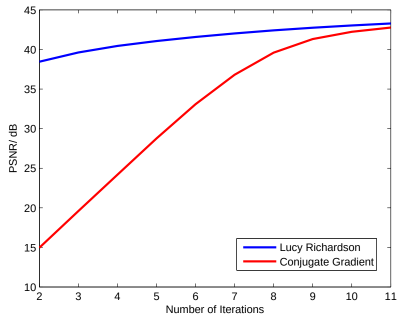


Fig. 3. Comparison between proposed conjugate gradient method and Lucy Richardson using `deconvlucy`. (a) Computation time: Our method is significantly faster than Lucy Richardson. (b) PSNR: Our method improves to a level as good as Lucy Richardson when number of iterations increases.

3. L TIMES DECIMATION

In this section the above proposed framework is extended to a higher decimation factor. In other words, we will demonstrate the method to down sample the image by L times, and then use L adjacent frames to reconstruct the solution.

For every L frames, the k-th image ($1 \leq k \leq L$) is sampled as $I(k : L : M, :)$, where M is number of rows of the original sized image, and the MATLAB index notation $(k : L : M, :)$ means to pick all rows from k to M with interval of every L rows.

Consider the k-th frame reconstruction. Suppose L frames $\{I_{k-L/2+1}, I_{k-L/2+2}, \ldots, I_k, \ldots, I_{k+L/2}\}$ are available from the conjugate gradient algorithm, and each of which is L times row decimated. We first register these frames so that the subpixels are aligned with the frame under reconstruction I_k. Since rows in one of these frames do not present in any of the other frames, the concatenation of all these L frames will form a full sized image. We do not use the information from the upper and lower pixel because when L is large, they will be too far (spatially vertical direction) from the pixel under consideration so they become irrelevant.

Fig. 4a shows the degradation of PSNR when the decimation factor is increased. As L increases, the separation between the current frame and the L-th frame also increases so the error from linear subpixel registration becomes larger. Since the pixels are not aligned well, PSNR decreases.

Fig. 4b shows the comparison of computing time versus the decimation factor. There are three key observations: First, the computing time of interlace interpolation is linearly increasing (a simple regression is performed to fit the observed data). Second, the conjugate gradient computing time decreases and reaches a steady state when decimation factor increases. This can be explained as when L becomes large, the time reduction from L times to $L + 1$ times becomes negligible, which implies that there is no need to decimate by a large factor. Third, the overall computing time shows that there is an optimum at a decimation factor of 4. This implies that further decimation beyond a factor of 4 is not worthwhile because the time spent on interlace interpolation would overwhelm the time saved in conjugate gradient.

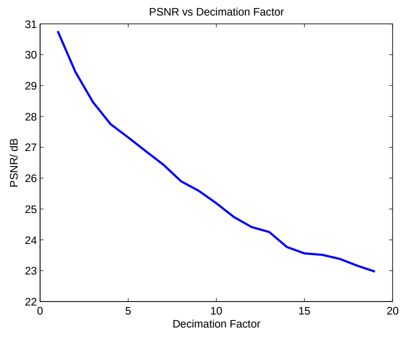
4. EXPERIMENTAL RESULTS

We now show a few computational results of our algorithm. Given an image, without performing any inverse computation the resultant image is blurred and is shown in Fig. 5a (PSNR = 35.0325dB). When applied standard Lucy Richardson and passed the solution through the PSF the resultant image is sharper, as shown in Fig. 5b (PSNR = 43.0294dB). Now, with our proposed method (using 10 iterations) we can obtain a resultant image as shown in Fig. 5c (PSNR = 42.2307dB). This image is sharper, especially around the edges of the buildings. Lastly, Fig. 5d shows the result using decimation $L = 2$ (PSNR = 41.4568dB). Despite the small degradation in PSNR, visually the difference between full scale CG and decimation by 2 is very small.

In addition to the PSNR comparison we also conducted visual subjective tests on 17 audience ($N = 17$). Using the same hardware setup as in [1] the audience are given 6 pairs of video sequences ((i) original and LR or (ii)original and proposed method). Audience is asked to give score based on a scale [-3, 3] where 0 represents “the same”, 3 represents “much better” and -3 represents “much worse” [4].

Statistical results [1] are shown in Table 1. To show that the means of L and D are very similar we first let null

[1]`http://videoprocessing.ucsd.edu/~stanleychan`



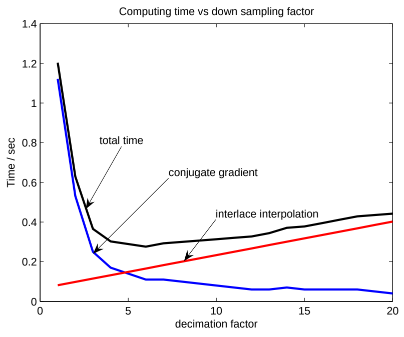


Fig. 4. Performance of L times decimation. (a) PSNR: PSNR drops as the decimation factor increases. (b) Computation time: The total computation time is the sum of conjugate gradient and interlace interpolation. Minimum exists when number of iterations is 4.

hypothesis $H_0 : \mu_L = \mu_D$ and alternative $H_1 : \mu_L \neq \mu_D$. Let 99% confidence interval ($\alpha = 0.01$). Using t-distribution tables we found that $t_{\alpha/2,N+N-2} = 2.763$. Thus H_0 is accepted if $-2.763 < t < 2.763$ where $t = \frac{\bar{x}_L - \bar{x}_D}{\Sigma_p}$ and $\Sigma_p = \frac{\sigma_L^2 + \sigma_D^2}{2}\sqrt{\frac{1}{N} + \frac{1}{N}}$. Using the values above we found that $t = 0.227, -0.154, -0.177$ respectively. Therefore H_0 is accepted and we can conclude that the proposed method is able to give similar results to LR but in a much shorter time.

5. CONCLUSION AND FUTURE RESEARCH

We formulated the LCD deblurring problem as an optimization problem. We proposed conjugate gradient with decimation and interlace interpretation method to speed up the computation. Results show that performance of our method is significantly faster than MATLAB's `deconvlucy`, with similar subjective performance.

As for future research, one can pay attention to the spatial coherence because the current method is unable to give spatially smooth images. Adding some regularization term to objective function may be a feasible way.

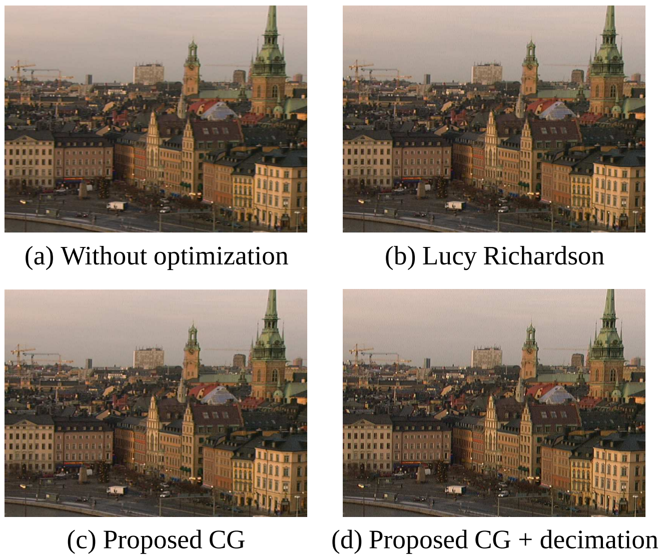


Fig. 5. Motion blur of various resultant images. (a) Using original unprocessed image. (b) Using Lucy Richardson `deconvlucy` with 10 iterations. (c) Using full scale conjugate gradient. (d) Using conjugate gradient with decimation factor 2.

Video Name	$\bar{x}_i$	σ_i
Stockholm L	1.206	1.016
Text L	1.235	1.091
Shield L	1.735	0.664
Stockholm D	1.294	1.047
Text D	1.176	0.951
Shield D	1.706	0.686

Table 1. Statistical results of visual subjective test on a scale of $[-3, 3]$. A positive value indicates that the audience prefer a processed video than the original. Notation L means Lucy Richardson, and D means our proposed decimation method.

6. REFERENCES

[1] S. Har-Noy and T.Q. Nguyen, "LCD motion blur reduction: A signal processing approach," *IEEE Trans. on Image Processing*, vol. 17, pp. 117–125, Feb 2008.

[2] L. B. Lucy, "An iterative technique for the rectification of observed distributions," *Astronomical Journal*, vol. 79, no. 6, Jun 1974.

[3] J. Nocedal and S. Wright, *Numerical Optimization*, Springer, second edition, 2000.

[4] ITU-R Recommendation, *Methodology for the subjective assessment of the quality of television pictures*, 2002, BT.500-11.