Space-Time Super-Resolution

Thesis for the M.Sc. Degree

by

Eli Shechtman

Under the Supervision of
Michal Irani
Faculty of Mathematics and Computer Science
The Weizmann Institute of Science

Submitted to the Feinberg Graduate School of
the Weizmann Institute of Science
Rehovot 76100, Israel

February 27, 2003

Acknowledgments

I would like to thank my advisor Prof. Michal Irani for her continuous support, encouragement, scientific guidance and friendship that have guided me throughout the work on my master's. Thank you Michal for giving me the opportunity to absorb some of your non-compromising quest for excellence, originality and quality.

Special thanks to Dr. Yaron Caspi, who was much more than the co-auther of my papers. Thank you for enriching me with your academic knowledge, professionalism and work tempo, for many challenging discussions and for the friendship.

I would like to thank Dr. Merav Galun and Prof. Achi Brandt for some very helpful discussions and suggestions. Thank you Achi for teaching me one of the more fascinating courses I took during my studies. Thanks also to Lihi Zelnik and Prof. Ronen Basri for reviewing the first drafts of the ECCV paper and noting valuable remarks.

I wish to thank the following people for contributing to my final decision to leave my master studies in Tel-Aviv Univ. and to move to the Weizmann Institute (listed in chronological order): My mother, Prof. Shimon Ullman, Prof. Harry Dim, Prof. Ronen Basri and Prof. Michal Irani. I had wonderful two years and I don't regret a moment for that decision. Many thanks also to my friends in the vision and robotics lab for their friendship and help during the work - Tal, Lihi, Byung-Woo, Aya, Denis, Michel and many others. Thanks also to the administrative and technical staff of the department for their essential help.

I am most grateful to my loving and supporting parents Mila and Radi for always emphasizing the importance of education, excellence and diligence, to my sister Sivan and to my dearest and beloved wife Irit who supported and strengthened me at day and (mostly) at night.

Thank you,

Eli

Contents

Abstract

We propose a method for constructing a video sequence of high space-time resolution by combining information from multiple low-resolution video sequences of the same dynamic scene. Super-resolution is performed simultaneously in time and in space. By "temporal super-resolution" we mean recovering rapid dynamic events that occur faster than regular frame-rate. Such dynamic events are not visible (or else observed incorrectly) in any of the input sequences, even if these are played in "slow-motion".

The spatial and temporal dimensions are very different in nature, yet are interrelated. This leads to interesting visual tradeoffs in time and space, and to new video applications. These include: (i) treatment of *spatial* artifacts (e.g., motion-blur) by increasing the *temporal* resolution, and (ii) combination of input sequences of different space-time resolutions (e.g., NTSC, PAL, and even high quality still images) to generate a high quality video sequence.

We further analyze and compare characteristics of temporal super-resolution to those of spatial super-resolution. These include: How many video cameras are needed to obtain increased resolution? What is the upper bound on resolution improvement via super-resolution? What is the optimal camera configuration for various scenarios? What is the temporal analogue to the spatial "ringing" effect?

1 Introduction

A video camera has limited spatial and temporal resolution. The spatial resolution is determined by the spatial density of the detectors in the camera and by their induced blur. These factors limit the minimal size of spatial features or objects that can be visually detected in an image. The temporal resolution is determined by the frame-rate and by the exposure-time of the camera. These limit the maximal speed of dynamic events that can be observed in a video sequence.

Methods have been proposed for increasing the spatial resolution of images by combining information from multiple low-resolution images obtained at sub-pixel displacements (e.g. [1, 2, 3, 6, 7, 11, 13, 14, 15, 16]. See [4] for a comprehensive review). An extension of [15] for increasing the spatial resolution in 3-dimensional (x,y,z) medical imagery has been proposed in [12], where MRI data was reconstructed both within image slices (x and y axis) and between the slices (z axis).

The above mentioned methods, however, usually assume static scenes with limited *spatial* resolution, and do not address the limited *temporal* resolution observed in dynamic scenes. In this thesis we extend the notion of super-resolution to the *space-time* domain. We propose a unified framework for increasing the resolution both in time and in space by combining information from multiple *video sequences* of dynamic scenes obtained at (sub-pixel) spatial and (sub-frame) temporal misalignments. As will be shown, this enables new visual capabilities of dynamic events, gives rise to visual tradeoffs between time and space, and leads to new video applications. These are substantial in the presence of very fast dynamic events. From here on we will use SR as an abbreviation for the frequently used term "super-resolution".

Rapid dynamic events that occur faster than the frame-rate of video cameras are not visible (or else captured incorrectly) in the recorded video sequences. This problem is often evident in sports videos (e.g., tennis, baseball, hockey), where it is impossible to see the full motion or the behavior of the fast moving ball/puck. There are two typical visual effects in video sequences which are caused by very fast motion. One effect (motion blur) is caused by the exposure-time of the camera, and the other effect (motion aliasing) is due to the temporal sub-sampling introduced by the frame-rate of the camera:

(i) *Motion Blur:* The camera integrates the light coming from the scene during the exposure time in order to generate each frame. As a result, fast moving objects produce a noted blur along their trajectory, often resulting in distorted or unrecognizable object shapes. The faster the object moves, the stronger this effect is, especially if the trajectory of the moving object is not linear. This effect is notable in the distorted shapes of the tennis ball shown in Fig. 1. Note also that the tennis racket also "disappears" in Fig. 1.b. Methods for treating motion blur in the context of image-based SR were proposed in [2, 1]. These methods however, require prior segmentation of moving objects and the estimation of their motions. Such motion analysis may be impossible in the presence of severe shape distortions of the type shown in Fig. 1. We will show that by increasing the *temporal resolution* using information from multiple video sequences, *spatial artifacts* such as motion blur can be handled without the need to separate static and dynamic scene components or estimate their motions. However, unlike spatial SR, in temporal SR a *minimum* number

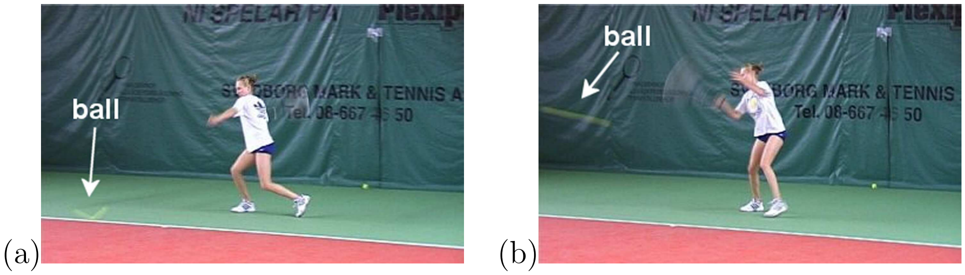


Figure 1: **Motion blur.** *Distorted shape due to motion blur of very fast moving objects (the tennis ball and the racket) in a real tennis video. The perceived distortion of the ball is marked by a white arrow. Note, the "V"-like shape of the ball in (a), and the elongated shape of the ball in (b). The racket has almost "disappeared".*

of input cameras (video sequences) is needed for motion-deblurring. Using less cameras might cause the opposite effect of increased motion-blur. A practical lower bound on the number of cameras will be derived.

(ii) *Motion-Based (Temporal) Aliasing:* A more severe problem in video sequences of fast dynamic events is false visual illusions caused by aliasing in time. Motion aliasing occurs when the trajectory generated by a fast moving object is characterized by frequencies which are higher than the frame-rate of the camera (i.e., the temporal sampling rate). When that happens, the high temporal frequencies are "folded" into the low temporal frequencies. The observable result is a distorted or even false trajectory of the moving object. This effect is illustrated in Fig. 2, where a ball moves fast in sinusoidal trajectory of high frequency (Fig. 2.a). Because the frame-rate is much lower (below Nyquist frequency of the trajectory), the *observed* trajectory of the ball over time is a straight line (Fig. 2.b). Playing that video sequence in "slow-motion" will not correct this false visual effect (Fig. 2.c). Another example of motion-based aliasing is the well-known visual illusion called the "wagon wheel effect": When a wheel is spinning very fast, beyond a certain speed it will appear to be rotating in the "wrong" direction.

Neither the motion-based aliasing nor the motion blur can be treated by playing such video sequences in "slow-motion", even when sophisticated temporal interpolations are used to increase the frame-rate (as in video format conversion or "re-timing" methods [10, 20]). This is because the information contained in a single video sequence is insufficient to recover the missing information of very fast dynamic events. The high temporal resolution has been lost due to excessive blur and excessive subsampling in time. Multiple video sequences, on the other hand, provide additional samples of the dynamic space-time scene. While none of the individual sequences provides enough visual information, combining the information from all the sequences allows to generate a video sequence of high space-time resolution, which displays the correct dynamic events. Thus, for example, a reconstructed high-resolution sequence will display the correct motion of the wagon wheel despite it appearing incorrectly in *all* of the input sequences.

The spatial and temporal dimensions are very different in nature, yet are inter-related.

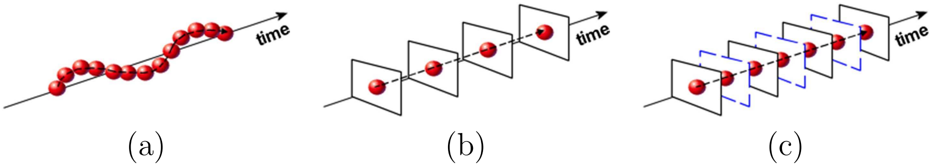


Figure 2: **Motion aliasing.** *(a) shows a ball moving in a sinusoidal trajectory over time. (b) displays an image sequence of the ball captured at low frame-rate. The perceived motion is along a straight line. This false perception is referred to in the thesis as "motion aliasing". (c) Illustrates that even using an ideal temporal interpolation for "slow-motion" will not produces the correct motion. The filled-in frames are indicated by dashed blue line. In other words, the false perception cannot be corrected by playing the video sequence in slow-motion, as the information is already lost in the video recording (b).*

This introduces visual tradeoffs between space and time, which are unique to spatio-temporal SR, and are not applicable in traditional spatial (i.e., image-based) SR. For example, output sequences of different space-time resolutions can be generated from the same input sequences. A large increase in the temporal resolution usually comes at the expense of a large increase in the spatial resolution, and vice versa.

Furthermore, input sequences of different space-time resolutions can be meaningfully combined in our framework. In traditional image-based SR there is no benefit in combining input images of different spatial resolutions, since a high-resolution image will subsume the information contained in a low-resolution image. This, however, is not the case here. Different types of cameras of different space-time resolutions may provide *complementary* information. Thus, for example, we can combine information obtained by high-quality still cameras (which have very high spatial-resolution, but extremely low "temporal resolution"), with information obtained by standard video cameras (which have low spatial-resolution but higher temporal resolution), to obtain an improved video sequence of high spatial and high temporal resolution.

Differences in the physical properties of temporal vs. spatial imaging lead to marked differences in performance and behavior of temporal SR vs. spatial SR. These include issues such as: the upper bound on improvement in resolution, optimal camera configurations, and more. These issues are also analyzed and discussed in this thesis.

The rest of this thesis is organized as follows: Sec. 2 describes our space-time SR algorithm. Sec. 3 shows some examples of handling motion aliasing and motion blur in dynamic scenes. Sec. 4 analyzes how temporal SR can treat motion blur, and provides a lower bound on the number of input cameras needed for effective motion deblurring. In Sec. 5 we discuss the visual tradeoffs between space and in time. Finally in Sec. 6 we analyze the commonalities and the differences between spatial SR and temporal SR.

2 Space-Time Super-Resolution

Let S be a dynamic space-time scene. Let $\{S_i^l\}_{i=1}^n$ be n video sequences of that dynamic scene recorded by n different video cameras. The recorded sequences have limited spatial and temporal resolution (the subscript "l" stands for "low" space-time resolution). Their limited resolutions are due to the space-time imaging process, which can be thought of as a process of blurring followed by sampling both in time and in space.

The blurring effect results from the fact that the color at each pixel in each frame (referred to as a "space-time point" and marked by the small boxes in Fig. 3.a) is an integral (a weighted average) of the colors in a space-time *region* in the dynamic scene S (marked by the large pink and blue boxes in Fig. 3.a). The temporal extent of this region is determined by the exposure-time of the video camera (i.e., how long the shutter is open), and the spatial extent of this region is determined by the spatial point-spread-function (PSF) of the camera (determined by the properties of the lens and the detectors [5]).

The sampling process also has a spatial and a temporal component. The spatial sampling results from the fact that the camera has a discrete and finite number of detectors (the output of each detector is a single pixel value), and the temporal sampling results from the fact that the camera has a finite frame-rate resulting in discrete frames (typically 25 $frames/sec$ in PAL cameras and 30 $frames/sec$ in NTSC cameras).

The above space-time imaging process inhibits high spatial and high temporal frequencies of the dynamic scene, resulting in video sequences of low space-time resolutions. Our objective is to use the information from all these sequences to construct a new sequence S^h

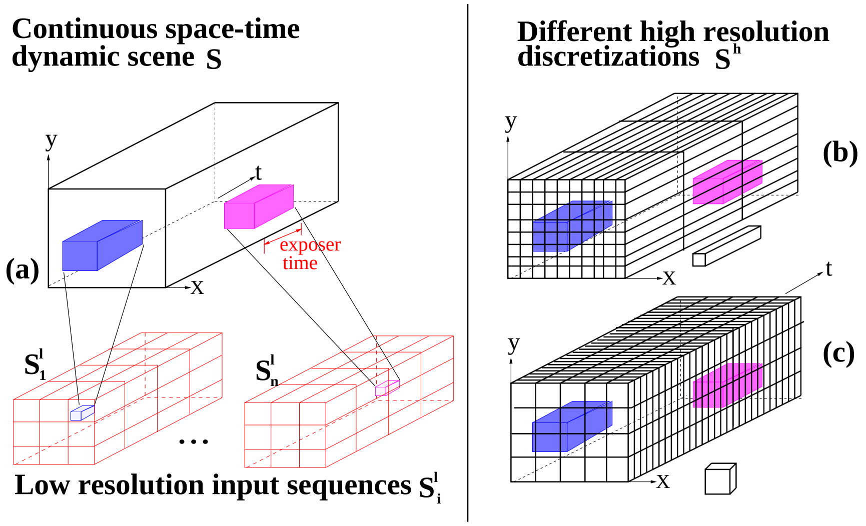


Figure 3: **The space-time imaging process.** *(a) illustrates the continuous space-time scene and two of the low resolution sequences. The large pink and blue boxes are the support regions of the space-time blur corresponding to the low resolution space-time measurements marked by the respective small boxes. (b,c) show two different possible discretizations of the continuous space-time volume S resulting in two different possible types of high resolution output sequences S^h. (b) has a low frame-rate and high spatial resolution, whereas (c) has a high frame-rate but low spatial resolution.*

of high space-time resolution. Such a sequence will ideally have smaller blurring effects and finer sampling in space and in time, and will thus capture higher space-time frequencies of the dynamic scene S. In particular, it will capture fine spatial features in the scene and rapid dynamic events which cannot be captured (and are therefore not visible) in the low-resolution sequences.

The recoverable high-resolution information in S^h is limited by its spatial and temporal sampling rate (or discretization) of the space-time volume. These rates can be different in space and in time. Thus, for example, we can recover a sequence S^h of very high spatial resolution but low temporal resolution (e.g., see Fig. 3.b), a sequence of very high temporal resolution but low spatial resolution (e.g., see Fig. 3.c), or a bit of both. These tradeoffs in space-time resolutions and their visual effects will be discussed in more detail later in Sec. 5.3.

We next model the geometrical relations (Sec. 2.1) and photometric relations (Sec. 2.2) between the unknown high-resolution sequence S^h and the input low-resolution sequences $\{S_i^l\}_{i=1}^n$.

2.1 The Space-time Coordinate Transformations

In general a space-time dynamic scene is captured by a 4D representation (x, y, z, t). For simplicity, in this thesis we deal with dynamic scenes which can be modelled by a 3D space-time volume (x, y, t) (see in Fig. 3.a). This assumption is valid if one of the following conditions holds: (i) the scene is planar and the dynamic events occur within this plane, or (ii) the scene is a general dynamic 3D scene, but the distances between the recording video cameras are small relative to their distance from the scene. (When the camera centers are very close to each other, there is no relative 3D parallax.) Under those conditions the dynamic scene can be modelled by a 3D space-time representation.

W.l.o.g., let S_1^l (one of the input low-resolution sequences) be a "reference" sequence. We define the coordinate system of the continuous space-time volume S (the unknown dynamic scene we wish to reconstruct), so that its x, y, t axes are parallel to those of the reference sequence S_1^l. S^h is a discretization of S with a higher sampling rate than that of S_1^l (see Fig. 3.b). Thus, we can model the transformation T_1 from the space-time coordinate system of S_1^l to the space-time coordinate system of S^h by a scaling transformation (the scaling can be different in time and in space). Let $T_{i\rightarrow 1}$ denote the space-time coordinate transformation from the i-th low resolution sequence S_i^l to the reference sequence S_1^l (see below). Then the space-time coordinate transformation of each low-resolution sequence S_i^l is related to that of the high-resolution sequence S^h by $T_i = T_1 \cdot T_{i\rightarrow 1}$.

The space-time coordinate transformation $T_{i\rightarrow 1}$ between two input sequences results from the different setting of the different cameras. A *temporal misalignment* between two video sequences occurs when there is a time-shift (offset) between them (e.g., if the two video cameras were not activated simultaneously), or when they differ in their frame rates (e.g., one PAL and the other NTSC). Such temporal misalignments can be modelled by a 1-D affine transformation in time, and is typically at sub-frame time units. The *spatial misalignment* between the two sequences results from the fact that the two cameras have different external and internal calibration parameters. In our current implementation, as mentioned above, because the camera centers are assumed to be very close or else the scene is planar, the spatial transformation between the two sequences can thus be modelled by an inter-camera homography (even if the scene is a cluttered 3D scene). We computed these space-time coordinate transformations using the method of [9], which provides high sub-pixel and high sub-frame accuracy.

Note that while the space-time coordinate transformations ($\{T_i\}_{i=1}^n$) *between the sequences* are very simple (a spatial homography and a temporal affine transformation), the motions occurring over time *within* each sequence (i.e., within the dynamic scene) can be very complex. Our space-time SR algorithm does *not* require knowledge of these complex intra-sequence motions, only the knowledge of the simple inter-sequence transformations $\{T_i\}_{i=1}^n$. It can thus handle very complex dynamic scenes. For more details see [9].

2.2 The Space-Time Imaging Model

As mentioned earlier, the space-time imaging process induces spatial and temporal blurring in the low-resolution sequences. The temporal blur in the low-resolution sequence S_i^l is caused by the exposure-time (shutter-time) of the i-th video camera (denoted henceforth by τ_i). The spatial blur in S_i^l is due to the spatial point-spread-function (PSF) of the i-th camera, which can be approximated by a 2D spatial Gaussian with std σ_i. (Methods to estimate the PSF of a camera can be found in [14, 8].)

Let $B_i = B_{(\sigma_i, \tau_i, p_i^l)}$ denote the combined space-time blur operator of the $i - th$ video camera corresponding to the low resolution space-time point $p_i^l = (x_i^l, y_i^l, t_i^l)$. Let $p^h = (x^h, y^h, t^h)$ be the corresponding high resolution space-time point $p^h = T_i(p_i^l)$ (p^h is not necessarily an integer grid point of S^h, but is contained in the continuous space-time volume S). Then the relation between the *unknown* space-time values $S(p^h)$, and the *known* low resolution space-time measurements $S_i^l(p_i^l)$, can be expressed by:

$$S_i^l(p_i^l) = (S * B_i^h)(p^h) = \underset{p\,=\,(x,\,y,\,t)\,\in\,Support(B_i^h)}{\int_x \int_y \int_t} S(p)\ B_i^h(p - p^h) dp \qquad (1)$$

where $B_i^h = T_i(B_{(\sigma_i, \tau_i, p_i^l)})$ is a point-dependent space-time blur kernel represented in the high resolution coordinate system. Its support is illustrated by the large pink and blue boxes in Fig. 3.a. To obtain a linear equation in the terms of the *discrete unknown* values of S^h we used a discrete approximation of Eq. (1). In our implementation we used a non-isotropic approximation in the temporal dimension, and an isotropic approximation in the spatial dimension. See [7] for a discussion of the different spatial discretization techniques in the context of image-based SR. See Appendix E for details about the discretization we used for the temporal blur kernel. Eq. (1) thus provides a linear equation that relates the unknown values in the high resolution sequence S^h to the *known* low resolution measurements $S_i^l(p_i^l)$.

When video cameras of different photometric responses are used to produce the input sequences, then a preprocessing step is necessary that histogram-equalizes all the low resolution sequences. This step is required to guarantee consistency of the relation in Eq. (1) with respect to all low resolution sequences.

2.3 The Reconstruction Step

Eq. (1) provides a single equation in the high resolution unknowns for each low resolution space-time measurement. This leads to the following huge system of linear equations in the unknown high resolution elements of S^h:

$$A\overrightarrow{h} = \overrightarrow{l} \tag{2}$$

where $\overrightarrow{h}$ is a vector containing all the unknown high resolution color values (in YIQ) of S^h, $\overrightarrow{l}$ is a vector containing all the space-time measurements from all the low resolution sequences, and the matrix A contains the relative contributions of each high resolution space-time point to each low resolution space-time point, as defined by Eq. (1).

When the number of low resolution space-time measurements in $\overrightarrow{l}$ is greater than or equal to the number of space-time points in the high-resolution sequence S^h (i.e., in $\overrightarrow{h}$), then there are more equations than unknowns, and Eq. (2) can be solved using LSQ methods. This, however, implies that a large increase in the spatial resolution (which requires very fine spatial sampling in S^h) will come at the expense of a significant increase in the temporal resolution (which also requires fine temporal sampling in S^h), and vice versa. This is because for a given set of input low-resolution sequences, the size of $\overrightarrow{l}$ is fixed, thus dictating the number of unknowns in S^h. However, the number high resolution space-time points (unknowns) can be distributed differently between space and time, resulting in different space-time resolutions (This issue is discussed in more detail in Sec. 5.3).

Directional space-time regularization: When there is an insufficient number of cameras relative to the required improvement in resolution (either in the entire space-time volume, or only in portions of it), then the above set of equations (2) becomes ill-posed. To constrain the solution and provide additional numerical stability (as in image-based SR [11, 6]), a space-time regularization term can be added to impose smoothness on the solution S^h in space-time regions which have insufficient information. We introduce a *directional* (or steerable [16]) space-time regularization term which applies smoothness only in directions within the space-time volume where the derivatives are low, and does *not* smooth across space-time "edges". In other words, we seek $\overrightarrow{h}$ which minimize the following error term:

$$min(||A\overrightarrow{h} - \overrightarrow{l}||^2 + ||W_x L_x \overrightarrow{h}||^2 + ||W_y L_y \overrightarrow{h}||^2 + ||W_t L_t \overrightarrow{h}||^2) \tag{3}$$

Where L_j $(j = x, y, t)$ is matrix capturing the second-order derivative operator in the direction j, and W_j is a diagonal weight matrix which captures the degree of desired regularization at each space-time point in the direction j. The weights in W_j prevent smoothing across space-time "edges". These weights are determined by the location, orientation and magnitude of space-time edges, and are approximated using space-time derivatives in the low resolution sequences.

Solving the equation: The optimization problem of Eq. (3) has a very large dimensionality. For example, even for a simple case of four low resolution input sequences, each of one-second length(25 frames) and of size 128×128 pixels, we get: $128^2 \times 25 \times 4 \approx 1.6 \times 10^6$ equations from the low resolution measurements alone (without regularization). Assuming a similar number of high resolution unknowns poses a severe computational problem. However, because the matrix A is sparse and local (i.e., all the non zero entries are located in a few diagonals), the system of equations can be solved using "box relaxation" [22]. For more details see Appendix B.

3 Examples of Temporal Super-Resolution

Before proceeding with more in-depth analysis and details, we first show a few examples of applying the above algorithm for recovering higher *temporal* resolution of fast dynamic events. In particular, we demonstrate how this approach provides a solution to the two previously mentioned problems encountered when fast dynamic events are recorded by slow video cameras: (i) motion aliasing, and (ii) motion blur.

Example 1: Handling Motion Aliasing

We used four independent PAL video cameras to record a scene of a fan rotating clockwise very fast. The fan rotated faster and faster, until at some stage it exceeded the maximal velocity that can be captured correctly by the video frame-rate. As expected, at that moment all four input sequences display the classical "wagon wheel effect" where the fan appears to be falsely rotating backwards (counter clock-wise). We computed the spatial and temporal misalignments between the sequences at sub-pixel and sub-frame accuracy using [9] (the recovered temporal misalignments are displayed in Fig. 4.a-d using a time-bar). We used the SR method of Sec. 2 to increase the temporal resolution by a factor of 3, while maintaining the same spatial resolution. The resulting high-resolution sequence displays the true forward (clock-wise) motion of the fan, as if recorded by a high-speed camera (in this case, 75 frames/sec). Example of a few successive frames from each low resolution input sequence are shown in Fig.4.a-d for the portion where the fan falsely appears to be rotating counter clock-wise. A few successive frames from the reconstructed high temporal-resolution sequence corresponding to the same time are shown in Fig.4.e, showing the correctly recovered (clock-wise) motion. It is difficult to perceive these strong dynamic effects via a static figure. We therefore urge the reader to view the video clips in www.wisdom.weizmann.ac.il/∼vision/SuperRes.html, where these effects are very vivid.

Note that playing the input sequences in "slow-motion" (using any type of temporal interpolation) will *not* reduce the perceived false motion effects, as the information is already lost in any individual video sequence (as illustrated in Fig. 2). It is only when the information is combined from all the input sequences, that the true motion can be recovered.

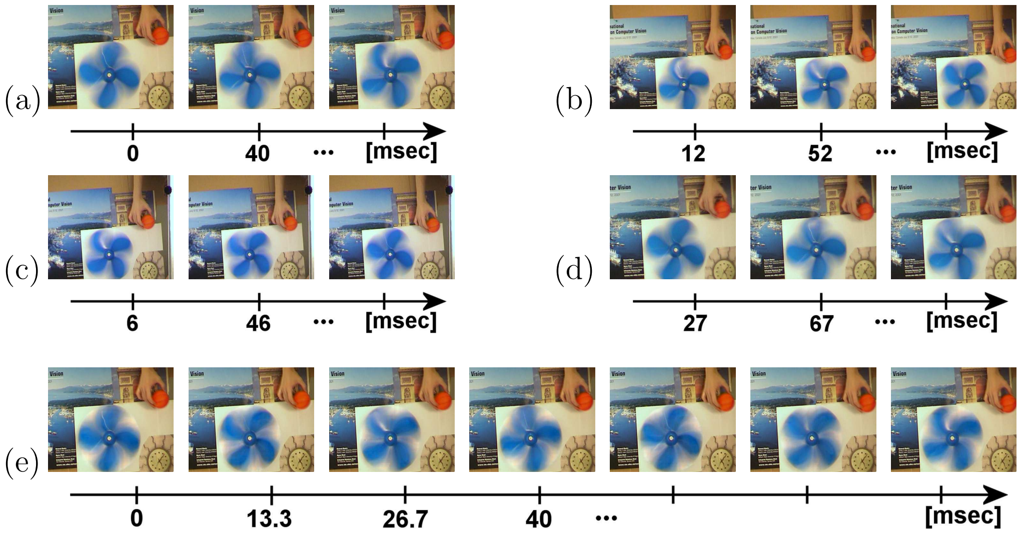


Figure 4: **Example 1: Handling motion aliasing - The "wagon wheel effect".** *(a)-(d) display 3 successive frames from four PAL video recordings of a fan rotating clock-wise. Because the fan is rotating very fast (almost* 90^o *between successive frames), the motion aliasing generates a false perception of the fan rotating slowly in the opposite direction (counter clock-wise) in all four input sequences. The temporal misalignments between the input sequences were computed at sub-frame temporal accuracy, and are indicated by their time bars. The spatial misalignments between the sequences (e.g., due to differences in zoom and orientation) were modeled by a homography, and computed at sub-pixel accuracy. (e) shows the reconstructed video sequence in which the temporal resolution was increased by a factor of 3. The new frame rate* ($75\frac{frames}{sec}$) *is also indicated by a time bars. The correct clock-wise motion of the fan is recovered. For video sequences see: www.wisdom.weizmann.ac.il/∼vision/SuperRes.html*

Example 2: Handling Motion Blur

In the following example we captured a scene of fast moving balls using 4 PAL video cameras of 25 frames/sec and exposure-time of 40 msec. Fig. 5.a-d shows 4 frames, one from each low-resolution input sequence, that were the *closest* to the time of collision of the two balls. In each of these frames at least one of the balls is blurred. We applied the SR algorithm and increased the frame-rate by factor 4. Fig. 5.e shows an output frame at time of collision. Motion-blur is reduced significantly. Such a frame did not exist in any of the input video sequences. Note that this effect was obtained by increasing the *temporal* resolution (not the spatial), and hence did *not* require the estimation of the motions of the balls. This phenomena is explained in more details in Sec. 4.

To examine the capabilities of the algorithm treating *severe* effects of motion-blur of the kind shown in Fig. 1, one needs many (usually more than 10) video cameras. A quantitative analysis of the amount of input data needed appears in Sec. 4. Since we didn't have so many cameras we used simulation as described in the next example.

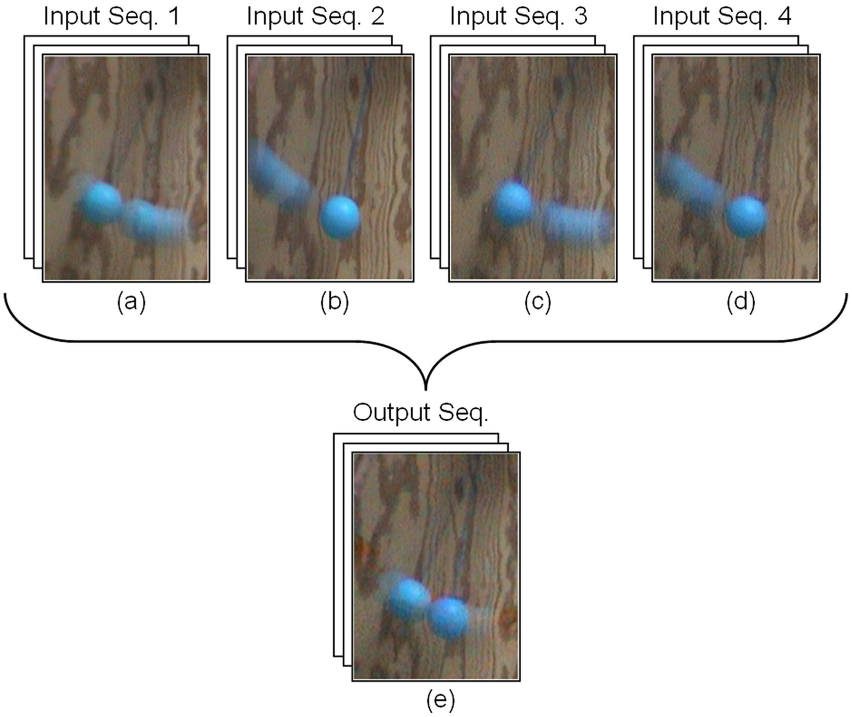


Figure 5: **Example 2: Handling motion blur via temporal SR.** *A "tic-tac" toy (2 balls hanging on strings and bouncing against each other) was shot by 4 video cameras. (a)-(d) display the 4 frames one from each of the input sequences, which were closest to the time of collision. In each one of these frames, at least one of the balls is blurred. The 4 input sequences were plugged into the temporal SR algorithm and the frame-rate was increased by a factor of 4. (e) shows the frame from the output, closest to the time of collision. Motion-blur is evidently reduced.*

Example 3: Handling Motion Aliasing & Motion Blur

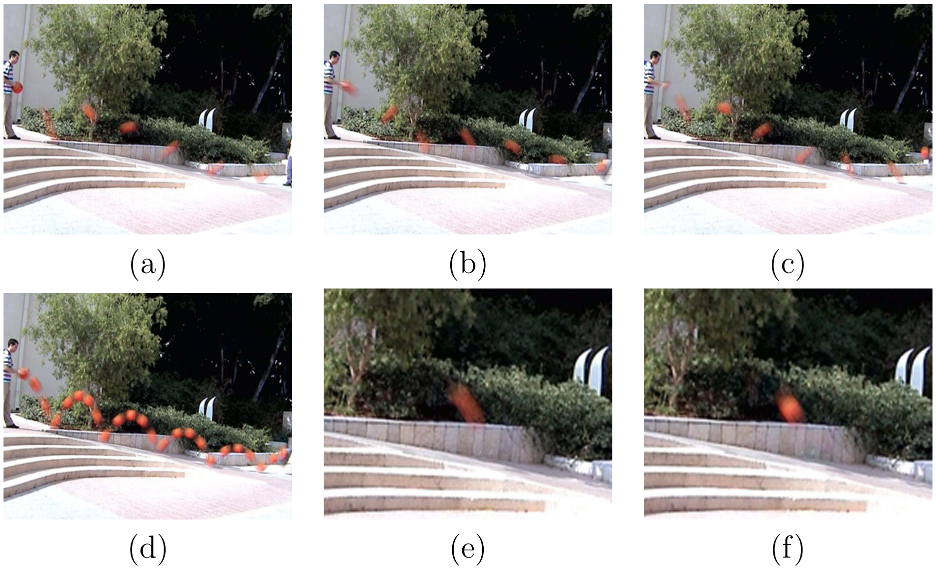
In the following example we simulated a sports-like scene with an extremely fast moving object (of the type shown in Fig. 1) recorded by many video cameras (in our example - 18 cameras). We examined the capabilities of temporal SR in the presence of both strong motion aliasing and strong motion blur.

To simulate such a scenario, we recorded a single video sequence of a slow moving object (a basketball bouncing on the ground). To simulate high speed of the ball relative to frame-rate and relative to the exposure-time, we temporally blurred the sequence using a large (9-frame) blur kernel, followed by a large subsampling in time by factor of $1 : 30$. Such a process results in a low temporal-resolution sequence of a very fast dynamic event having an "exposure-time" of about $\frac{1}{3}$ of its frame-time. We generated 18 such low resolution sequences by starting the temporal sub-sampling at *arbitrary* starting frames. Thus, the input low-resolution sequences are related by *non-uniform* sub-frame temporal offsets. Because the original video sequence contained 250 frames, each generated "low-resolution" sequence contains only 7 frames. Three of the 18 sequences are presented in Fig 6.a-c. To visually display the event captured in each of these sequences, we super-imposed all 7 frames in each sequence. Each ball in the super-imposed image represents the location of the ball at a different frame. None of the 18 low resolution sequences captures the correct trajectory of the ball. Due to the severe motion aliasing, the perceived ball trajectory is roughly a smooth curve, while the true trajectory was more like a cycloid (the ball jumped 5 times on the floor). Furthermore, the shape of the ball is completely distorted in all input image frames, due to the strong motion blur.

We applied the SR algorithm of Sec. 2 on these 18 low-resolution input sequences, and constructed a high-resolution sequence whose frame-rate is 30 times higher than that of the input sequences. (In this case we requested an increase only in the temporal sampling rate). The reconstructed high-resolution sequence is shown in Fig. 6.d. This is a super-imposed display of some of the reconstructed frames (every 8'th frame). The true trajectory of the bouncing ball has been recovered. Furthermore, Figs. 6.e-f show that this process has significantly reduced effects of motion blur and the true shape of moving ball has been automatically recovered, although no single low resolution frame contains the true shape of the ball. Note that no estimation of the ball motion was needed to obtain these results. This effect is explained in more details in Sec. 4.2.

The above results obtained by temporal SR cannot be obtained by playing any low-resolution sequence in "slow-motion" due to the strong motion aliasing. Moreover, such results cannot be obtained by interleaving frames from the 18 input sequences, due to the non-uniform time shifts between the sequences and due to the severe motion-blur observed in the individual image frames.

A method for treating motion blur in the context of *image-based* SR was proposed by [2, 1]. However, these methods require a prior segmentation of moving objects and the estimation of their motions. These methods will have difficulties handling complex motions or motion aliasing. The distorted shape of the object due to strong blur (e.g., Fig. 1) will pose severe problems in motion estimation. Furthermore, in the presence of motion aliasing, the direction of the estimated motion will not align with the direction of the induced blur. For example, the motion blur in Fig. 6.a-c. is along the true trajectory and not along the perceived one. In contrast, our approach does not require separation of static and dynamic scene components, nor their motion estimation, thus can handle very complex scene dynamics. However, we require multiple cameras. These issues are explained in more details in sec. 4.


(a) (b) (c)

(d) (e) (f)

Figure 6: **Example 3: Handling motion blur & motion aliasing.** *We simulated 18 low-resolution video recordings of a rapidly bouncing ball inducing strong motion blur and motion aliasing (see text). (a)-(c) Display the dynamic event captured by three representative low-resolution sequences. These displays were produced by super-position of all 7 frames in each low-resolution sequence. All 18 input sequences contain severe motion aliasing (evident from the falsely perceived curved trajectory of the ball) and strong motion blur (evident from the distorted shapes of the ball). (d) The reconstructed dynamic event as captured by the recovered high-resolution sequence. The true trajectory of the ball is recovered, as well as its correct shape. (e) A close-up image of the distorted ball in one of the low resolution sequences. (f) A close-up image of the ball at the exact corresponding frame in time in the high-resolution output sequence. For video sequences see: www.wisdom.weizmann.ac.il/∼vision/SuperRes.html*

4 Temporal Treatment of Spatial Artifacts

When an object moves fast relative to the exposure time of the camera, it induces observable motion-blur (e.g., see Fig. 1). The perceived distortion is spatial, however the cause is temporal. We next show that by increasing the *temporal* resolution we can handle the *spatial* artifacts caused by motion blur.

4.1 Why is Temporal Treatment Enough?

The camera integrates over its exposure time any temporal changes in the intensity of a point. Temporal changes in intensity can be caused by motion, by changes in illumination, etc. In Fig. 7 we show the intensity changes of one pixel that observes a bouncing ball (Fig. 7.a). Temporal integration of the camera may be expressed as a convolution with a rectangular function over time (Fig. 7.b). The result is a “smearing” of the intensity changes in each point/pixel over time. The percieved global visual effect is spatial, in

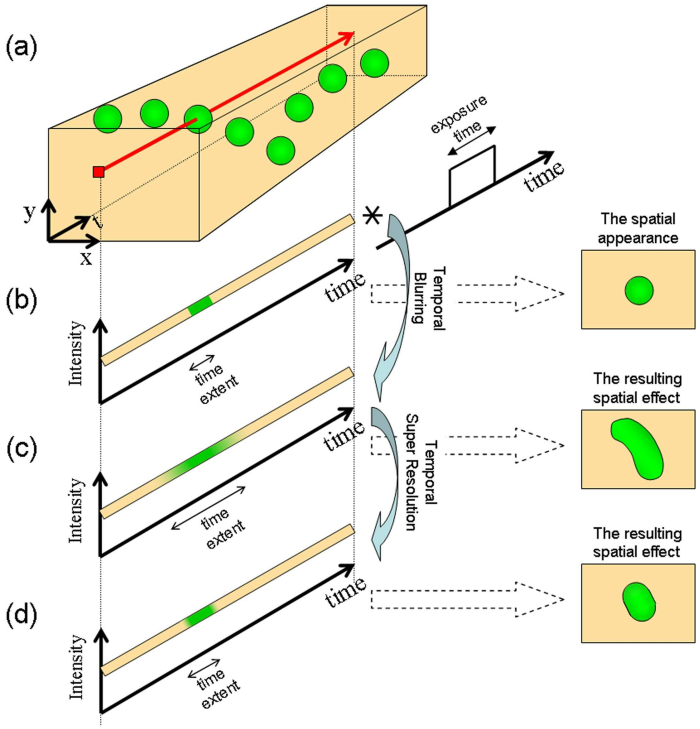


Figure 7: **Motion-blur - pixel-wise temporal blurring.** *Consider the space-time volume in (a) and the dynamic event of the falling ball in front of a static background. We will concentrate on the effect of motion-blur on the observed intensity at the red pixel as a function of time. The intensity (color) profile over time of that pixel is shown in (b). A frame of the ball when it crosses the pixel is shown on the right. The integration operation of the camera can be modelled by convolution with a rectangular kernel. The convolution result in (c) shows that the intensity profile is smoothed, and the temporal extent of the dynamic event is increased. The resulting visual effect is the elongated shape on the right. More pixels "observe" the ball during the exposure-time, causing the motion-blur effect. Fig. (d) shows that by applying SR in time, we can reduce the temporal extent of the intensity, and therefore reduce the visual effect of motion-blur (on the right). Therefore we can treat the spatial effect of motion-blur by temporal SR , without any motion estimation of the ball.*

the form of an elongated blob along the trajectory, since many pixels "experience" the integration (Fig. 7.b on the right). Fig. 7.d shows that by applying SR in time, we can reduce the temporal extent of the intensity, and therefore reduce the visual effect of motion-blur (on the right).

The crucial observation is that we can treat the *spatial* effect of motion-blur by decreasing the *temporal* blur in each pixel independently. Trying to treat motion-blur spatially (e.g., by applying spatial filters to the images), would require *different filters* for different motions. However, when motion blur is treated temporally, the *same operation* is applied to the entire space-time volume, without the need to detect or compute the different motions.

This allows for treatment of motion blur in very complex dynamic scenes.

We show next that by applying our temporal SR algorithm, the “effective” support of the temporal blur in the output sequence can be decreased relative to the exposure-time of the input cameras, leading to a reduction in motion blur. However, this effect is achieved only if there is a minimal increase in the temporal sampling rate of the output. If we do not increase the output temporal sampling-rate *enough*, we will not obtain a decrease in the motion blur. Moreover, an insufficient increase in the temporal sampling-rate might introduce *additional* motion-blur. This dictates the minimum number of input cameras needed for an effective decrease in the motion-blur.

4.2 What is the Minimum Number of Required Video Cameras?

The reconstructed high resolution sequence represents the ideal continuous signal convolved with a new temporal blur kernel. In Appendix A we derive an approximation to the “effective” exposure-time τ_{out} of the output sequence of the SR algorithm.

The residual temporal blur in the high resolution output sequence depends on its frame-rate FR_{out} in the following way:

$$\tau_{out} \approx \frac{1}{FR_{out}}$$

Note that the output frame-rate cannot be increased by a factor larger than the number of input sequences, since we must have more measurements (equations) than unknowns, i.e.,

$$FR_{out} \leq FR_{in} \cdot N_{cam}$$

Given the above observation we obtain the following practical connection between the required number of input video sequences (cameras) N_{cam}, and the desired exposure time of the high resolution output sequence. Assuming that all the input cameras have the same frame-rate and the same exposure-time τ_{in}, and assuming the optimal case where the cameras are activated at uniform gaps *in time*, then:

$$\tau_{out} \approx \frac{1}{FR_{out}} \geq \frac{1}{FR_{in} \cdot N_{cam}} \tag{4}$$

To get an effective decrease in the motion-blur, τ_{out} should be smaller than the exposure time of the input sequences τ_{in}, i.e.,

$$\tau_{in} > \tau_{out} \tag{5}$$

Combining Eqs. (4) and (5) we obtain the following lower bound on the number of video cameras required to obtain an effective reduction in the motion blur:

$$N_{cam} > \frac{1}{FR_{in} \cdot \tau_{in}} \tag{6}$$

We verified this bound empirically using the example of Fig. 6. The relation between the exposure-time and the frame-rate was $\tau_{in} = \frac{1}{3 \cdot FR_{in}}$. In this case the lower bound on the number of cameras for motion-blur reduction is $N_{cam} > 3$. Recall that in this example we increased the frame-rate by a factor of 15 ($N_{cam} = 15$) input cameras[1]. This explains the prominent effect of motion deblurring in the example of Fig. 6.f.

Now suppose we had fewer than 15 input cameras, then we would be bound to increase the frame-rate by a factor smaller than 15. Fig. 8 shows the resulting motion deblurring of the basketball when the number of input cameras is 15, 10, 5, 2 and 1, respectively (the output frame-rate is determined accordingly). We can see that the bound of Eq. (6) is validated. When the number of input cameras exceeds the lower bound of Eq. (6) (Fig. 8.a-d), then the effects of motion-blur are decreased relative to the input sequences. However, when there are only 2 input cameras (Fig. 8.e), the visual effect of the motion-blur is more pronounced in the output sequence than that observed in the original input sequences! Moreover, when we have only one input camera (Fig. 8.f), and the output frame-rate equals to the input frame-rate, the motion-blur is significantly increased, and the output sequence is visibly poorer than the input sequence.

This example shows that in order to achieve an effective motion-deblurring in standard video recording where the exposure-time τ_{in} is approximately $\frac{1}{3 \cdot FR_{in}} \approx \frac{1}{75} sec$, at least 3 cameras (preferably more) are needed. Shorter exposure-time τ_{in} will dictate using more input cameras for decreasing motion-blur. A real example showing that the output motion-blur might be degraded if the output frame-rate is not increased enough, appears in Fig. 11.c.3 where the bound is $N_{cam} > 3$ and the frame-rate is increased only by a factor of 2.

[1] In fact we had 18 input cameras, but since the "effective" exposure-time is determined by the output frame-rate, having *more* input cameras than needed *cannot* improve motion-deblurring

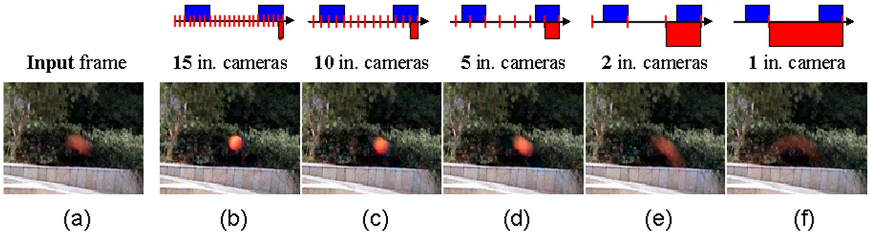


Figure 8: **Motion deblurring vs. number of input sequences.** *In (a) appear a motion-blurred ball fragment from one of the input sequences in Fig. 6. The relation between the exposure-time (marked by the blue rectangles) and the frame-rate was* $\tau_{in} = \frac{1}{3 \cdot FR_{in}}$*, and the lower bound on the number of cameras* $N_{cam} > 3$ *. We have tried to reconstruct output sequences the following number of input cameras - 15, 10, 5, 2 and 1 (Figs. (b)-(f) respectively). The "effective" exposure-time in the output sequence is marked by the red rectangles). As can be seen, the amount of motion-blur increases as we use more input cameras as explained in the text. Note also that if we are not above the lower bound, and the "effective" exposure-time of the output is larger than the exposure-time in the input (Figs. (e),(f)), than we get larger motion-blur in the high resolution sequence than in any one of the input low resolution sequences.*

5 Space-Time Visual Tradeoffs

The spatial and temporal dimensions are very different in nature, yet are inter-related. This introduces visual tradeoffs between space and time, which are unique to spatio-temporal SR, and are not applicable to traditional spatial (i.e., image-based) SR.

5.1 Combining Different Space-Time Inputs

So far we assumed that all input sequences were of similar spatial and temporal resolutions. The space-time SR algorithm of Sec. 2 is not restricted to this case, and can also handle input sequences of varying space-time resolutions. Such a case is meaningless in *image-based* super-resolution SR (i.e., combining information from *images* of varying spatial resolution), because a high resolution input image would always contain the information of a low resolution image. In space-time SR, however, this is not the case. One camera may have high spatial resolution but low temporal resolution, and the other vice-versa. Thus, for example, it is meaningful to combine information from NTSC and PAL video cameras. NTSC has higher temporal resolution than PAL (30 frames/sec vs. 25 frames/sec), but lower spatial resolution (640×480 pixels vs. 768×576 pixels). An extreme case of this idea is to combine information from *still* and *video* cameras. Such an example is shown in Fig. 9. Two high quality still images (Fig. 9.a) of high spatial resolutions (1120×840 pixels) but extremely low "temporal resolution" (the time gap between the two still images was 1.4 sec), were combined with an interlaced (PAL) video sequence using the algorithm of Sec. 2. The video sequence (Fig. 9.b) has 3 times lower spatial resolution (we used fields of size 384×288 pixels), but a high temporal resolution (50 frames/sec). The goal is to construct a new sequence of high spatial and high temporal resolutions (i.e., 1120×840 pixels at 50 frames/sec). The output sequence shown in Fig. 9.c contains the high spatial resolution from the still images (the sharp text) and the high temporal resolution from the video sequence (the rotation of the toy dog and the brightening and dimming of illumination).

In the example of Fig. 9 we used only one input video sequence and two still images, thus we did not attempt to exceed the temporal resolution of the video or the spatial resolution of the stills. However, when multiple video sequences and multiple still images are used (so that the number of input measurements exceeds the number of output high resolution unknowns), then an output sequence can be recovered, that exceeds the spatial resolution of the still images and temporal resolution of the video sequences.

In the example of Fig. 9, the number of unknowns was significantly larger than the number of low resolution measurements (the input video and the two still images). Although theoretically this is an ill-posed set of equations, the reconstructed output is of high quality. This is achieved by applying physically meaningful space-time directional regularization, that exploits the high redundancy in the video sequence. This issue is further discussed in Sec. 5.2.

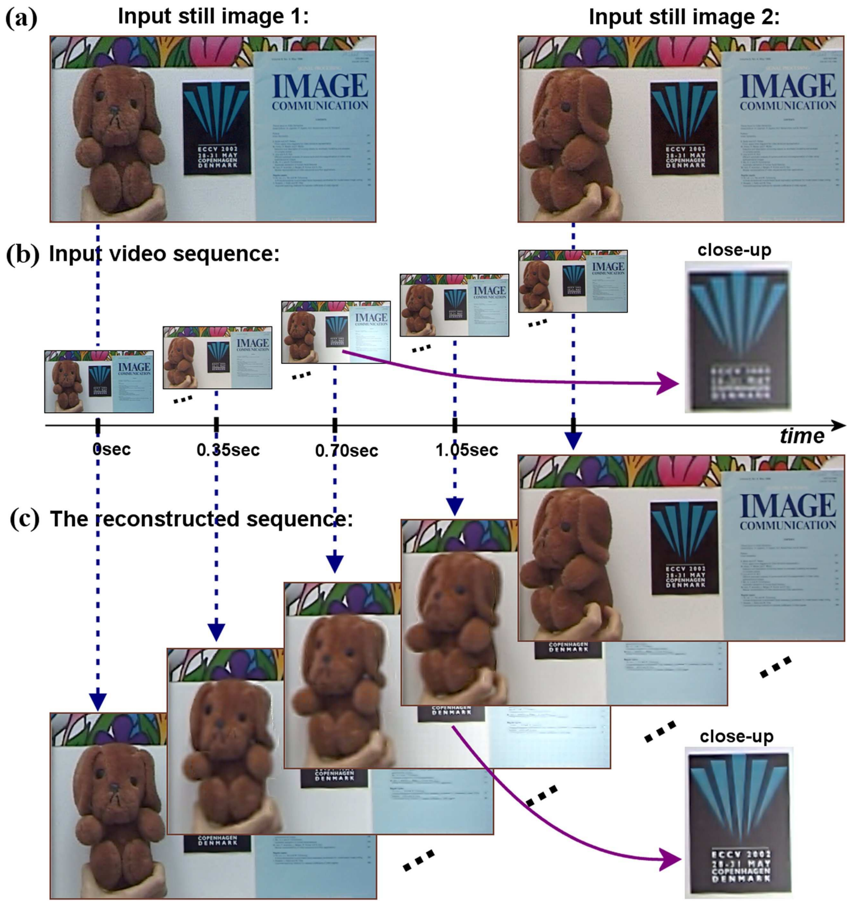


Figure 9: **Combining Still and Video.** *A dynamic scene of a rotating toy-dog and varying illumination was captured by: (a) A still camera with spatial resolution of* 1120×840 *pixels, and (b) A video camera with* 384×288 *pixels at 50 f/sec. The video sequence was* 1.4*sec long (*70 *frames), and the still images were taken* 1.4*sec apart (together with the first and last frames). The algorithm of Sec. 2 is used to generate the high resolution sequence (c). The output sequence has the spatial dimensions of the still images and the frame-rate of the video (*$1120\times 840\times 50$*). It captures the temporal changes correctly (the rotating toy and the varying illumination), as well the high spatial resolution of the still images (the sharp text). Due to lack of space we show only a portion of the images, but the proportions between video and still are maintained. For video sequences see: www.wisdom.weizmann.ac.il/∼vision/SuperRes.html*

5.2 Space-Time Regularization

Video sequences, as opposed to images, have an additional temporal dimension, that increases the data redundancy. The data redundancy in the space-time volume (x, y, t) is significantly larger than the redundancy in spatial information (x, y) alone. The redundancy provides more flexibility in applying physically meaningful space-time directional regularization. This will be demonstrated on the example of Fig. 4.

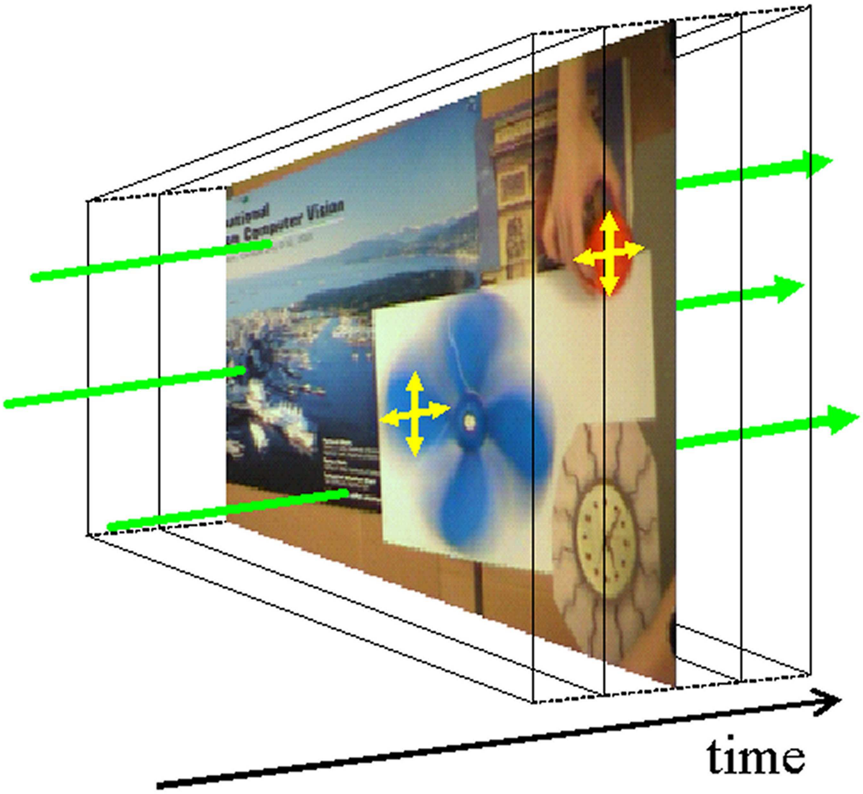


Figure 10: **Space-Time Regularization.** *The above figure shows the space-time volume with one high resolution resolution frame from the example of Fig. 4. In our algorithm we can apply space-time regularization in a physical meaningful way. In regions that have high spatial resolution but small (or no) motion (such as in the static background)the* temporal *regularization is strong (green arrow). Similarly, in regions with fast dynamic changes but low spatial resolution (such as in the rotating fan) the spatial regularization is strong (yellow arrows).*

In Fig. 10 we can see the space-time volume with one high resolution frame of that example. In this example, regions that have high spatial resolution but small (or no) motion (such as the static background), strong *temporal* regularization can be applied without decreasing the space-time resolution (the green arrows in Fig. 10). Similarly, in regions with fast dynamic changes but low spatial resolution (such as in the rotating fan), strong *spatial* regularization can be employed without degradation in space-time resolution (the yellow arrows in Fig. 10). More generally, because a video sequence has much more data redundancy than an image has, the use of *directional space-time regularization* in video-based SR is physically more meaningful and gives rise to recovery of higher space-time resolution than that obtainable by image-based SR with image-based regularization. More quantitative details about the regularization can be found in Appendix C.

5.3 Producing Different Space-Time Outputs

In standard spatial SR the increase in sampling rate is equal in all spatial dimensions. This is necessary in order to maintain the aspect ratio of image pixels, and to prevent distorted-looking images. However, this is not the case in space-time SR. As explained in Sec. 2, the increase in sampling rate in the spatial and temporal dimensions need not be the same. Moreover, increasing the sampling rate in the spatial dimension comes at the expense of increase in the temporal frame rate, and vice-versa. This is because the number of unknowns in the high-resolution space-time volume depends on the manner in which the space-time volume is discretized, whereas the number of equations provided by the low resolution measurements is fixed.

For example, assume that 8 video cameras are used to record a dynamic scene. One can increase the temporal frame-rate alone by a factor of 8 on increase the spatial sampling rate alone by a factor of $\sqrt{8}$ in x and in y (i.e., increase the number of pixels by a factor of 8), or do a bit of both: increase the sampling rate by a factor of 2 in all three dimensions x, y, t. Such an example is shown in Fig. 11. Fig. 11.a1 displays one of 8 low resolution input sequences. (Here we used only 4 video cameras, but split them into 8 sequences of even and odd fields). Figs. 11.a2 and 11.a3 display two possible outputs. In Fig. 11.a2 the increase is by a factor of 8 in the temporal axis with no increase in the spatial axes, and in Fig. 11.a3 the increase is by a factor of 2 in all three axes x,y,t. Rows (b) and (c) display the resulting visual tradeoffs. The "×1× 1× 8" option (column 2) decreases the motion blur of the moving object (the toothpaste in (c.2)), while the "×2× 2× 2" option (column 3) improves the spatial resolution of the static background (b.3). Note that although the temporal sampling rate (frame-rate) in column 3 was *increased* by a factor of 2, there was no decrease in the motion blur of the moving object. On the contrary, there was an *increase* in the motion blur of the toothpaste (c.3). The latter is because the increase in frame rate was only by factor 2 and did not exceed $\frac{1}{exposure\ time}$ of the video camera (see Sec. 4.2). In order to obtain any reduction in motion blur in this example, the minimum required increase in the frame-rate is 3.

5.4 Optimal Camera Configurations

The task of SR requires there to be sub-unit shifts between the input samples (sub-pixel shifts for spatial SR , and sub-frame shifts for temporal SR). So far (and in all our examples) we assumed the video sequences are recorded arbitrarily, and then the spatial and temporal misalignment between them are estimated (at sub-pixel and sub-frame accuracy) using [9]. However, if we could somehow control those spatio-temporal shifts in the acquisition process already, then we would ideally like to acquire them at uniform distances to obtain uniform sampling at the high-resolution space-time volume.

One of the differences between spatial SR and temporal SR tasks, is the ability to control the shifts between the input data source. It is very difficult to *a-priori* control the spatial "synchronization" between video cameras at *sub-pixel* accuracy, since the spatial misalignment depends on the relative positions and orientations of the cameras, their internal calibration parameters, and their distance from the scene. This, however, is not the

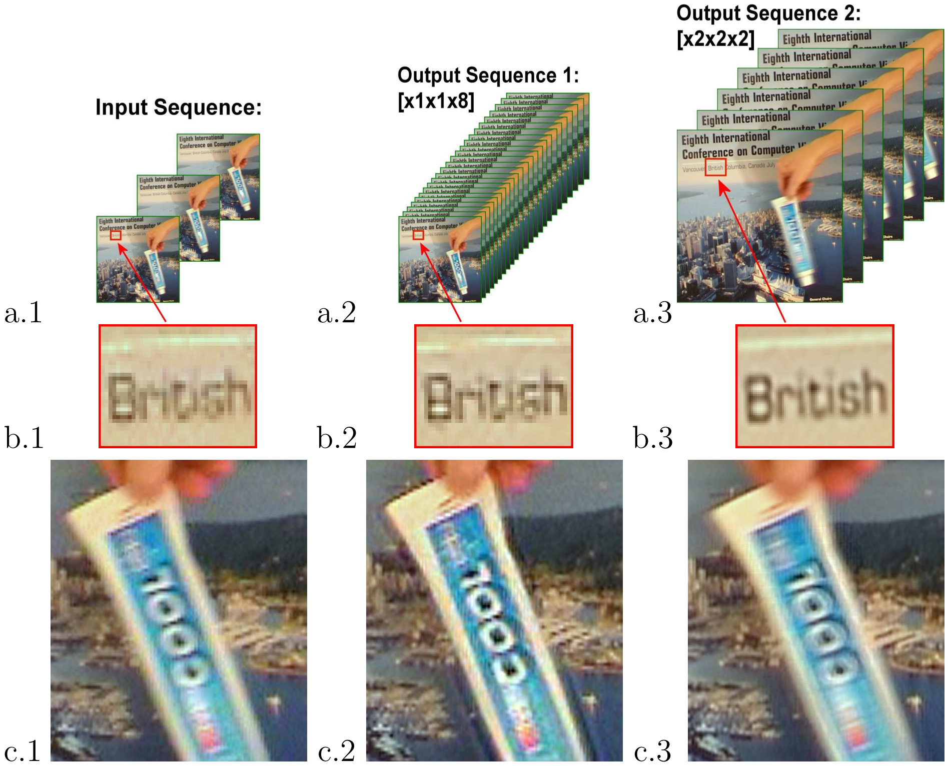


Figure 11: **Tradeoffs between spatial and temporal resolution.** *This figure compares the visual tradeoffs resulting from applying space-time super-resolution SR with different discretization of the space-time volume. (a.1) displays one of eight low-resolution input sequences of a toothpaste in motion against a static background. (b.1) shows a close-up image of a static portion of the scene (the writing on the poster), and (c.1) shows a dynamic portion of the scene (the toothpaste). Column 2 (a.2, b.2, c.2) displays the resulting spatial and temporal effects of applying SR by a factor of 8 in time only. Motion blur of the toothpaste is decreased. Column 3 (a.3, b.3, c.3) displays the resulting spatial and temporal effects of applying SR by a factor of 2 in all three dimensions x, y, t. The spatial resolution of the static portions is increased (see "British" and the yellow line above it in b.3), but the motion blur is also increased (c.3). See text for an explanation of these visual tradeoffs. For video sequences see: www.wisdom.weizmann.ac.il/∼vision/SuperRes.html*

case with sub-frame temporal synchronization. The temporal misalignments between the cameras are determined only by their activation time, and by their frame-rate (e.g., NTSC or PAL). The sub-frame shifts between the input sequences *can be controlled externally* at sub-frame accuracy with simple electronic devices. If the input sequences are of the same frame-rate, then an initial offset at half-frame time-unit would be enough, and the time-shifts between them will remain the same throughout the entire sequence.

The ability to synchronize the input videos gives rise to several practical optimal arrangement of the video equipment for the various SR tasks. Some of these are listed below:

- **Task 1: Increasing only temporal resolution -** In order to get the best improvement in temporal resolution, the video cameras should be temporally "synchronized" so that their temporal offsets are uniformly distributed in time. Such a configuration guarantees greatest numerical stability (or noise immunity) and fastest convergence of the linear system. We have shown analytically that the conditional number of the linear system increases if it is perturbed from the uniform temporal distribution. The uniform distribution is also needed for achieving similar decrease of the "induced" exposure time in all frames. We have also shown that in order to achieve best decrease in the "induced" exposure time, one should increase the frame-rate by the maximal factor, i.e. by the number of input sources.

- **Task 2: Increasing only spatial resolution -** In order to get best improvement in the spatial resolution both in static and in dynamic regions, one should fully synchronize all the video cameras to have *no* temporal offset between their sequences, and then increase only the spatial resolution. In this case the output frame-rate will remain the same as in the input sequences. This case is equivalent to perform the traditional spatial image-based SR using all the corresponding frames in time in all the input sequences.

- **Task 3: Capturing fast and dark events -** One possible application of the temporal SR may be to replace an expensive fast video camera (e.g., 300 frames/sec) by using several slow cheap video cameras. Except for the price matter, the main drawback of using the fast camera is its very short exposure-time (that is limited by the high frame-rate). Recording with a very fast video camera usually requires adding strong external illumination to the scene, to compensate for the short exposure time. Without such external light sources, the video would be noisy and dark objects would not be visible. However, it is not always possible to illuminate the scene with additional light sources. We claim that one can use several slow (or "regular") cameras with long exposure times and apply temporal SR to achieve the same effective exposure time at the output, without require additional artificial lighting.

6 Differences between Super Resolution in Time and in Space

The space-time resolution of a video sequence is determined by the blur and the subsampling of the camera. These have different characteristics in time and in space. The temporal blur induced by the exposure time has approximately the shape of a rectangular kernel, while the spatial blur has a Gaussian-like shape. Furthermore, the supports of the spatial and temporal blurs are very different. The spatial blur has typically a radius larger than one pixel (its "standard deviation" σ is approximately 1 pixel), whereas the exposure time is usually smaller than a single frame-time (i.e., $\tau <$frame-time). This is depicted in Fig. 12.

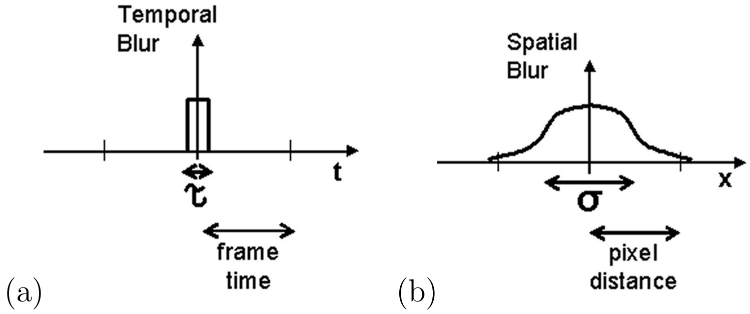


Figure 12: **Temporal vs. Spatial Blur Kernels.** *The temporal blur (a) induced by the exposure time has approximately the shape of a rectangular kernel, while the spatial blur (b) has a Gaussian-like shape (here shown only the x-axis). Furthermore, the supports of the spatial and temporal blurs are very different. The spatial blur has a radius of approximately one pixel (i.e., $\sigma \approx 1$ pixel), whereas the exposure time is usually smaller than a single frame-time (i.e., $\tau <$frame-time).*

The different support and shapes of the blur kernels result in much stronger temporal aliasing in the input sequences than spatial aliasing. This in turn leads to the following three differences between temporal and spatial SR :

(i) The upper bound on the possible increase in temporal resolution is significantly larger than the upper bound on the possible increase in spatial resolution.

(ii) Artifacts of temporal "ringing" in temporal SR are more prominent than spatial "ringing" in spatial SR .

(iii) All spatial samples in images are viewed simultaneously. However, the temporal samples are viewed sequentially in time. This leads to two different types of temporal aliasing: "motion aliasing" and "gray-level aliasing", whereas in space there is only "gray-level aliasing".

These issues are discussed in more detail in the next few sub-sections.

6.1 Upper Bound on Space-Time Super-Resolution

The upper limit of spatial SR has been discussed thoroughly in [3, 18]. Baker & Kanade [3] showed that the noise related to high frequencies which are not "suppressed" by the SR algorithm, grow quadratically with the SR magnification factor, and that large magnification factors are therefore not practical. Lin & Shum [18] showed using the least squares perturbation theorem that in practical conditions the maximal effective SR magnification factor for real images is 1.6 (and a theoretical factor of 5.7 is claimed for synthetic images with very low noise). Other image-based SR algorithms (e.g. [1, 2, 6, 7, 11, 13, 14, 15, 16]) were also performed with limited magnification factors (usually up to 2).

Because of the differences between the spatial and temporal properties of the imaging process, the upper-bound on *temporal* SR is significantly larger than the upper-bound on *spatial* SR. In other words, the temporal resolution can be increased effectively by higher magnification factors than the spatial resolution. We refer here to an actual increase in resolution, i.e., a decrease in the width (support) of the blurring kernel (the exposure-time in the case of temporal SR, and the point spread function in the case of spatial SR), and not an artificial increase of the frame-rate/sampling-rate.

The reason for this difference has to do with the rectangular *shape* of the temporal blur and the extent of its *support*. When the blur function is an "ideal" low-pass filter, no SR can be obtained, since *all* high frequencies are eliminated in the blurring process. However, when the blur function is *not* an ideal low-pass filter, high frequencies are not completely

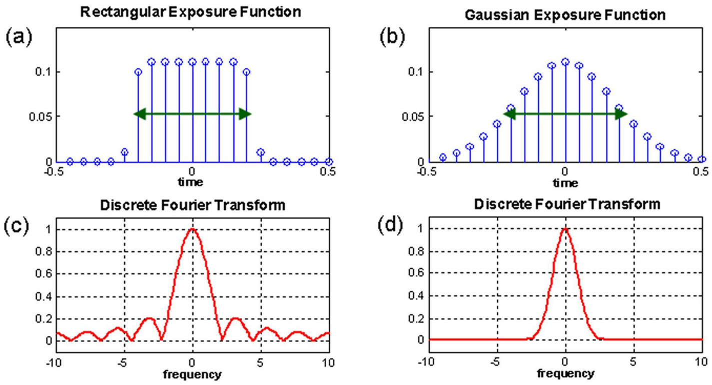


Figure 13: **Frequency support of the rectangular and gaussian blurs.** *Two discrete exposure functions are presented: a rectangular kernel (a) and a Gaussian kernel (b). Both kernels are with the same width (marked by the green arrows), and with the same "energy" (the sum of the discrete weights in each one of them is 1). Their absolute Discrete Fourier Transform (DFT) is shown in (c) and (d) correspondingly. We can see that due to its rectangular shape, the frequency spectrum of the rectangular kernel has a sinc shape (c), while the frequency shape of the Gaussian kernel is Gaussian (d). The frequency support of the rectangular kernel is larger than the Gaussian kernel. Therefore, blurring a signal with a rectangular function causes higher frequencies to be preserved (in aliased form due to sampling), that can be resolved by SR.*

eliminated and are found in aliased form in the low resolution data. It is those frequencies that are recovered in the SR process. The spatial blur function (the point spread function) has a Gaussian shape, and its support extends over *several* pixels (samples). As such, it a much "stronger" low-pass filter than the temporal blur function (the exposure time), which has a rectangular shape, and whose extent is *sub*-frame (i.e., less than one sample). The spatial blur function thus eliminates more high frequencies. Fig. 13 compares the frequency support of typical Gaussian and rectangular blurs of approximately the same width. We can see that the rectangular kernel contains much many high frequencies than the Gaussian one. Shrinking its width (support) in Fig. 13.a will result in a streching of its Fourier transform, thus allowing for even higher frequencies. Therefore more aliased frequencies can be reconstructed by temporal SR than by spatial SR.

Several empirical examples were shown (Fig. 8, Fig. 16) where the "induced" exposure-time was decreased by a factor of up to 5. Fig. 14 shows that significant larger magnification factors can be expected in temporal SR.

To show this, we took 4 sets of 30 input sequences with different exposure times. Each set was synthetically generated by temporal blurring followed by subsampling, similarly to the way described in Sec. 3. Small Gaussian noise was added to the input sequences in a way that in all of them the temporal noise would be the same ($\sigma \approx 2.5$ gray-levels). Fig. 14.a shows one frame of the ball in the original basketball sequence (before temporal blurring). Figs. 14.b-e show matching frames from each set of the simulated sequences with exposure times: $\tau_{in,1} = \frac{5}{30}$, $\tau_{in,2} = \frac{13}{30}$, $\tau_{in,3} = \frac{21}{30}$ and $\tau_{in,4} = \frac{29}{30}$, respectively (in units of $\frac{1}{FR_{in}}$). We increased the frame-rate by factor 15 in each of the sets using the temporal SR algorithm. *No regularization* was applied to show the "pure" output of the temporal SR algorithm without any smoothing. Figs. 14.f-i are the corresponding frames in the reconstructed sequences. The measured noise was amplified linearly (from 5.6 to 14.7) with

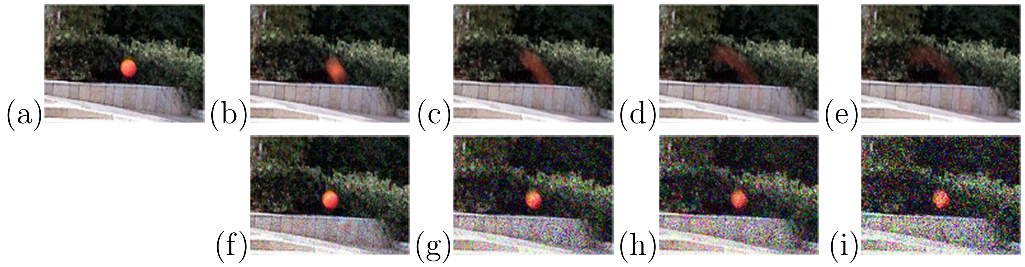


Figure 14: **Temporal SR with large magnification factors.** *In the following example we simulated 4 sets of 30 sequences with different exposure times for each set. (a) One frame of the ball in the original basketball sequence (before temporal blurring). (b)-(e) The corresponding frame from each set of the simulated low resolution sequences. (f)-(i) The corresponding frames in the reconstructed sequence with frame-rate increased by a factor of 15. The resulting SR magnification factors (of temporal resolution) for the four sets are:* $M_1 = 2.5$, $M_2 = 6.5$, $M_3 = 10.5$ *and* $M_4 = 14.5$, *respectively.*

the SR magnification factor. However, the residual motion-blur in the output sequences is small and similar regardless of the SR magnification (the reconstructed shape of the ball is correct). Hence, the "induced" exposure behaves according to Eq. (4), i.e $\tau_{out} \approx 1/15$ in all output sequences. The resulting SR magnification factors ($M_k = \tau_{in,k}/\tau_{out} \;\; k = 1..4$) for each of the four sets are: $M_1 = 2.5$, $M_2 = 6.5$, $M_3 = 10.5$ and $M_4 = 14.5$, respectively.

The upper bound on SR depends on the allowable output noise. However it is evident that typical magnifications factors of temporal SR are likely to be larger than in spatial super resolution. Furthermore, temporal noise is integrated by the eye and is therefore more tolerable than spatial noise. Adding temporal regularization would reduce the output noise, but would also increase τ_{out}, and therefore will not increase the upper bound (as was similarly shown in the analysis for the spatial case [3, 18]).

6.2 "Ringing" Effects

So far we have shown that the rectangular shape of the blur kernel has an advantage over a Gaussian shape in the ability to increase resolution. On the other hand the rectangular shape of the temporal blur is more likely to introduce a temporal artifact which is similar to the spatial "ringing" ([11, 7, 3]). This effect is expressed in temporal super-resolved video sequences as a trail that is moving before and after the fast moving object. We refer to this temporal effect as "ghosting". Fig. 15.a-c shows an example of the "ghosting" effect resulting in the basketball example when temporal SR is applied *without* any space-time regularization. (The effect is magnified by factor of 5 to make it more visible).

The explanation of the "ghosting" effect is simple if we look in the frequency domain. The SR algorithm (spatial or temporal) can reconstruct correctly the true temporal signal in the entire spectrum domain except for those specific frequencies that have been set to zero by the temporal rectangular blur. The system of equations (2) will not provide any constraints on those frequencies. If such frequencies are somehow "born" in the iterative process, they will stay in the solution and will not be suppressed. These "unsuppressed" frequencies are connected directly to the shape of the rectangular blur kernel through its exposure-time width. The integral over a periodic signal whose time period that is contained an integer number of times in the exposure-time, will be always zero. This is illustrated in Fig. 15.d where the "unsuppressed" frequencies are shown as temporal sinusoidal signals in one of the pixels of the "ghosting" trail. Those frequencies are upper-bounded by the frame-rate of the output sequence.

Figure 16 shows a quantitative example of the reconstruction and the "ghosting" effect using a simple linear and constant velocity motion of synthesized objects. We performed several such tests with different velocities. We noticed that unlike effects of motion blur (where different velocities of moving objects induce different degrees of motion blur), the temporal frequency of the "ghosts" remained the same regardless of the velocity of the object. We derived an analytical expression for the frequencies that generate the "ghosting" in Appendix D. We have also performed a spectral analysis to the SR system of equations and verified empirically the expression.

The "ghosting" effect, is significantly reduced by the space-time regularization. The

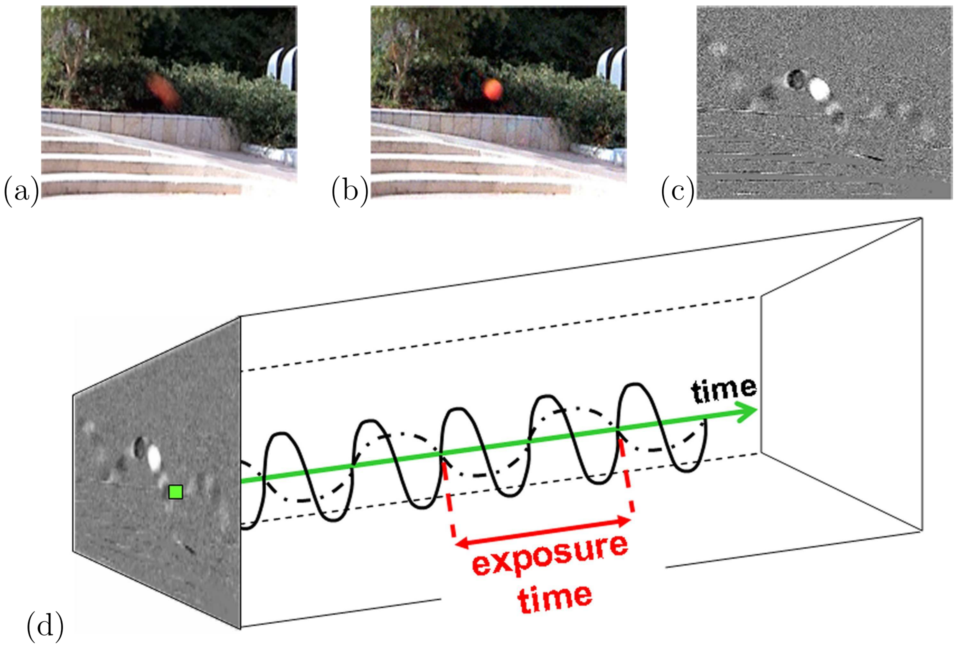


Figure 15: **"Ghosting" effect in video sequence.** *In order to show the "ghosting" effect caused by temporal SR , we applied the algorithm without any regularization to the basket-ball example (see Sec. 3). One input frame of the blurred ball is shown in (a). The temporal SR matching frame is shown in (b). The "ghosting" effect is usually hard to see in a single frame but is observable when watching a video sequence (due to the high sensitivity of the eye to motion). In order to show the effect in a single frame we magnified by factor 5 the difference between the frame in (b) and a matching frame of the background. The resulting "ghosting" trail of the ball is shown in (c). Note that some of the trail values are positive (bright) and some are negative (dark). (d) illustrates that although this effect has spatial artifacts, its origin is purely* temporal. *As explained in the text, due to the rectangular shape of the temporal blur, for each pixel (as the one marked in green) there are some specific temporal frequencies (e.g., the sinusoids marked in black) that will remain in the reconstructed sequence. The reason is that the integral over those frequencies whose temporal wave-length is contained a whole number in the exposure-time, is* 0*, and they will contribute nothing to the minimized error.*

space-time regularization naturally smoothes those trails in regions where no spatial or temporal edges are expected to be. This is why the ghosting effect is barely visible in our example output videos.

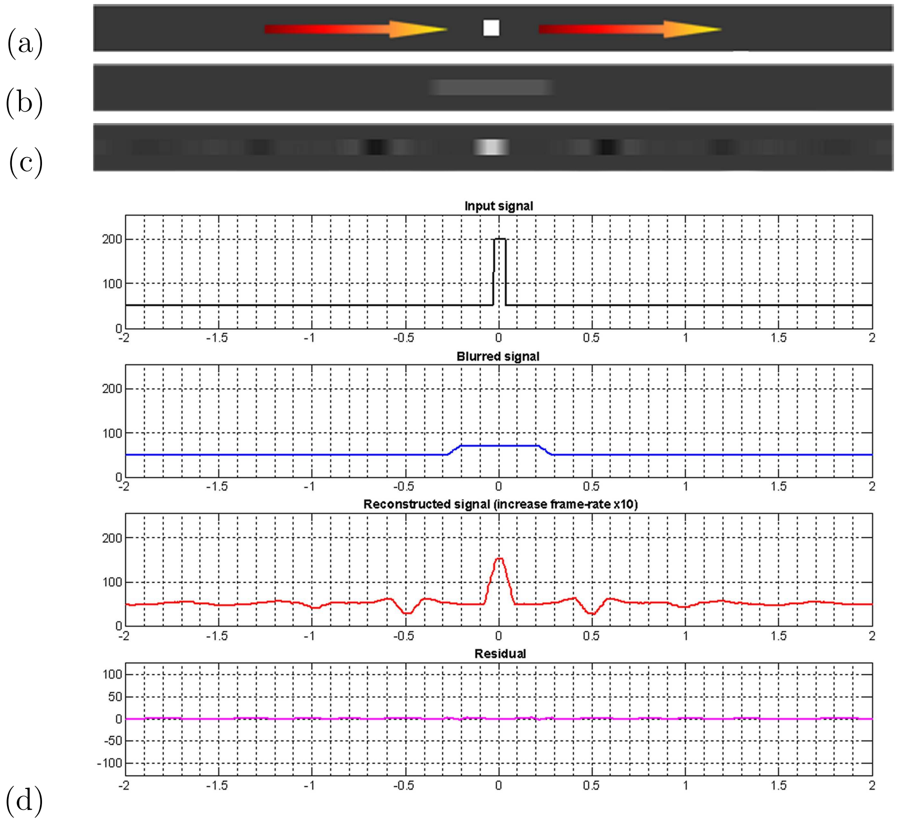


Figure 16: **Empirical measurement of the "ghosting" effect.** *We have synthetically generated a set of 10 video sequences of bright squares moving against a dark background. All squares were moving along a straight line with the same constant velocity. For simplification the sequences were uniformly distributed in time (1/10 frame-time shift between them) and no noise was added. Fig. (a) shows a frame with a moving square, where the arrows represent the direction of motion. Motion-blur in (b) was applied by using an exposure-time of 1/2 frame-time. The 10 sequences were plugged into the algorithm and the frame-rate was increased by factor 10. Fig. (c) shows an output frame where the reconstruction quality and the "ghosting" trail can be seen. Figs. (d)(1)-(3) shows quantitatively the gray levels of (a)-(c), where the x-axis is represented in frame-time units and the y-axis is in gray-levels. In order to validate that the reconstructed sequence is a theoretically valid solution, we re-applied to it the same temporal blur kernel as we used to generate the blurred low resolution sequences (b) from the high resolution ground truth sequences (a) (i.e., exposure-time of 1/2 frame-time). The result looked similar to the blurred sequences (as in (b)), and the residual difference was smaller than 3 gray levels as seen in Fig. (d)(4).*

6.3 Resolving Aliasing

Spatial aliasing is a known phenomenon in images. It occurs when an image is sampled below the Nyquist sampling rate, and high spatial frequencies are "folded" into the low frequencies. The visible result is artificial gray-level patterns at low frequencies (see for example [23] where spatial aliasing is caused by image warping interpolation). Unlike spatial aliasing in images, there are *two* kinds of temporal aliasings in video sequences: “motion aliasing” and “gray-level aliasing”. Both are caused by sampling a rapidly changing function over time below the Nyquist rate (due to the camera frame-rate). *Motion aliasing* is caused by large changes in image coordinates of moving objects over time (i.e., large displacements from frame to frame), while *gray-level* aliasing is caused by strong intensity changes over time at a pixel. Note that gray-level aliasing may occur even for static objects that change their intensity faster than frame-rate (e.g., a flickering flash-light).

Resolving gray-level aliasing in temporal SR means increasing the frame-rate and reducing the “effective” exposure-time in the output sequence (i.e., motion deblurring), just like in image-based SR it means increasing the pixel density and reducing the “effective” PSF width in the output image. Resolving motion aliasing means increasing the frame-rate such that true trajectories of objects are revealed.

In video sequences, changes in motion of pixels are often much “slower” than changes in intensity of pixels. Thus gray-level aliasing usually occurs before motion aliasing occurs. For example, if a bright object is moving faster than 1 pixel/frame on a dark background, then the intensity of a pixel located on the object boundary will change rapidly from bright to dark from one frame to the next. In this case, gray-level aliasing will occur at that pixel. However, as long as the object moves with constant velocity, no motion aliasing will occur. In general, motion aliasing is evident in the presence of high acceleration in the motion of an object (e.g. when there is a sudden change of direction or in radial motion).

Consequently, if we apply temporal SR to video sequences containing motion aliasing, and increase the SR factor gradually, then we will first resolve the motion aliasing by recovering object’s true trajectory, and only then, if the magnification factor is high enough (beyond the bound of Sec. 4.2), will we be able to resolve the gray-level aliasing and recover object’s shape (motion deblurring). As opposed to spatial SR, temporal SR may be meaningful even if motion blur is not resolved, since motion aliasing may still be resolved, thus providing new information about the object trajectory, even if not about its accurate shape. Such an example was shown in Fig. 4, where by increasing the frame-rate by a factor of 3, the true motion of the vent was revealed, even though the “induced” exposure-time remained approximately the same.

A The Induced Temporal "Exposure" in the Output Sequence

In this appendix we derive the expected shape of the output blur kernel, that is "induced" by temporal SR. While we cannot find the exact shape, we can estimate its width, i.e. the "effective" exposure-time of the output sequence. This "effective" exposure-time estimate leads to the lower bound (derived in Sec. 4.1) on the number of input sequences (cameras) needed for obtaining an effective reduction in the motion-blur.

The "induced" high resolution blur kernel:

Eq. (1) describes the relation between the low resolution space-time measurements and the unknown high resolution space-time sequence. As explained in Sec. 4.1, to analyze effects of *temporal* SR, it is enough to look at 1D temporal signals at each pixel. Thus, the low resolution and high resolution sequences of Eq. (1) become 1D temporal signals - S^l_i and S^h, and the space-time blur kernels are now simple continuous exposure-time functions - $B_i = B_{(\tau_i)}$.

The discretization of Eq. (1) then becomes:

$$S^l_i = S^h * W_i \tag{7}$$

where W_i is the discretization of the continuous low resolution blur kernel B_i on the high resolution grid points (see for example the discretization method we chose in Appendix E).

In addition, we can express the input measurements S^l_i by the unknown continuous input signal S convolved with the continuous blur B_i and sampled at the low resolution points t_i:

$$S^l_i = (S * B_i)(t_i) \tag{8}$$

Our goal is to find the high resolution continuous blur kernel denoted by B^{ind}, that is "induced" by the SR algorithm, such that it relates the recovered high resolution sequence S^h at the high resolution points t_j to the continuous space-time volume S (similar to Eq. (8)):

$$S^h = (S * B^{ind})(t_j) \tag{9}$$

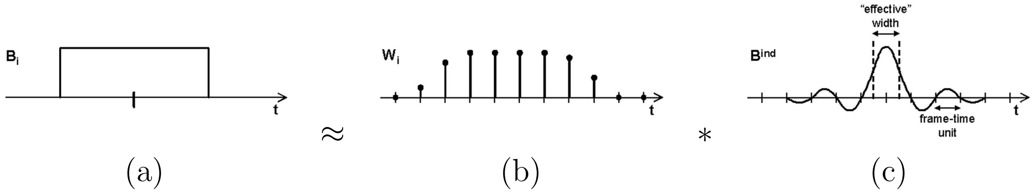
By substituting Eq. (9) into Eq. (7) we get:

$$S^l_i = S * B^{ind} * W_i \tag{10}$$

Comparing Eq. (10) with Eq. (8) yields:

$$B_i = B^{ind} * W_i \tag{11}$$

Since W_i is generated by discretizing B_i at the high resolution points, then B_i is interpolated from W_i with B^{ind} as an "interpolation function". Eq. (11) is illustrated in Fig. 17.



(a) (b) (c)

Figure 17: **High resolution blur kernel as an "interpolation function".** *As explained in the text - the low resolution blur function B_i may be interpolated by its samples W_i on the high resolution grid (the discretization weights of the blur), and some "interpolation function" b^{ind}. For the temporal exposure-time function (a), a typical W_i would look like in (b) and the "induced" high resolution blur kernel would be some "interpolation function" (c) of width* $\sim$1 *frame-time unit.*

An "interpolating function" is used to interpolate accurately a continuous signal from its discrete samples. There are many families of analytical "interpolation functions" that differ by their shape, support and interpolation accuracy (a comprehensive survey and comparison can be found in [21]). Fig. 18 shows a few common functions. Interpolation functions have several properties in common. First, their value is 1 at the origin, and 0 in all other sampling points (marked by the red circles). Second, these functions should be well bandlimited.

It is a known property of the Fourier transform [19], that the width[2] of a band-limited signal is approximately the inverse to its frequency bandwidth:

$$width \; \approx \frac{1}{BW}.$$

Since the bandwidth of an interpolation function is limited by the sampling rate (Nyquist sampling theorem [19]):

$$BW \approx sampling\text{–}rate.$$

Combining the above two approximations we get that the width of the output blur kernel is approximately *1 sampling-unit.* The widths of the functions in Fig. 18 (marked by the green arrows), show that this approximation is tight.

B Solving the Equations

The large dimensionality of the optimization problem was noted in [11, 4, 6] for the simpler case of image-based SR. This is amplified here because of the added temporal dimension. As we saw in Sec. 2.3 the size of the equation system is usually huge, and therefore it is practically infeasible to solve them straightforward.

However the system is very *sparse* and *local.* It is sparse because the space-time blur kernel is small relatively to the entire sequence (typically $5 \times 5 \times \tau$ where $\tau \in [3..10]$), and

[2] Usually defined as the length between signal points that are at half of the maximum amplitude.

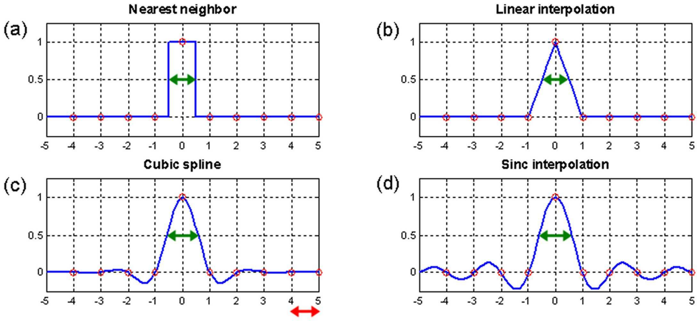


Figure 18: **Interpolation functions and their "effective" width.** *(a) The nearest neighbor interpolation function. (b) The linear interpolation function. (c) The cubic cardinal spline. (d) The sinc function (which is the optimal according to the sampling theorem). The green arrows mark the "effective" width of each of the interpolation functions. W.l.o.g., the width is measured here as the length between points that are at amplitude of 0.5. The red arrow marks the sampling-unit length. As can be seen, typical interpolation function are of approximately 1 sampling-unit length.*

local since each pixel in each frame depends only on its nearby space-time neighborhoods (algebraically, matrix A from Eq. (2) can be arranged such that it is sparse matrix with 75..250 diagonals). Therefore, it can be solved using "box relaxation" [22].

The underlying idea is to divide the high resolution space-time volume into $s$mall overlapping space-time blocks, and thus solve multiple independent and substantially smaller systems of equations. These boxes are solved using an iterative method with an initial state. Following the iterative methods presented in [17, 11], we chose the Conjugate-Gradient algorithm to solve each box as its convergence is relatively fast. The only thing to be aware is the unknown boundaries unknowns of the boxes, which are solved with deficient ("cropped") equations. The solution in these boundaries will therefore be wrong. To prevent this boundary error from diffusing into the "good" variables, we chose the boxes to be overlapping, and throw away the boundary values at the end of each iteration. An ad-hoc optimization was done for the size of the blocks, where the trade-off is between the box size (and the time for each box iteration) and the number of overall iterations. Typical sizes were: $11 \times 11 \times \tau, \tau \in [10..50]$ for the original blocks, and 2 pixel wide boundaries were thrown from all 6 directions of the box to get the resulting "good" variables.

In order to increase the speed of convergence we made a few global sweep iterations, where each iteration started with the results of the previous iteration as the initial state. The first initial state was generated by interpolating the low resolution inputs.

Fig. 19 shows convergence process on the example of Sec. 5.1. Since this example is inherently ill-posed (many more high resolution unknowns than low resolution data points), a strong space-time regularization was needed, and the convergence was relatively slow - up to 20 local and 4 global iterations were needed. An interesting thing happened in this example when several boxes "converged" closely to their initial blurry state already at the

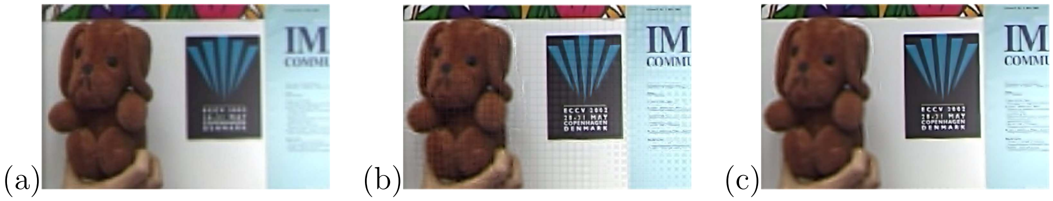


Figure 19: **"Box Relaxation" - Example.** *A demonstration of the equation solution using "box relaxation" on the example of Sec. 5.1 (showing only a small part of the mid frame of the sequence). The boxes in this example were of size* $7 \times 7[pixels] \times 79[frames]$. *(a) The initial state generated by an interpolation. Since the mid frame is far from the still images, the bi-linear interpolation is done only from the corresponding video frame (enlarging it by factor 3). (b) The solution after the first global iteration. Note the "blocky" texture over the image due to bad convergence of box boundaries There are also trails of the moving toy that also did not converge well. (c) After 4 iterations the solution mostly converges.*

first conjugate-gradient iteration. The reason for this behavior was that the residual error *increased* in the first few iterations and only then decreased to the global minimum after a few more iterations. Therefore, we requested a minimum of 5 local box iterations.

C Implementation of Regularization

In this appendix we provide more implementation details on the space-time regularization terms W_j and L_j that were introduced in Eq. (3).

L_j is a matrix which applies the second order derivative operator in the direction j (based on the [-1 2 -1] kernel). Tests were also done with a first order derivative operator but with less success. The matrix W_j contains weights that represent the amount of the desired smoothness for each high resolution pixel in each direction j. It has two roles.

The first is to reflect the relative weighting of the pixels according to an "inverse" function of the second derivative. The second derivative is estimated at the closest input samples, i.e. the exact value for each high resolution pixel is an interpolation of values of the nearest low resolution neighborhood data points[3]. The "inverse" relation that we implemented was:

$$W_j(p^h) = (1 - r_{min})\gamma^{-\alpha * (D_j(p^h))^{\beta}} + r_{min},$$

where r_{min} is the minimal regularization value, $D_j(p^h)$ is the absolute second derivative value at high resolution point p^h, and α, β and γ are parameters that depend on the histograms of D_j. The above expression gives each pixel for each direction a basic weight between 0 and 1, representing the relative regularization strength. The exact parameter values were defined manually for each input data set, so that only real edges would be above the "soft" threshold.

[3] In an iterative algorithm, the derivatives in the current iteration can be updated directly from the reconstructed high resolution sequence from previous iteration.

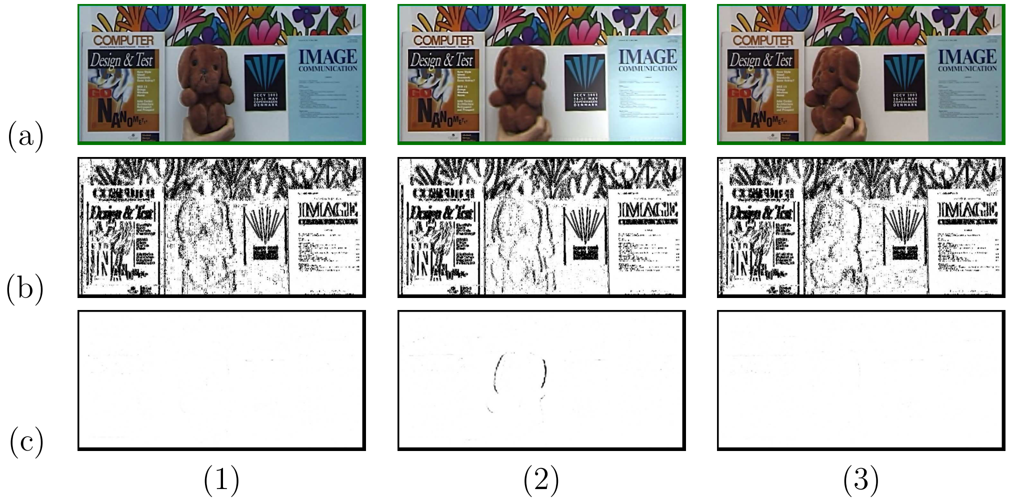


Figure 20: **Space-Time Regularization - Example.** *This figure shows the regularization weights used for controlling the convergence in the extreme example of Sec. 5.1). Images (a1)-(a3) are the first, mid and last output frames. The dominating data at the first and last frames are the two still images of high (spatial) resolution, while in the middle of the sequence the only data comes from the low spatial resolution video frames. The role of the regularization operators is to preserve the high resolution spatial features at static regions along the entire sequence, while avoiding temporal smoothing across dynamic edges. (b1)-(b3) are the corresponding regularization weights in the x direction (W_x from Eq. (3)), where brighter values correspond to larger weights, i.e., stronger smoothing in the x direction. These weights are inversely proportional to the second derivative of the input data, and were calculated from the still images (spatial weights of middle frames, where there were no still images, were generated by interpolation). Analogously (c1)-(c3) are the regularization weights in the t direction, that were calculated as the inverse of the temporal derivatives from the video data. Because nothing was moving at the sequence, (c1) and (c3) contain no temporal derivatives, whereas in the middle of the sequence the toy was moving fast, showing strong temporal derivatives in (c2).*

The second role of W_j is to determine the relative balance between the regularization and the "real" data equations, as well as the relative strength between the temporal and the spatial regularization terms. A global scalar λ_j is used for multiplying the basic weighting terms in W_j independently for each direction j (usually $\lambda_x = \lambda_y \neq \lambda_t$). λ_j are typically small (0.1-0.5) for well-posed cases, where the number of data equations (i.e., low resolution measurements) is larger than the number of unknowns, leading to weak regularization for slight smoothing. However, where the number of unknowns is larger than the number of equations (measurements), then λ_j must be larger. In the extreme case of the example in Fig. 9, where the number of unknowns was larger by factor of 7.5 than the number of measurements, we have used $\lambda_j = 20$. That example was therefore strongly controlled by the space-time regularization. We illustrate the implementation of the weight matrices W_j in Fig. 20.

D Analysis of Temporal "Ghosting"

This appendix analyzes the "ghosting" effects introduced in Sec. 6.2, and provides an analytical description of these "undesired" temporal frequencies. As we mentioned, the "ghosting" effect consists of specific temporal frequencies f_{ghost}, whose time-period $T_{ghost} = \frac{1}{f_{ghost}}$ is contained an integer number of times N_g in the exposure time of the inputs τ_{in}:

$$\tau_{in} = N_g * T_{ghost}, \quad N_g = 1, 2, ...$$

Those frequencies are also upper-bounded by the Nyquist sampling rate induced by the output sequence frame-rate FR_{out}:

$$f_{ghost} \leq \frac{FR_{out}}{2}$$

Therefore,

$$f_{ghost} = \frac{N_g}{\tau_{in}} \ {\scriptstyle [frames/sec]}, \quad N_g = 1, 2, ..., \left\lfloor \frac{FR_{out} \cdot \tau_{in}}{2} \right\rfloor \tag{12}$$

As can be seen, the larger the frame-rate of the output is, the more undesired "ghosting" frequencies will appear in the output. A similar analysis for the evolution of high spatial frequencies in spatial image-based SR algorithms was presented in [3]. A major difference between spatial "ringing" and temporal "ghosting" effects, is their growth rate with the increase of the SR magnification factor: *quadratic* in the spatial 2D case, and *linear* in the temporal 1D case. This is another justification why larger magnification factors are feasibly possible in temporal SR than in spatial SR.

In order to verify empirically that our system of equations indeed behaves according to Eq. (12), we performed a spectral analysis and measured the transfer function of the system. We generated as input data 10 "empty" (blank) sequences (uniformly 0 gray level) of small frames[4]. Those sequences were used as input to the space-time SR algorithm, and the frame-rate of the output was increased by a factor 10. In principle, the true output should also be an "empty" sequence. Since the system of equations is solved iteratively, we took as the initial guess, a sequence that contained one "pure" temporal frequency, i.e. the uniform gray-level of the frames was changing over time as a sine function with the desired frequency. By sweeping the frequency from 0 to $\frac{FR_{out}}{2} = 5FR_{in}$ we measured the frequency transfer function as the ratio between the output signal amplitude and the initial guess sine amplitude. The exact ratio values depend on the kind of iterations we use, the number of iterations, the length of the sequences, and the effect of boundary inaccuracies. However the overall behavior of the transfer function clearly matched the prediction of Eq. (12).

A typical transfer function is shown in Fig. 21.a. The parameters in this example are: $\tau_{in} = \frac{0.5}{FR_{in}}$, $FR_{out} = 10FR_{in}$, $\frac{FR_{out} \cdot \tau_{in}}{2} = 2.5$. Substituting these parameters into Eq. (12) we get:

$$f_{ghost} = 2N_g FR_{in} \ {\scriptstyle [frames/sec]}, \quad N_g = 1, 2, ..., \lfloor 2.5 \rfloor$$

[4] In fact, a single pixel would have been enough since the "ghosting" effect is a completely temporal phenomenon.

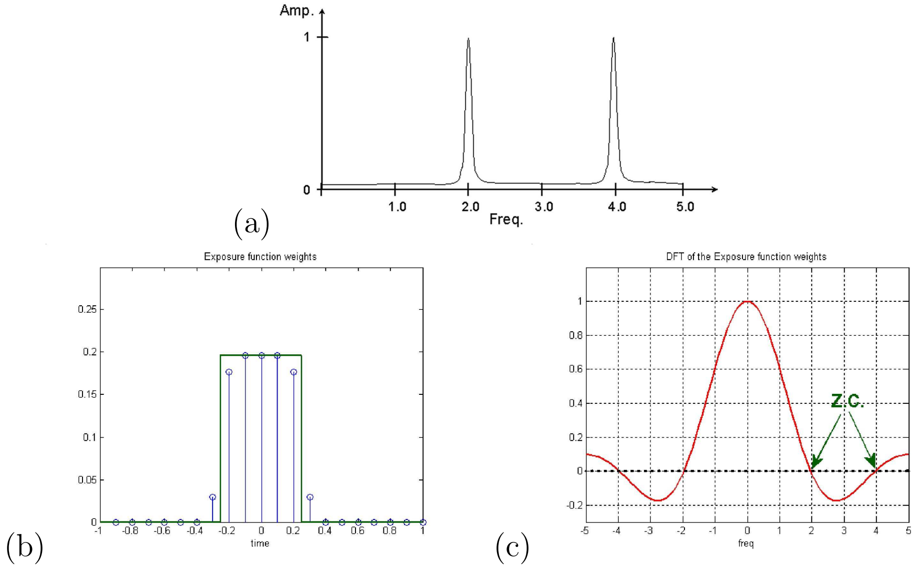


Figure 21: **Blur kernels and the Super-Resolution Transfer Function.** *(a) One measured transfer function of the SR system. The conditions of this example are described in the text. The frequency axis units are relative to FR_{in}. The "ghosting" non-suppressed frequencies are $f^1_{ghost} = 2FR_{in}$ and $f^2_{ghost} = 4FR_{in}$ as predicted by Eq. (12). (b) The rectangular blur function of the example and the discretized kernel weights that where used in the algorithm to represent this function. (c) DFT of the rectangular weights, showing the zero-crossing points at the "ghosting" points. Signals at these frequencies may be added to the input data (or be "born" during the solution process) without changing the output.*

Therefore we get the two "ghosting" frequencies from Fig. 21.a:
$f_{ghost,1} = 2FR_{in}$ and $f_{ghost,2} = 4FR_{in}$.

The rectangular continuous exposure-time function and its discretization (as was used in the algorithm) are shown in Fig. 21.b. The DFT (Discrete Fourier Transform) of the function is shown in Fig. 21.c. The source to the two frequencies is evident from the two zero-crossing points of the graph.

E Non-Isotropic Discretization of the Temporal Blur

Because the output high resolution sequence is discrete, an important part in construction of the SR equations is transforming the continuous temporal blur of the cameras into discrete weights. These discrete weights are applied to the high resolution samples to simulate the imaging process (see illustration in Fig. 22). There are two different possible discrete approximations to the continuous form of Eq. (1) - an isotropic and a non-isotropic approximation. In the non-isotropic approximation, the analytic blur function is transformed and discretized for each input sample separately, and the original input data is not changed. In the isotropic approximation the same weights are used for all samples but all input data

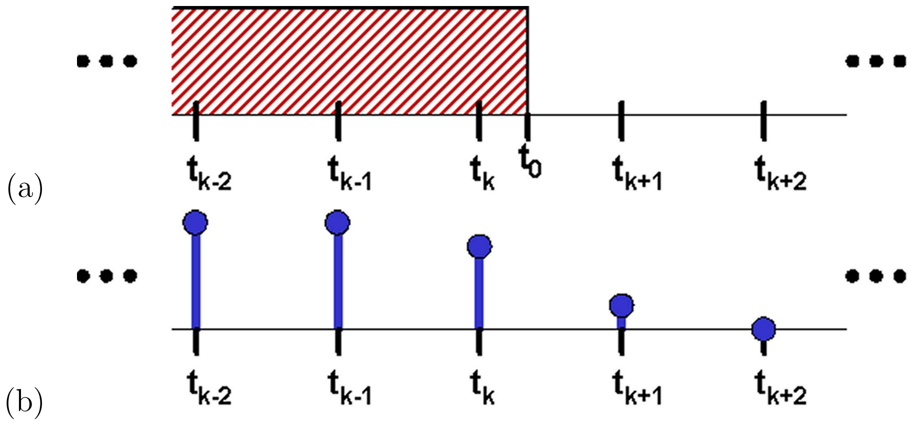


Figure 22: **Blur Kernel Discretization.** *(a) shows the continuous blur function caused by the temporal integration of the camera, referring to one low resolution sample (ending in t_0). (b) is the discrete high resolution weights that simulate the continuous function. See text for a method to calculate the weights.*

is warped to the high resolution coordinate system. This warping process involves interpolation errors and distorts the aliased frequencies that are needed for SR , and therefore the isotropic approximation is less accurate. See [8] for more discussion of the different discretization techniques in the context of image-based SR .

In our implementation we used a non-isotropic approximation in the temporal dimension, and an isotropic approximation in the spatial dimension. Therefore, in the spatial case, the same weights were used for all low resolution points. In the temporal case however, we used several different *sets* of weights[5]. We next explain how to construct this set of weights for the temporal blur discretization (i.e., how to translate the continuous warped function (Fig. 22.a) into a discrete kernel (Fig. 22.b)). [8] showed several discretization methods for Gaussian functions. We adopted one common discretization method based on *linear interpolation.* The key assumption is that the values of the gray levels (of each pixel) change smoothly enough, so that any mid-value between two consecutive high resolution frames can be expressed as a linear combination of the two. Then the continuous integration operation over the continuous space-time volume can be translated to a linear operation on the high resolution frames.

We now show how to calculate the discrete kernel using the example of Fig. 22. If we are interested in the weights corresponding to the low resolution sample whose exposure-time is marked in Fig. 22.a, then the weights on the left to t_k will receive the value 1 and the weights on the right to t_{k+1} will be 0. According to the linear interpolation assumption the intensity of each pixel between t_k and t_{k+1} is: $I(t) = (1-\alpha)I(t_k) + \alpha I(t_{k+1})$, where $\alpha = \frac{t-t_k}{T}$ and T is the high resolution frame-time ($T = t_{k+1} - t_k$). Therefore the weights of

[5] When the temporal transformation is a simple shift in time (which is the case when all input sequences have the same frame-rate), the same weights are used for all frames within a single low resolution sequence. These weights, however, vary from one sequence to another.

t_k and t_{k+1} will be:

$$\begin{array}{rcl} W_k & = & \frac{1}{2}+\frac{1}{T}\int_0^{t_0-t_k}(\frac{T-t'}{T})dt' = \frac{1}{2}+\frac{t_0-t_k}{T}-\frac{(t_0-t_k)^2}{2T^2} \\ W_{k+1} & = & \frac{1}{T}\int_0^{t_0-t_k}(\frac{t'}{T})dt' = \frac{(t_0-t_k)^2}{2T^2} \end{array} \tag{13}$$

Finally, all weights corresponding to a low resolution sample should be normalized so that their sum is 1.

Note that the rectangular blur function is very "non-bandlimited" (much of the energy exists in frequencies that are beyond the Nyquist sampling rate of the high resolution output). Hence this discretization is only an approximation, and the real function cannot be interpolated accurately from these weights. This means that the "induced" high resolution temporal blur kernel (B^{ind} in Apendix A) cannot be an exact interpolator of the low resolution blur kernel.

As we mentioned earlier, the underlying assumption in the above-described discretization is that the gray-level values for each pixel that are between high resolution frames can be represented as a linear combination of the values at same pixels in these frames (i.e., that there is no gray-level aliasing in the high resolution sequence). We call this the "gray-level interpolation" assumption. However this assumption is not always true. Whenever the fast moving object induces temporal frequencies that are higher than the Nyquist frequency defined by the output frame-rate, then these frequencies cannot be interpolated using regular "grid-based" gray level interpolation. This can be solved either by increasing the high resolution frame-rate (up to a certain limit dictated by the number of inputs), or by replacing the grid-based interpolation with "motion based interpolation".

By motion based interpolation we refer to computing intermediate gray levels by linear combination of pixels that are correlated by the motion within a sequence. This is opposed to gray level interpolation where the interpolated values are linear combination of neighboring pixels.

References

[1] M. I. Sezan A. J. Patti and A. M. Tekalp. Superresolution video reconstruction with arbitrary sampling lattices and nonzero aperture time. In *IEEE Trans. on Image Processing*, volume 6, pages 1064–1076, August 1997.

[2] A. Blake B. Bascle and A.Zisserman. Motion deblurring and super-resolution from an image sequence. In *European Conference on Computer Vision*, pages 312–320, 1996.

[3] S. Baker and T. Kanade. Limits on super-resolution and how to break them. In *IEEE Trans. on Pattern Analysis and Machine Intelligence*, volume 24, September 2002.

[4] S. Borman and R. Stevenson. Spatial resolution enhancement of low-resolution image sequences - a comprehensive review with directions for future research. Technical report, Laboratory for Image and Signal Analysis (LISA), University of Notre Dame, Notre Dame, July 1998.

[5] M. Born and E. Wolf. *Principles of Optics.* Permagon Press, 1965.

[6] D. Capel and A. Zisserman. Automated mosaicing with super-resolution zoom. In *CVPR*, pages 885–891, June 1998.

[7] D. Capel and A. Zisserman. Super-resolution enhancement of text image sequences. In *ICPR*, pages 600–605, 2000.

[8] D. P. Capel. Image mosaicing and super-resolution. Ph.D. Thesis, Departement of Engineering Science, University of Oxford, 2001.

[9] Y. Caspi and M. Irani. Parametric sequence-to-sequence alignment, to appear in. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 2001.

[10] G. de Haan. Progress in motion estimation for video format conversion. In *IEEE Transactions on Consumer Electronics*, volume 46, pages 449–459.

[11] M. Elad. Super-resolution reconstruction of images. Ph.D. Thesis, Technion Israel Institute of Technology, December 1996.

[12] H. Greenspan, S. Peled, G. Oz, and N. Kiryati. Super-resolution in MRI. *IEEE Transactions on Medical Imaging*, July 2001.

[13] T.S. Huang and R.Y. Tsai. Multi-frame image restoration and registration. In T.S. Huang, editor, *Advances in Computer Vision and Image Processing*, volume 1, pages 317–339. JAI Press Inc., 1984.

[14] M. Irani and S. Peleg. Improving resolution by image registration. *CVGIP: Graphical Models and Image Processing*, 53:231–239, May 1991.

[15] M. Irani and S. Peleg. Motion analysis for image enhancement: Resolution, occlusion and transparency. In *Journal of Visual Communication and Image Representation*, volume 4, pages 324–335, December 1993.

[16] J. R. Price J. Shin, J. Paik and M.A. Abidi. Adaptive regularized image interpolation using data fusion and steerable constraints. In *SPIE Visual Communications and Image Processing*, volume 4310, January 2001.

[17] R.L. Lagendijk and J. Biemond. *Iterative Identification and Restoration of Images.* Kluwer Academic Publishers, Boston/Dordrecht/London, 1991.

[18] Z. Lin and H. Y. Shum. On the fundamental limits of reconstruction-based super-resolution algorithms. In *IEEE Conference on Computer Vision and Pattern Recognition*, Kauai, Hawaii, September 2001.

[19] A.V. Oppenheim and R.W. Schafer. *Discrete-Time Signal Processing.* Prentice-Hall, Englewood Cliffs, NJ, USA, 1989.

[20] REALVIZ^{TM}. Retimer. www.realviz.com/products/rt, 2000.

[21] P. Thévenaz, T. Blu, and M. Unser. Image interpolation and resampling. In I.N. Bankman, editor, *Handbook of Medical Imaging, Processing and Analysis*, chapter 25, pages 393–420. Academic Press, San Diego CA, USA, 2000.

[22] U. Trottenber, C. Oosterlee, and A. Schüller. *Multigrid.* Academic Press, November 2000.

[23] L. Wang, S. Kang, R. Szeliski, and H. Shum. Optimal texture map reconstruction from multiple views. In *CVPR*, volume 1, pages 347–354, Kauai, Hawaii, September 2001.