# Preconditioning for temporal video superresolution

Stéphane Pelletier and Jeremy R. Cooperstock
Department of Electrical and Computer Engineering
McGill University
Montréal, H3A 2A7, CANADA
stephane.pelletier@mail.mcgill.ca
jer@cim.mcgill.ca

**Abstract**

Temporal superresolution produces a video sequence of high temporal resolution (HTR), i.e. frame rate, from a set of video sequences of low temporal resolution (LTR). A method for preconditioning large systems of linear equations arising in temporal video superresolution problems is presented. We show that circulant block preconditioners can accelerate the convergence of iterative techniques such as the conjugate gradient method when solving such problems. Simulations demonstrate the effectiveness of our approach.

## 1 Introduction

Computational perception systems often rely on image sensors in order to obtain information about the surrounding environment. In such applications, high-resolution images are desired and often required. Image resolution depends on the physical characteristics of the imaging device, such as the number of photosensitive elements and the quality of the optics. Increasing resolution by using a better camera can be costly and sometimes infeasible. One possible solution employs image processing techniques such as superresolution [10][3] in order to overcome the limitations of the imaging devices. This solution, which may cost less than replacing the camera, also has the advantage of allowing the reutilisation of existing low-resolution cameras.

Most superresolution problems boil down to solving a large system of linear equations whose number of unknowns is the number of pixels in the desired high-resolution image or video sequence. Since the coefficient matrix associated with the system is generally sparse and structured, the problem can be solved using iterative techniques such as steepest descent or conjugate gradients [11]. Such iterative methods often benefit from performance improvements due to preconditioning, which transforms a system into another having the same solution, but that can be solved either more accurately or faster [7].

This paper presents a method for preconditioning systems of linear equations associated with video frame rate improvement problems, which can be assimilated to superresolution in the *time* domain. First, a brief description of superresolution and relevant preconditioning techniques is provided in Section 2. Next, our image formation model and the particular problem we are tackling are described in Section 3. Our preconditioning method is then explained in Section 4, followed by a summary of experimental results in Section 5.

## 2 Background

### 2.1 Spatial and temporal video superresolution

Superresolution techniques produce high-resolution images from a set of degraded and aliased low-resolution images by exploiting knowledge of the relative subpixel displacements of each low-resolution image with respect to a reference frame. These displacements provide different *views* of the same scene, which in turn provide complementary information that can be used to reconstruct images of better resolution. Subpixel displacements between the observed images can be the result of uncontrolled motion in a scene captured using a static camera, e.g., surveillance camera, or they can be due to controlled motion of the camera, e.g., terrestrial images captured from a satellite. When the displacements are unknown, they can be measured using image registration or motion flow techniques (see survey by Brown [4] for a comprehensive review).

Recently, Shechtman *et al* [12] introduced the concept of *spatio-temporal* superresolution, whose objective is to increase the resolution of video sequences in both the spatial and temporal domains. In addition to requiring spatial subpixel displacements between the observed images, space-time superresolution also relies on the presence of temporal *subframe* displacements between frames, as shown in Figure 1. The observed dynamic

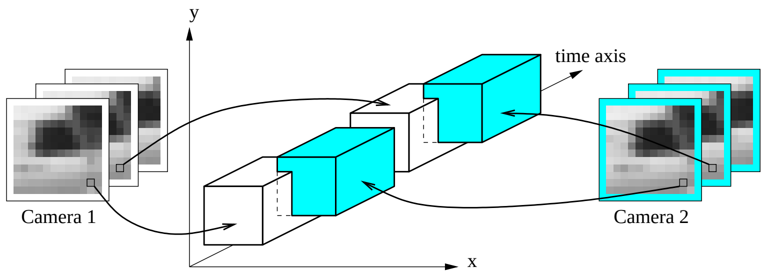

Figure 1: Displacements in space and time between video sequences are necessary for space-time superresolution.

scene is modeled as a space-time volume where each point has its own light intensity. An observed pixel is the result of an integration process over a subregion of this volume. In this figure, boxes having the same color represent the support of the space-time volumes associated to a *same* physical pixel in *different* frames. Using frames from temporally overlapping sources allows for the synthesis of a video sequence of higher temporal resolution, i.e, higher frame rate.

### 2.2 Preconditioning

Preconditioning can be very effective for accelerating the convergence of large linear systems of the form $A\mathbf{x} = b$. The idea is to find a non-singular matrix $M$ whose inverse can be computed quickly and such that $M^{-1}A\mathbf{x} = M^{-1}b$ is well-conditioned. Many image reconstruction problems such as image deblurring involve matrices whose structures can be well approximated by circulant matrices, which in turn are easily inverted using the fast Fourier transform (FFT) [6].

Even though matrices associated with spatial superresolution problems do not have the required structure, Nguyen *et al* [9] showed that by reordering the columns of the coefficient matrix, i.e. the pixels of the desired high-resolution image, it is possible to transform the original matrix into a block matrix whose blocks can be approximated efficiently by such preconditioners. In the following sections, we show that it is also possible to apply circulant preconditioners to temporal superresolution problems by properly reordering the *frames* of the desired high temporal resolution video sequence.

## 3 Problem formulation

### 3.1 Image formation model

The problem involves synthesizing a video sequence of high temporal resolution (HTR) from a set of $r$ video sequences of low temporal resolution (LTR), with a desired frame-rate improvement factor of $l$. We model each LTR video sequence as a noisy, temporally downsampled version of the HTR video sequence that has been blurred and shifted in time. Figure 2 illustrates our discrete approximation of the image formation model. Similar to the work of Shechtman *et al*, the space-time volume representing the dynamic

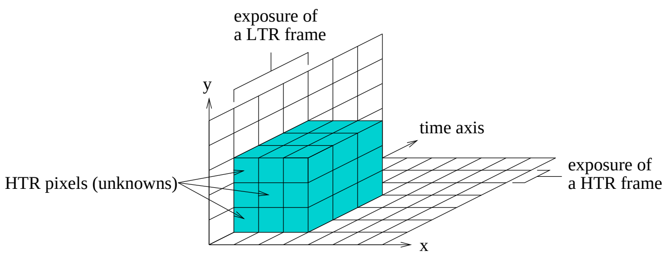

Figure 2: Image formation model in the discrete domain. Each LTR pixel (darker box) is a weighted average of several HTR pixels (smaller boxes).

scene is partitioned into smaller blocks, each one corresponding to a HTR pixel. Each slice of blocks in the $xy$ plane corresponds to a HTR frame, and its length along the time axis gives its exposure. The value of a LTR pixel is a weighted combination of several HTR pixels values in a region (darker box) whose coordinates are determined by the space-time location sensed by the LTR pixel. This information, which depends on camera parameters such as its point-spread function (PSF), exposure time and position, can be obtained using video sequence alignment methods [5]. Note that the volumes corresponding to different LTR pixels from the *same* video sequence can overlap in space, but not in time. Whereas spatial overlapping might be due to a camera that is out of focus or simply to blur induced by the imaging system, temporal overlapping is impossible, unless a single image sensor can capture multiple frames simultaneously.

A LTR video sequence is the result of a spatial and temporal blurring process applied to the HTR video sequence followed by a downsampling process along the time axis only. Spatial blur is caused by the spatial PSF of each camera, whereas temporal blur is due to the exposure period. In the following, we assume that

1. the PSF of each camera is invariant in space and in time,
2. the positions and orientations of the cameras do not change over time,
3. there is no 3D parallax effect (camera centers are close to one another with respect to the distance to the scene),
4. all LTR sequences have the same frame rate, but can have different exposures.

## 3.2 Mathematical model

Let $\mathbf{x}_1,\ \mathbf{x}_2,\ \ldots,\ \mathbf{x}_p$ be $p$ column vectors, each representing a frame of the HTR video sequence whose pixels are stored in lexicographic order [14]. The unknown HTR video sequence $\mathbf{x}$ can then be represented in vector form as

$$\mathbf{x} = \left[\mathbf{x}_1^T,\ \mathbf{x}_2^T,\ \ldots,\ \mathbf{x}_p^T\right]^T. \tag{1}$$

Similarly, if $\mathbf{y}_{(i,\ j)}$ denotes the $j$th frame of the $i$th observed LTR video sequence $\mathbf{y}_i$, the latter can be represented as

$$\mathbf{y}_i = \left[\mathbf{y}_{(i,\ 1)}^T,\ \mathbf{y}_{(i,\ 2)}^T,\ \ldots,\ \mathbf{y}_{(i,\ q)}^T\right]^T \tag{2}$$

where we assume for simplicity that the number of frames $q$ is the same for all LTR sequences. The LTR video sequences can be combined into a single vector

$$\mathbf{y} = \left[\mathbf{y}_1^T,\ \mathbf{y}_2^T,\ \ldots,\ \mathbf{y}_r^T\right]^T. \tag{3}$$

Thus, all observed pixel values are represented in $\mathbf{y}$. Using the previous equations, the relationship between the observed LTR frames and the unknown HTR frames can be expressed by

$$\mathbf{y} = H\mathbf{x} + \eta \tag{4}$$

where $H$ is a matrix representing the relationship between all LTR frames and the HTR frames and $\eta$ is a vector representing additive noise.

Let $B_i$ be a matrix representing the spatial PSF associated with the $i$th camera. The LTR frame $\mathbf{y}_{(i,\ j)}$ is then expressed as:

$$\mathbf{y}_{(i,\ j)} = \sum_{k=1}^{p} \omega_{(i,\ j,\ k)}\ B_i\ \mathbf{x}_k \tag{5}$$

where $\omega_{(i,\ j,\ k)}$ is a scalar indicating the contribution of the HTR frame $\mathbf{x}_k$ in $\mathbf{y}_{(i,\ j)}$. Since only a few HTR pixels actually contribute to any particular LTR pixel, the matrix $H$ is generally sparse and structured. For example, if $p = 9$, $r = 2$, $q = 3$ and $l = 3$, equation 4 might be expanded as

$$\begin{bmatrix} \mathbf{y}_{(1,\,1)} \\ \mathbf{y}_{(1,\,2)} \\ \mathbf{y}_{(1,\,3)} \\ \mathbf{y}_{(2,\,1)} \\ \mathbf{y}_{(2,\,2)} \\ \mathbf{y}_{(2,\,3)} \end{bmatrix} = \begin{bmatrix} B_1 & B_1 & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & B_1 & B_1 & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & B_1 & B_1 & \mathbf{0} \\ \mathbf{0} & B_2 & B_2 & B_2 & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & B_2 & B_2 & B_2 & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} \end{bmatrix} \begin{bmatrix} \mathbf{x}_1 \\ \mathbf{x}_2 \\ \mathbf{x}_3 \\ \mathbf{x}_4 \\ \mathbf{x}_5 \\ \mathbf{x}_6 \\ \mathbf{x}_7 \\ \mathbf{x}_8 \\ \mathbf{x}_9 \end{bmatrix} + \eta \qquad (6)$$

When there is an insufficient number of observations, the system of equations 4 will be ill-conditioned [1]. Therefore, we reformulate the problem as that of finding the minimum norm solution to the least squares problem

$$\min_{\mathbf{x}} \alpha(\mathbf{y} - H\mathbf{x})^T(\mathbf{y} - H\mathbf{x}) + \mathbf{x}^T\mathbf{x}. \qquad (7)$$

The solution to this under-constrained problem is

$$\mathbf{x} = H^T(HH^T + \lambda I)^{-1}\mathbf{y},\ \lambda = \frac{1}{\alpha} \qquad (8)$$

where $\lambda$ is the regularization parameter. Increasing $\lambda$ makes the system better conditioned, but also more different from the original one.

## 4 Preconditioning approach

We solve the temporal superresolution problem using the conjugate gradients (CG) algorithm, which is an iterative technique for solving systems whose coefficient matrix is Hermitian and positive-definite [8]. One advantage of iterative methods such as the CG algorithm is that one does not need to form the matrix $H$ explicitly. That is, it is only necessary to be able to compute the effect of applying $H$ to a vector. The convergence of CG depends on the distribution of the eigenvalues of the coefficient matrix [2]; the method converges rapidly when the system is well-conditioned or when its spectrum is clustered around one. Consequently, a good preconditioner should attempt to achieve these criteria in order to improve the performance of CG.

The structure of $H$ can be used to develop a preconditioner. To make it more obvious, the frames of $\mathbf{x}$ and the columns of $H$ are reordered. Let $D^{(k)}_{(i,\,j)}$ be a linear operator that downsamples an image sequence $\mathbf{x}$ of length $k$ by taking every $i$th frame, starting with the $j$th frame. For example,

$$D^{(9)}_{(3,\,2)} \left[\mathbf{x}_1^T,\, \mathbf{x}_2^T,\, \mathbf{x}_3^T,\, \mathbf{x}_4^T,\, \mathbf{x}_5^T,\, \mathbf{x}_6^T,\, \mathbf{x}_7^T,\, \mathbf{x}_8^T,\, \mathbf{x}_9^T\right]^T = \left[\mathbf{x}_2^T,\, \mathbf{x}_5^T,\, \mathbf{x}_8^T\right]^T. \qquad (9)$$

Then, the desired reordering for our problem is

$$\mathbf{x}_R = \begin{bmatrix} (D^{(p)}_{(l,\,1)}\mathbf{x})^T & (D^{(p)}_{(l,\,2)}\mathbf{x})^T & \cdots & (D^{(p)}_{(l,\,p/l)}\mathbf{x})^T \end{bmatrix}^T. \qquad (10)$$

In order not to modify the solution to the system of equations, the columns of matrix $H$ must also be reordered in the same way as the elements of $\mathbf{x}$. For example, when applying the previous reordering to equation 6, one gets

$$\begin{bmatrix} \mathbf{y}_{(1,\,1)} \\ \mathbf{y}_{(1,\,2)} \\ \mathbf{y}_{(1,\,3)} \\ \mathbf{y}_{(2,\,1)} \\ \mathbf{y}_{(2,\,2)} \\ \mathbf{y}_{(2,\,3)} \end{bmatrix} = \begin{bmatrix} B_1 & \mathbf{0} & \mathbf{0} & B_1 & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & B_1 & \mathbf{0} & \mathbf{0} & B_1 & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & B_1 & \mathbf{0} & \mathbf{0} & B_1 & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & B_2 & \mathbf{0} & B_2 & \mathbf{0} & \mathbf{0} & B_2 & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & B_2 & \mathbf{0} & B_2 & \mathbf{0} & \mathbf{0} & B_2 & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} \end{bmatrix} \begin{bmatrix} \mathbf{x}_1 \\ \mathbf{x}_4 \\ \mathbf{x}_7 \\ \mathbf{x}_2 \\ \mathbf{x}_5 \\ \mathbf{x}_8 \\ \mathbf{x}_3 \\ \mathbf{x}_6 \\ \mathbf{x}_9 \end{bmatrix} + \eta \qquad (11)$$

We note the similarity between our method and that of Nguyen *et al.* In their case, they reorder the pixels of the high-resolution image, whereas we reorder the frames of the LTR sequences. The reordered matrix $H_R$ has the following form:

$$H_R = \begin{bmatrix} H_{11} & H_{12} & \dots & H_{1l} \\ H_{21} & H_{22} & \dots & H_{2l} \\ \vdots & \vdots & \ddots & \vdots \\ H_{r1} & H_{r2} & \dots & H_{rl} \end{bmatrix} \qquad (12)$$

where $H_{ij}$ is a block Toeplitz matrix whose blocks are either $B_i$ or the zero matrix. For instance, in the case of equation 11, we have

$$H_{21} = \begin{bmatrix} \mathbf{0} & B_2 & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & B_2 \\ \mathbf{0} & \mathbf{0} & \mathbf{0} \end{bmatrix} \text{ and } H_{12} = \begin{bmatrix} B_1 & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & B_1 & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & B_1 \end{bmatrix}. \qquad (13)$$

Since the PSFs of all cameras are assumed to be spatially invariant, every block $B_i$ is *almost* Toeplitz; that is, all elements along a diagonal of $B_i$ have the same value, except for a few elements, whose values are equal to zero due to the fact that the discrete PSFs have finite dimensions. Moreover, by our assumption that the PSF functions do not change over time, corresponding diagonals from *different* blocks along a *block diagonal* within $H_{ij}$ are aligned. For example, the repetition of matrix $B_1$ in $H_{12}$ represents such a block diagonal. We approximate each matrix $H_{ij}$ by $T_{ij}$, where $T_{ij}$ is the Toeplitz matrix obtained by properly filling the holes, i.e. elements whose values are zero, along the diagonals of $H_{ij}$. This results in the following approximation for $H_R$:

$$H_R \approx \begin{bmatrix} T_{11} & T_{12} & \dots & T_{1l} \\ T_{21} & T_{22} & \dots & T_{2l} \\ \vdots & \vdots & \ddots & \vdots \\ T_{r1} & T_{r2} & \dots & T_{rl} \end{bmatrix}. \qquad (14)$$

Toeplitz matrices can be preconditioned by Strang's circulant preconditioner [13]. This is formed by copying the central diagonals of a Toeplitz matrix and wrapping them around to complete the circulant requirement, as illustrated in the following example:

$$\begin{bmatrix} a & b & c & 0 & 0 & 0 \\ 0 & a & b & c & 0 & 0 \\ 0 & 0 & a & b & c & 0 \\ 0 & 0 & 0 & a & b & c \\ 0 & 0 & 0 & 0 & a & b \\ 0 & 0 & 0 & 0 & 0 & a \end{bmatrix} \Rightarrow \begin{bmatrix} a & b & c & 0 & 0 & 0 \\ 0 & a & b & c & 0 & 0 \\ 0 & 0 & a & b & c & 0 \\ 0 & 0 & 0 & a & b & c \\ \mathbf{c} & 0 & 0 & 0 & a & b \\ \mathbf{b} & \mathbf{c} & 0 & 0 & 0 & a \end{bmatrix}. \tag{15}$$

The advantage of a circulant preconditioner $C$ is that it is unitary similar to a diagonal matrix through the Fourier transform, namely

$$C = F^H D F \tag{16}$$

where $F$ is the Fourier transform, $D$ is a diagonal matrix containing the eigenvalues of $C$ and $H$ is the conjugate transpose operator. The eigenvalues of $C$ are computed easily by taking the Fourier transform of its first column, and thus, circulant matrices can be inverted efficiently using two FFT operations. Based on equations 14 and 16, we define our preconditioner $M$ as

$$M = \begin{bmatrix} F^H & 0 & \dots & 0 \\ 0 & F^H & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & F^H \end{bmatrix} \begin{bmatrix} D_{11} & D_{12} & \dots & D_{1l} \\ D_{21} & D_{22} & \dots & D_{2l} \\ \vdots & \vdots & \ddots & \vdots \\ D_{r1} & D_{r2} & \dots & D_{rl} \end{bmatrix} \begin{bmatrix} F & 0 & \dots & 0 \\ 0 & F & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & F \end{bmatrix}. \tag{17}$$

where $D_{ij}$ is the diagonal matrix corresponding to $T_{ij}$ through its circulant approximation. Since $M$ can be expressed as a block matrix whose blocks are diagonal matrices, it can be inverted easily using FFTs.

## 5 Experiments

To verify the validity of our preconditioning approach, we begin with an idealized HTR video sequence of forty 64x64 pixel frames, hereafter called the *reference* sequence. This is blurred with a kernel whose support along the time axis corresponds to four HTR frames. Next, we downsample the blurred result using different time offsets to produce four LTR video sequences, simulating the capture of a dynamic scene using four unsynchronized cameras, as illustrated in Figure 3. For comparison purposes, a frame from the

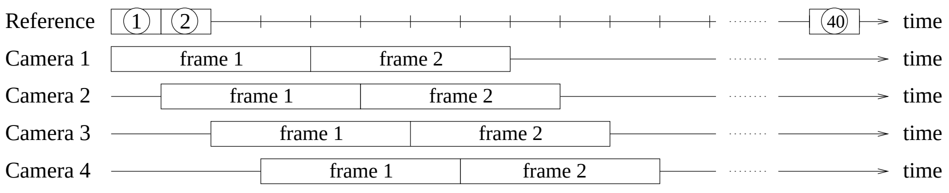

Figure 3: The exposure periods of the four LTR cameras overlap in time.

reference sequence and its corresponding frame in one of the LTR sequences are shown

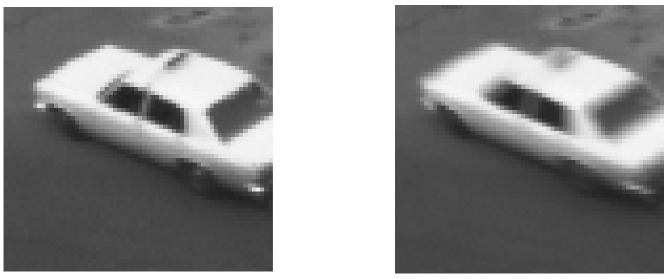

Figure 4: A reference frame and an observed LTR frame.

in Figure 4. The longer exposure period of the (simulated) LTR cameras produces frames that are more sensitive to motion blur. We then synthesized a video sequence of higher frame rate using the method of conjugate gradients applied to the four LTR video sequences. The regularization parameter $\lambda$ was chosen manually and set to 0.001. The reconstructed video sequence was then compared against the reference sequence. Figure 5 shows the results obtained after five iterations both with and without preconditioning. For space reasons, only three frames are shown for each sequence. As can be seen, the preconditioned algorithm produces frames of higher quality for the same number of iterations. Figure 6 illustrates the convergence, as indicated by the sum of squared error (SSE) between the reconstructed and original sequences for these two cases. In this example, five iterations were sufficient for the preconditioned algorithm to converge to the solution, whereas the unpreconditioned algorithm required approximately twenty iterations. Thus, circulant preconditioners are clearly useful for improving performance of iterative methods when solving temporal video superresolution problems.

## 6 Conclusion

In this paper, we presented a method for preconditioning large systems of linear equations associated with temporal video superresolution problems. Nguyen *et al* previously showed that circulant preconditioners can be adapted to the spatial superresolution problem by reordering the pixels of the desired high-resolution image. We extend their result to demonstrate that such preconditioners can also be used in temporal superresolution problems by properly reordering the frames of the desired high temporal resolution video sequence.

## 7 Acknowledgments

This work was generously supported by a Phase IIIb research grant from the Institute for Robotics and Intelligent Systems (IRIS) and a Precarn scholarship. The authors would like to thanks the KOGS/IAKS Universität Karlsruhe for providing the video sequence used in this paper.

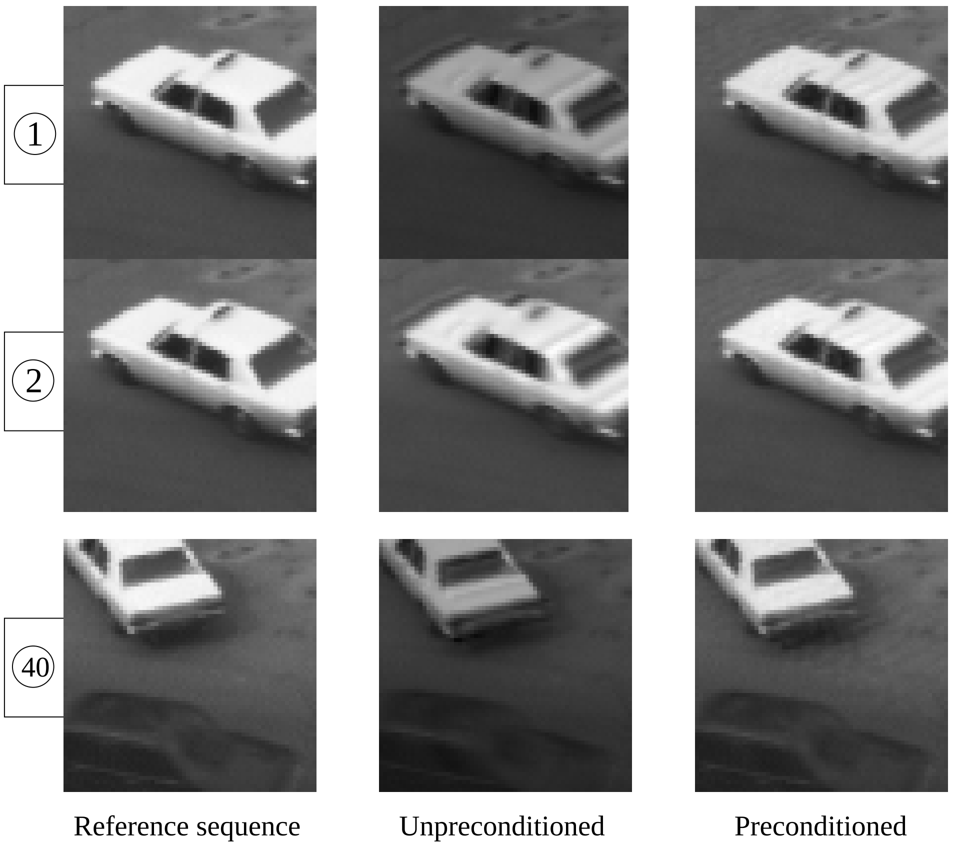

Figure 5: Qualitative comparison between the results obtained with and without preconditioning after 5 iterations, using $\lambda = 0.001$.

## References

[1] Harry C. Andrews and B. R. Hunt. *Digital Image Restoration*. Prentice Hall Professional Technical Reference, 1977.

[2] O. Axelsson and V. A. Barker. *Finite Element Solution of Boundary Value Problems, Theory and Computation*. Academic Press, Inc, New York, 1984.

[3] Sean Borman and Robert L. Stevenson. Spatial resolution enhancement of low-resolution image sequences: A comprehensive review with directions for future research. Technical report, Department of Electrical Engineering, University of Notre Dame, Notre Dame, Indiana, USA, July 1998.

[4] Lisa Gottesfeld Brown. A survey of image registration techniques. *ACM Computing Surveys*, 24(4):325–376, 1992.

[5] Yaron Caspi and Michal Irani. Spatio-temporal alignment of sequences. *IEEE Trans. Pattern Anal. Mach. Intell.*, 24(11):1409–1424, 2002.

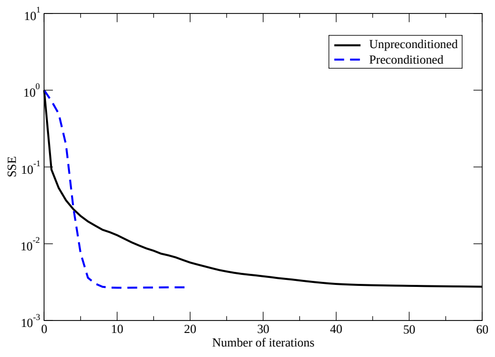

Figure 6: Sum of Squared Error between the solution and the reference HTR sequence as a function of the number of iterations.

[6] R. Chan and M. Ng. Conjugate gradient methods for toeplitz systems, 1996.

[7] Ke Chen. *Matrix Preconditioning Techniques and Applications*, volume 19 of *Cambridge Monographs on Applied and Computational Mathematics*. Cambridge University Press, Cambridge, UK, 2005.

[8] G. H. Golub and C. F. Van Loan. *Matrix Computations*. The Johns Hopkins University Press, Baltimore, MD, USA, second edition, 1989.

[9] Nhat Nguyen, Peyman Milanfar, and Gene H. Golub. A computationally efficient superresolution image reconstruction algorithm. *IEEE Transactions on Image Processing*, 10(4):573–583, 2001.

[10] S. C. Park, M. K. Park, and M. G. Kang. Super-resolution image reconstruction: A technical overview. *IEEE Signal Processing Magazine*, 20(3):21–36, May 2003.

[11] Y. Saad. *Iterative Methods for Sparse Linear Systems*. Society for Industrial and Applied Mathematics, Philadelphia, PA, USA, 2003.

[12] Eli Shechtman, Yaron Caspi, and Michal Irani. Space-time super-resolution. *IEEE Trans. Pattern Anal. Mach. Intell.*, 27(4):531–545, 2005.

[13] Gilbert Strang. A proposal for toeplitz matrix calculations. *Stud. Appl. Math.*, 74(2):171–176, 1986.

[14] E. W. Weisstein. Lexicographic order., From MathWorld–A Wolfram Web Resource. *http://mathworld.wolfram.com/LexicographicOrder.html*.